Man Pages (1M): Maintenance and Administration Commands

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part No: 805-8067
November 1999

Contents

PREFACE

Overview

A man page is provided for both the naive user and the sophisticated user who is familiar with the Trusted Solaris operating environment and is in need of online information. A man page is intended to answer concisely the question “What does it do?” The man pages in general comprise a reference manual. They are not intended to be a tutorial.

Trusted Solaris Reference Manual

In the AnswerBook2™ and online man command forms of the man pages, all man pages are available:

- Trusted Solaris man pages that are unique for the Trusted Solaris environment
- SunOS 5.7 man pages that have been changed in the Trusted Solaris environment
- SunOS 5.7 man pages that remain unchanged.

The printed manual, the *Trusted Solaris 7 Reference Manual* contains:

- Man pages that have been added to the SunOS operating system by the Trusted Solaris environment
- Man pages that originated in SunOS 5.7, but have been modified in the Trusted Solaris environment to handle security requirements.

Users of printed manuals need both manuals in order to have a full set of man pages, since the *SunOS5.7 Reference Manual* contains the common man pages that are not modified in the Trusted Solaris environment.

Man Page Sections

The following contains a brief description of each section in the man pages and the information it references:

- Section 1 describes, in alphabetical order, commands available with the operating system.
- Section 1M describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes.
- Section 2 describes all of the system calls. Most of these calls have one or more error returns. An error condition is indicated by an otherwise impossible returned value.
- Section 3 describes functions found in various libraries, other than those functions that directly invoke UNIX system primitives, which are described in Section 2 of this volume.
- Section 4 outlines the formats of various files. The C structure declarations for the file formats are given where applicable.
- Section 5 contains miscellaneous documentation such as character set tables.
- Section 6 contains available games and demos.
- Section 7 describes various special files that refer to specific hardware peripherals, and device drivers. STREAMS software drivers, modules and the STREAMS-generic set of system calls are also described.
- Section 9 provides reference information needed to write device drivers in the kernel operating systems environment. It describes two device driver interface specifications: the Device Driver Interface (DDI) and the Driver/Kernel Interface (DKI).
- Section 9E describes the DDI/DKI, DDI-only, and DKI-only entry-point routines a developer may include in a device driver.
- Section 9F describes the kernel functions available for use by device drivers.
- Section 9S describes the data structures used by drivers to share information between the driver and the kernel.

Below is a generic format for man pages. The man pages of each manual section generally follow this order, but include only needed headings. For example, if there are no bugs to report, there is no BUGS section. See the `intro` pages for more information and detail about each section, and `man`(1) for more information about man pages in general.

NAME

This section gives the names of the commands or functions documented, followed by a brief description of what they do.

SYNOPSIS

This section shows the syntax of commands or functions. When a command or file does not exist in the standard path, its full pathname is shown. Options and arguments are alphabetized, with single letter arguments first, and options with arguments next, unless a different argument order is required.

The following special characters are used in this section:

[ ] The option or argument enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified.

. . . Ellipses. Several values may be provided for the previous argument, or the previous argument can be specified multiple times, for example, ‘ "`filename` . . ." .

| Separator. Only one of the arguments separated by this character can be specified at time.

{ } Braces. The options and/or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.

PROTOCOL

This section occurs only in subsection 3R to indicate the protocol description file.

DESCRIPTION

This section defines the functionality and behavior of the service. Thus it describes concisely what the command does. It does not discuss OPTIONS or cite EXAMPLES. Interactive commands, subcommands, requests, macros, functions and such, are described under USAGE.

IOCTL

This section appears on pages in Section 7 only. Only the device class which supplies appropriate parameters to the ioctl (2) system call is called `ioctl` and generates its own heading. `ioctl` calls for a specific device are listed alphabetically (on the man page for that specific device). `ioctl` calls are used for a particular class of devices all of which have an `io` ending, such as `mtio`(7I)

OPTIONS

This lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the SYNOPSIS section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.

OPERANDS

This section lists the command operands and describes how they affect the actions of the command.

OUTPUT

This section describes the output - standard output, standard error, or output files - generated by the command.

RETURN VALUES

If the man page documents functions that return values, this section lists these values and describes the conditions under which they are returned. If a function can return only constant values, such as 0 or –1, these values are listed in tagged paragraphs. Otherwise, a single paragraph describes the return values of each function. Functions declared void do not return values, so they are not discussed in RETURN VALUES.

ERRORS

On failure, most functions place an error code in the global variable `errno` indicating why they failed. This section lists alphabetically all error codes a function can generate and describes the conditions that cause each error. When more than one condition can cause the same error, each condition is described in a separate paragraph under the error code.

USAGE

This section is provided as a guidance on use. This section lists special rules, features and commands that require in-depth explanations. The subsections listed below are used to explain built-in functionality:

- Commands
- Modifiers
- Variables
- Expressions
- Input Grammar

EXAMPLES

This section provides examples of usage or of how to use a command or function. Wherever possible a complete example including command line entry and machine response is shown. Whenever an example is given, the prompt is shown as `example%` or if the user must be root, `example#`. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS and USAGE sections.

ENVIRONMENT VARIABLES

This section lists any environment variables that the command or function affects, followed by a brief description of the effect.

EXIT STATUS

This section lists the values the command returns to the calling program or shell and the conditions that cause these values to be returned. Usually, zero is returned for successful completion and values other than zero for various error conditions.

FILES

This section lists all filenames referred to by the man page, files of interest, and files created or required by commands. Each is followed by a descriptive summary or explanation.

ATTRIBUTES

This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See `attributes`(5) for more information.

SUMMARY OF TRUSTED SOLARIS CHANGES

This section describes changes to a Solaris 7 item by Trusted Solaris software. It is present in man pages that have been modified from Solaris 7 software.

SEE ALSO

This section lists references to other man pages, in-house documentation and outside publications. The references are divided into two sections, so that users of printed manuals can easily locate a man page in its appropriate printed manual.

DIAGNOSTICS

This section lists diagnostic messages with a brief explanation of the condition causing the error.

WARNINGS

This section lists warnings about special conditions which could seriously affect your working conditions. This is not a list of diagnostics.

NOTES

This section lists additional information that does not belong anywhere else on the page. It takes the form of an aside to the user, covering points of special interest. Critical information is never covered here.

BUGS

This section describes known bugs and wherever possible, suggests workarounds.

CHAPTER

Maintenance Commands

16

NAME

Intro – Introduction to maintenance commands and application programs

DESCRIPTION

This section describes Trusted Solaris™ commands that are used chiefly for system maintenance and administration purposes. These commands can be:

- Commands that are unique to and originate in the Trusted Solaris environment, such as `adminvi`(1M), which enables administrators and other users to edit files while preventing certain `vi` actions that present a security risk.
- SunOS 5.7 commands that have been modified to work within the Trusted Solaris security policy, such as `mount`(1M). Man pages for modified commands have been rewritten to remove information that is not accurate for how the command behaves within the Trusted Solaris environment. Modified man pages also add descriptions for any new features, options, and arguments.
- SunOS 5.7 commands that remain unchanged from the Solaris 7 release, such as `ln`(1).

In the Trusted Solaris environment, even if a particular command is installed, not all users may be configured to use that command. Your site's security administrator may restrict the use of any command and may change any command's *security attributes* using *execution profiles.* (Security attributes, execution profiles, and other new Trusted Solaris terms are defined in the `DEFINITIONS` section of `Intro`(1) and explained further in the *Trusted Solaris Administration Overview* and *Trusted Solaris Administrator's Procedures* manuals.) Users who do not have a particular command in any of their execution profiles cannot use that command. Even if a command is in one of a user's execution profiles, that command still may not work as expected because the *label range* or another of the command's *security attributes* specified in the execution profile may limit how the command can be used. If any of the commands described in this section does not work at all or does not work as expected, check with your security administrator.

Because of command restructuring for the Virtual File System architecture, there are several instances of multiple manual pages that begin with the same name. For example, the `mount`, pages – `mount`(1M), `mount_hsfs`(1M), `mount_nfs`(1M), `mount_tmpfs`(1M), and `mount_ufs`(1M). In each such case the first of the multiple pages describes the syntax and options of the generic command, that is, those options applicable to all FSTypes (file system types). The succeeding pages describe the functionality of the FSType-specific modules of the command. These pages list the command followed by an underscore (_) and the FSType to which they pertain. Note that the administrator should not attempt to call these modules directly. The generic command provides a common interface to all of them. Thus the FSType-specific manual pages should

not be viewed as describing distinct commands, but rather as detailing those aspects of a command that are specific to a particular FSType.

Trusted Solaris Information Label Changes

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as ADMIN_LOW.

Objects still have CMW labels, and CMW labels still include the IL component: IL[SL]; however, the IL component is fixed at ADMIN_LOW.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return ADMIN_LOW.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs cannot be set on any objects.
- Sensitivity labels, not information labels, display on printer banners.
- Options related to information labels in the label_encodings(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

- IL–related privileges are no longer used.
- In auditing, the ilabel token is recorded as ADMIN_LOW, when it is recorded. The audit event numbers 519 (AUE_OFLOAT), 520 (AUE_SFLOAT), and 9036 (AUE_iil_change) continue to be reserved, but those events are no longer recorded.

COMMAND SYNTAX

Unless otherwise noted, commands described in this section accept options and other arguments according to the following syntax:

name [*option*(*s*)] [*cmdarg*(*s*)]

where:

name The name of an executable file.

option – *noargletter*(*s*) or,

– *argletter*< >*optarg*

where < > is optional white space.

noargletter	A single letter representing an option without an argument.
argletter	A single letter representing an option requiring an argument.
optarg	Argument (character string) satisfying preceding *argletter*.
cmdarg	Pathname (or other command argument) *not* beginning with – or, – by itself indicating the standard input.

Rules for the Display and Entering of Labels

When entering labels on the command line in a UNIX shell, follow these rules. For rules for entering labels in graphical user interfaces, see `Rules for the Display and Entering of Labels`. For rules for entering labels in configuration files, see RULES FOR INCLUDING LABELS IN A CONFIGURATION FILE in `Intro`(4).

Enter a sensitivity label (SL), information label (IL), or clearance, in text in the form:

```
{ + } { classification } { { +|– }word } …
```

Items in curly brackets are optional. A vertical bar (|) represents a choice between two items. Items followed by an ellipsis may be repeated zero or more times. Leading and trailing white space is ignored. Items may be separated by blanks, tabs, commas or slashes (/).

The system always displays labels in uppercase. Users may enter labels in any combination of uppercase and lowercase.

The classification part of the label must be a valid classification name as defined in `label_encodings`(4). Classification names may contain embedded blanks or punctuation, if they are so defined in `label_encodings`. Short and long forms of classification names may be used interchangeably.

The words (*compartments* and *markings*) used in labels must be valid words as defined in `label_encodings`. Words may contain embedded blanks or punctuation if they are so defined in `label_encodings`.

Short and long forms of words may be used interchangeably. Words may be specified in any order; however they are processed left to right, so that where words conflict with each other, the word furthest to the right takes precedence.

You may used plus and minus signs when modifying an existing label to turn on or off the compartments and markings associated with the words.

A CMW label is represented in text in the form:

```
{ INFORMATION LABEL } { [ SENSITIVITY LABEL ] }
```

Items in curly brackets are optional. Leading and trailing white space is ignored. Items may be separated by blanks, tabs, commas, or slashes (/). Note

that information labels are no longer supported — see `Trusted Solaris Information Label Changes` in `Intro` for a fuller discussion.

EXAMPLES

EXAMPLE 1 Using quotes in labels

On the command line, enclose any label with more than one word in double quotes because, without quotes, a second word or letter separated by a space is interpreted as a second argument. Enclose labels containing [and] characters in quotes to suppress the shell's use of those characters in filename substitution.

```
$ setlabel "ADMIN_LOW[ts a b]" somefile
$ setlabel "admin_low[ts,a,b]" somefile
$ setlabel "admin_low[ts/a     b]" somefile
```

EXAMPLE 2 Using case in labels

Use any combination of upper and lowercase letters. You may separate items in a label with blanks, tabs, commas or slashes (/). Do not use any other punctuation.

```
$ setlabel –s SECRET somefile
```

EXAMPLE 3 Using brackets in labels

When entering a full CMW label, enter the IL first, followed by the SL in brackets.

```
Information Label[Sensitivity Label]
```

When entering an SL with a command option that sets the SL, you do not need to use brackets around the SL.

```
$ setlabel –s "TOP SECRET A B" somefile
```

EXAMPLE 4 Setting a label

To set somefile's SL to SECRET A.

```
$ setlabel "[Secret a]" somefile
```

To turn on compartment B in *somefile*'s SL.

```
$ setlabel –s  +b somefile
```

To turn off compartment A in *somefile*'s SL.

```
$ setlabel –s –A somefile
```

TRUSTED SOLARIS DIFFERENCES

The responsibilities and privileges of the super-user have been divided among several administrative roles. When a man page that has not been modified for the Trusted Solaris system states that super-user is required to execute a certain command or option, remember that one or more privileges are required instead. The site's security administrator can perform privilege debugging [see `runpd`(1M)] to find out which privileges are needed and can then decide to give the privilege to the command after assessing whether the command and any users set up to use that command can make use of the privilege in a manner that does not violate the site's security policy.

The ability of the UNIX super-user to bypass access restrictions, to execute restricted commands, and to use some command options not available to other users has been replaced with the *profile mechanism*, which allows the security administrator to assign to various users different sets of commands and to assign

different privileges to the commands using *execution profiles.* When a command or one of its options needs a privilege in order to succeed, that privilege is a *required* privilege; if the required privilege is not given to the command in a user's execution profile by the security administrator, the command will not work. Required privileges are indicated on the man page with the words "must have," as shown in this sentence: "The `ifconfig`(1M) command must have the `sys_net_config` privilege to modify network interfaces."

In other cases, when the command is designed to work within security policy and it fails when certain DAC or MAC checks are not passed, an *override* privilege may be assigned at the security administrator's discretion. On man pages, the names of privileges that may be used to override access restrictions are given in the `ERRORS` section. The override privileges that may be given to bypass DAC or MAC restrictions on files or directories are given below:

The DAC override privileges are `file_dac_read` and `file_dac_write`. If a user does not have DAC access to a file, the security administrator may assign one or both of these privileges to the command, depending on whether read or write access or both are desired. The MAC override privileges are `file_mac_read` and `file_mac_write`. If a user does not have MAC access to a file, the security administrator may assign one or both of these privileges to the command, depending on whether read or write access or both are desired.

Besides being able to assign an override privilege, the security administrator has other options. For example, to avoid the use of privilege the security administrator may specify that the command will execute with another user's ID (usually the root ID 0) or group ID, one that allows access to the file or directory based on its permissions or its ACL.

To find out how privileges are made available to commands and to find out exactly which tasks, commands, and privileges are assigned to each of the roles by means of execution profiles shipped with the default system, see the *Trusted Solaris Administrator's Procedures.*

Also, check with your security administrator to find out which roles are configured at your site and if any of the roles have been reconfigured to suit your site's security policy.

SUMMARY OF TRUSTED SOLARIS CHANGES

Commands may not work as expected in the Trusted Solaris environment because Trusted Solaris administrators may limit the conditions under which commands may be accessed by each user or restrict commands from being accessed by certain users.

The printed *Trusted Solaris 7 Reference Manual* contains only the Trusted Solaris original and modified (from the Solaris environment) man pages. The online set of man pages viewed by the `man` command accesses all man pages; AnswerBook2™ can access all man pages in the AnswerBook2 collections.

For a fuller description, see `Trusted Solaris Manual Page Display` in `Intro`(1). The `SEE ALSO` man page heading has been subdivided to help users of the printed manual locate a referenced man page.

Besides the usual UNIX DAC checks performed when a process acting on behalf of a user attempts to access a file or directory, *mandatory access* checks also must be passed. For each possible type of access failure, a specific override *privilege* may be assigned to the command at the security administrator's discretion.

When a `SUMMARY OF TRUSTED SOLARIS CHANGES` is provided on a modified man page, it is intended as a convenience to summarize for you the major changes all in one place. Do not rely on the `SUMMARY OF TRUSTED SOLARIS CHANGES` alone, but also read the entire man page.

ATTRIBUTES

See `attributes`(5) in the *SunOS 5.7 Reference Manual* for a discussion of the attributes listed in this section.

SEE ALSO

Commands that are listed under the `Trusted Solaris 7 Reference Manual` heading in the SEE ALSO section are commands that have been changed or added in the Trusted Solaris environment. Commands that are listed under the `SunOS 5.7 Reference Manual` heading in the SEE ALSO section are commands that are unchanged in the Trusted Solaris environment. If you are using printed manuals, refer to the *SunOS 5.7 Reference Manual* for Solaris commands that are unchanged in the Trusted Solaris environment.

Trusted Solaris 7 Reference Manual

`runpd`(1M)

Trusted Solaris Administration Overview, *Trusted Solaris Administrator's Procedures*

SunOS 5.7 Reference Manual

`getopt`(1), `getopt`(3C), `attributes`(5)

DIAGNOSTICS

Upon termination, each command returns `0` for normal termination and non-zero to indicate troubles such as erroneous parameters, bad or inaccessible data, or other inability to cope with the task at hand. It is called variously "exit code," "exit status," or "return code," and is described only where special conventions are involved.

NOTES

Unfortunately, not all commands adhere to the standard syntax.

Name	Description
`accept`(1M)	Accept or reject print requests
`add_allocatable`(1M)	Add entries to allocation databases and create ancillary file
`add_drv`(1M)	Add a new device driver to the system

adminvi(1M)	Edit text with restrictions
allocate(1M)	Device Allocation
arp(1M)	Address resolution display and control
atohexlabel(1M)	Convert a character-coded label to its hexadecimal equivalent
audit(1M)	Control the behavior of the audit daemon
audit_startup(1M)	Audit subsystem initialization script
audit_warn(1M)	Audit daemon warning script
auditconfig(1M)	Configure auditing
auditd(1M)	Audit daemon
auditreduce(1M)	Merge and select audit records from audit trail files
auditstat(1M)	Display kernel audit statistics
automount(1M)	Install automatic mount points
automountd(1M)	Autofs mount/unmount daemon
autopush(1M)	Configures lists of automatically pushed STREAMS modules
bootparamd(1M)	See rpc.bootparamd(1M)
bsmconv(1M)	Enable or disable the Basic Security Module (BSM)
bsmunconv(1M)	See bsmconv(1M)
chk_encodings(1M)	Check the label encodings file syntax
chroot(1M)	Change root directory for a command
clist(1M)	See pfsh(1M)
cron(1M)	Clock daemon
deallocate(1M)	Device deallocation
device_clean(1M)	Device clean programs
devpolicy(1M)	Configure device policy
dfmounts(1M)	Display mounted resource information

dfshares(1M)	list available resources from remote or local systems
dispadmin(1M)	Process scheduler administration
dl_booting(1M)	Inform the kernel that a machine is in the state of disklessly booting or in the normal state
dl_restore(1M)	See dl_booting(1M)
dminfo(1M)	Report information about a device entry in a device maps file
drvconfig(1M)	Configure the /devices directory
du(1M)	Summarize disk usage
eeprom(1M)	EEPROM Display and Load Utility
format(1M)	Disk partitioning and maintenance utility
fsdb_ufs(1M)	ufs File System Debugger
ftpd(1M)	See in.ftpd(1M)
fuser(1M)	Identify processes using a file or file structure
getfsattr(1M)	Display file system security attributes
getfsattr_ufs(1M)	Display file system security attributes
halt(1M)	Stop the processor
hextoalabel(1M)	Convert a hexadecimal label to its character-coded equivalent
ifconfig(1M)	Configure network interface parameters
in.ftpd(1M)	File transfer protocol server
in.named(1M)	Internet domain name server
in.rarpd(1M)	DARPA Reverse Address Resolution Protocol server
in.rdisc(1M)	Network router discovery daemon
in.rexecd(1M)	Remote execution server
in.rlogind(1M)	Remote login server
in.routed(1M)	Network routing daemon
in.rshd(1M)	Remote shell server

in.tftpd(1M)	Internet Trivial File Transfer Protocol server
inetd(1M)	Internet services daemon
init(1M)	Process control initialization
install(1M)	Install commands
list_devices(1M)	List allocatable devices
lockd(1M)	Network lock daemon
lpadmin(1M)	Configure the LP print service
lpfilter(1M)	Administer filters used with the LP print service
lpforms(1M)	Administer forms used with the LP print service
lpmove(1M)	Move print requests
lpsched(1M)	Start the LP print service
lpshut(1M)	Stop the LP print service
lpsystem(1M)	Register remote systems with the print service
lpusers(1M)	Set printing queue priorities
modload(1M)	Load a kernel module
modunload(1M)	Unload a module
mount(1M)	Mount or unmount file systems and remote resources
mount_hsfs(1M)	Mount hsfs file systems
mount_nfs(1M)	Mount remote NFS resources
mount_pcfs(1M)	Mount pcfs file systems
mount_tmpfs(1M)	Mount tmpfs file systems
mount_ufs(1M)	Mount ufs file systems
mountall(1M)	Mount, unmount multiple file systems
mountd(1M)	Server for NFS mount requests and NFS access checks
named(1M)	See in.named(1M)
ndd(1M)	Get and set driver configuration parameters
netstat(1M)	Show network status

newsecfs(1M)	See setfsattr(1M)
nfsd(1M)	NFS daemon
nfsstat(1M)	NFS statistics
nis_cachemgr(1M)	NIS+ utility to cache location information about NIS+ servers
nisd(1M)	See rpc.nisd(1M)
nisd_resolv(1M)	See rpc.nisd_resolv(1M)
nispasswdd(1M)	See rpc.nispasswdd(1M)
nispopulate(1M)	Populate the NIS+ tables in a NIS+ domain
nissetup(1M)	Initialize a NIS+ domain
nscd(1M)	Name service cache daemon
nslookup(1M)	Query name servers interactively
nstest(1M)	DNS test shell
pbind(1M)	Control and query bindings of processes to processors
pfsh(1M)	Profile shell
poweroff(1M)	See halt(1M)
praudit(1M)	Print contents of an audit trail file
prtconf(1M)	Print system configuration
psradm(1M)	Change processor operational status
rarpd(1M)	See in.rarpd(1M)
rdate(1M)	Set system date from a remote host
rdisc(1M)	See in.rdisc(1M)
reboot(1M)	Restart the operating system
reject(1M)	See accept(1M)
rem_drv(1M)	Remove a device driver from the system
remove_allocatable(1M)	Remove entries from allocation databases and delete ancillary file
rexecd(1M)	See in.rexecd(1M)
rlogind(1M)	See in.rlogind(1M)

route(1M)	Manually manipulate the routing tables
routed(1M)	See in.routed(1M)
rpc.bootparamd(1M)	Boot parameter server
rpc.getpeerinfod(1M)	Getpeerinfo service daemon
rpc.nisd(1M)	NIS+ service daemon
rpc.nisd_resolv(1M)	NIS+ service daemon
rpc.nispasswdd(1M)	NIS+ password update daemon
rpc.tbootparamd(1M)	Trusted Solaris boot parameter server
rpcbind(1M)	Universal addresses to RPC program number mapper
rpcinfo(1M)	Report RPC information
rshd(1M)	See in.rshd(1M)
runpd(1M)	Run a command for privilege debugging
rwall(1M)	Write to all users over a network
sendmail(1M)	Send mail over the internet
setaudit(1M)	Run a command with the audit mask set
setfsattr(1M)	Set security attributes on an existing or newly created file system
setmnt(1M)	Establish mount table
setuname(1M)	Change machine information
share(1M)	Make local resource available for mounting by remote systems
share_nfs(1M)	Make local NFS file systems available for mounting by remote systems
shareall(1M)	Share, unshare multiple resources
showmount(1M)	Show all remote mounts
snoop(1M)	Capture and inspect network packets
spray(1M)	Spray packets
statd(1M)	Network status monitor
swap(1M)	Swap administrative interface

sysdef(1M)	Output system definition
sysh(1M)	System shell
tbootparam(1M)	Send a request to rpc.tbootparamd to inform it that a host is in normal (labeled) state now
telinit(1M)	See init(1M)
tftpd(1M)	See in.tftpd(1M)
tnchkdb(1M)	Check file syntax of trusted network databases
tnctl(1M)	Configure Trusted Solaris network-daemon control parameters
tnd(1M)	Trusted network daemon
tninfo(1M)	Print information and statistics about kernel-level network
tokmapctl(1M)	Configure token-mapping daemon
tokmapd(1M)	Token-mapping daemon
traceroute(1M)	Print the route packets take to network host
uadmin(1M)	Administrative control
umount(1M)	See mount(1M)
umountall(1M)	See mountall(1M)
unshare(1M)	Make local resource unavailable for mounting by remote systems
unshare_nfs(1M)	Make local NFS file systems unavailable for mounting by remote systems
unshareall(1M)	See shareall(1M)
updatehome(1M)	Update the home directory copy and link files for the current label
writeaudit(1M)	Write an audit record

NAME

accept, reject – Accept or reject print requests

SYNOPSIS

accept *destination…*

reject [–r *reason*] *destination…*

DESCRIPTION

`accept` allows the queueing of print requests for the named destinations.

`reject` prevents queueing of print requests for the named destinations.

Use `lpstat –a` to check if destinations are accepting or rejecting print requests.

`accept` and `request` must be run on the print server; they have no meaning on a client system.

OPTIONS

The following options are supported for `reject` .

–r *reason* — Assigns a reason for rejection of print requests for *destination* . Enclose *reason* in quotes if it contains blanks. *reason* is reported by `lpstat –a` . By default, *reason* is `unknown reason` for existing destinations, and `new destination` for destinations added to the system but not yet accepting requests.

OPERANDS

The following operands are supported.

destination — The name of the destination accepting or rejecting print requests. Destination specifies the name of a printer or class of printers [see `lpadmin`(1M)]. Specify *destination* using atomic name. See `printers.conf`(4) for information regarding the naming conventions for atomic names.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/var/spool/lp/*` — LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
CSI	Enabled (see `NOTES`)

SEE ALSO

Trusted Solaris 7 Reference Manual

`enable`(1) , `lp`(1) , `lpstat`(1) , `lpadmin`(1M) , `lpsched`(1M)

SunOS 5.7 Reference Manual

`printers.conf` (4), `attributes`(5)

NOTES

`accept` and `reject` only affect queuing on the print server's spooling system. Requests made from a client system remain queued in the client system's queuing mechanism until they are cancelled or accepted by the print server's spooling system.

`accept` is CSI -enabled except for the *destination* name.

NAME

add_allocatable – Add entries to allocation databases and create ancillary file

SYNOPSIS

/usr/sbin/add_allocatable [–f] –n *name* –t *type* –d *device-list* [–l *minSL*] [–h *maxSL*] [–a *authorization*] [–c *clean*]

DESCRIPTION

`add_allocatable` creates or updates database entries for allocatable devices and certain non-allocatable devices and creates an ancillary file used by the `allocate`(1M) command to control access to an allocatable device. `add_allocatable` updates the `device_allocate`(4) and `device_maps`(4) databases, and it creates the ancillary file in `/etc/security/dev` for the specified device. The database entries and the ancillary file are needed when devices are user-allocatable. The database entries are also needed for the frame buffer and printers because the label ranges for these non-allocatable devices are managed by the device allocation mechanism.

`add_allocatable` can be used in shell scripts, such as installation scripts for driver packages, to automate the administrative work of setting up a new device.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–f — Force an update of an already-existing entry with the specified information. `add_allocatable` exits with an error if this option is not specified when an entry with the specified device name already exists.

–n *name* — Puts a device name *name* into the device-name fields of the device's entries in `device_allocate` and `device_maps` and create a zero-length ancillary file of that name in `/etc/security/dev`.

–t *type* — Puts a device type *type* into the device-type fields of the device's entries in `device_allocate` and `device_maps`.

–d *device-list* — Puts the device special file names specified in *device-list* into the device-list field of the device's entry in `device_maps`. The list of devices must be separated by white space, and the list must be quoted because white spaces are treated by the shell as argument delimiters.

–l *minSL* — Puts the minimum sensitivity label *minSL* of the device into the dev-minimum field of the device's entry in `device_allocate`. The default sensitivity label

ADMIN_LOW is used when this optional argument is not specified.

–h *maxSL* Puts the maximum sensitivity label *maxSL* into the dev-maximum field of the device's entry in device_allocate. The default sensitivity label ADMIN_HIGH is used when this optional argument is not specified.

–a *authorization* Puts one or more authorizations or other characters specified in *authorization* into the device authorization field of the device's entry in device_allocate. When more than one authorization is specified, the list must be separated by commas and must be quoted. When the device is not allocatable, *authorization* is specified with an asterisk (*) and must be quoted. When the device is allocatable and is allocatable by any user, *authorization* is specified with the at sign (@) and must be quoted. When this optional argument is not specified, the default value '@' is used, and the device is allocatable by any user with no authorization required.

–c *clean* Puts the device_clean(1M) program *clean* into the device-clean field of the device's entry in device_allocate(4). The default value /bin/true is used when this optional argument is not specified.

ERRORS

When successful, add_allocate returns an exit status of 0 (true). add_allocate returns a nonzero exit status in the event of an error. The exit codes are as follows:

1 Invocation syntax error

2 Unknown system error

3 A device_allocate entry already exists. This error occurs only when the –f option is not specified.

4 Permission denied. User does not have DAC or MAC access to database.

FILES

/etc/security/device_allocate
Mandatory access control file for devices

/etc/security/tsol/device_maps
List of physical devices associated with a device name and type

SEE ALSO

Trusted Solaris 7 Reference Manual	`allocate`(1M), `device_allocate`(4), `device_clean`(1M), `device_maps`(4), `remove_allocatable`(1M)
SunOS 5.7 Reference Manual	`attributes`(5)

NAME

add_drv – Add a new device driver to the system

SYNOPSIS

add_drv [–b *basedir*] [–c *class_name*] [–i '*identify_name*...'] [–m '*permission*','...']
[–n] [–f] [–v] *device_driver*

DESCRIPTION

The `add_drv` command is used to inform the system about newly installed device drivers.

Each device on the system has a name associated with it. This name is represented by the `name` property for the device. Similarly, the device may also have a list of driver names associated with it. This list is represented by the `compatible` property for the device.

The system determines which devices will be managed by the driver being added by examining the contents of the `name` property and the `compatible` property (if it exists) on each device. If the value in the `name` property does not match the driver being added, each entry in the `compatible` property is tried, in order, until either a match occurs or there are no more entries in the `compatible` property.

In some cases, adding a new driver may require a reconfiguration boot. See the `NOTES` section.

OPTIONS

–b *basedir* — Installs the driver on the system with a root directory of *basedir* rather than installing on the system executing `add_drv`. This option is typically used in package post-installation scripts when the package is not being installed on the system executing the `pkgadd` command. The system using *basedir* as its root directory must reboot to complete the driver installation.

–c *class_name* — The driver being added to the system exports the class *class_name*.

–i '*identify_name*' — A white-space separated list of aliases for the driver *device_driver*.

–m '*permission*' — Specify the file system permissions for device nodes created by the system on behalf of *device_driver*.

–n — Do not try to load and attach *device_driver*, just modify the system configuration files for the *device_driver*.

–f — Normally if a reconfiguration boot is required to complete the configuration of the driver into the system, `add_drv` will not add

the driver. The force flag forces `add_drv` to add the driver even if a reconfiguration boot is required. See the `–v` flag.

`–v` The verbose flag causes `add_drv` to provide additional information regarding the success or failure of a driver's configuration into the system. See the EXAMPLES section.

EXAMPLES

EXAMPLE 1 Adding The SUNW, Example Driver to the System

The following example adds the `SUNW,example` driver to the system, with an alias name of `SUNW,alias`. It assumes the driver has already been copied to `/usr/kernel/drv`.

```
example# add_drv –m '* 0666 bin bin','a 0644 root sys' \
        –i 'SUNW,alias' SUNW,example
```

Every minor node created by the system for the `SUNW,example` driver will have the permission `0666`, and be owned by user `bin` in the group `bin`, except for the minor device `a`, which will be owned by `root`, group `sys`, and have a permission of `0644`.

EXAMPLE 2 Adding The Driver To The Client `/export/root/sun1`

The following example adds the driver to the client `/export/root/sun1`. The driver is installed and loaded when the client machine, `sun1`, is rebooted. This second example produces the same result as the first, except the changes are on the diskless client, `sun1`, and the client must be rebooted for the driver to be installed.

```
example# add_drv –m '* 0666 bin bin','a 0644 root sys' \
        –i 'SUNW,alias' -b /export/root/sun1 \
 SUNW,example
```

EXAMPLE 3 Adding A Driver For A Device That Is Already Managed By An Existing Driver

The following example illustrates the case where a new driver is added for a device that is already managed by an existing driver. Consider a device that is currently managed by the driver `dumb_framebuffer`. The `name` and `compatible` properties for this device are as follows:

```
name="display"
compatible="whizzy_framebuffer", "dumb_framebuffer"
```

If `add_drv` is used to add the `whizzy_framebuffer` driver, the following will result.

```
example# add_drv whizzy_framebuffer
Error: Could not install driver (whizzy_framebuffer)
Device managed by another driver.
```

If the `–v` flag is specified, the following will result.

```
example# add_drv -v whizzy_framebuffer
Error: Could not install driver (whizzy_framebuffer)
Device managed by another driver.
Driver installation failed because the following
entries in /devices would be affected:
```

```
        /devices/iommu@f,e0000000/sbus@f,e0001000/display[:*]
        (Device currently managed by driver "dumb_framebuffer")

  The following entries in /dev would be affected:

        /dev/fbs/dumb_framebuffer0
```

If the –v and –f flags are specified, the driver will be added resulting in the following.

```
  example# add_drv -vf whizzy_framebuffer
  A reconfiguration boot must be performed to complete the
  installation of this driver.

  The following entries in /devices will be affected:

        /devices/iommu@f,e0000000/sbus@f,e0001000/display[:*]
        (Device currently managed by driver "dumb_framebuffer"

  The following entries in /dev will be affected:

        /dev/fbs/dumb_framebuffer0
```

The above example is currently only relevant to devices exporting a generic device name.

EXIT STATUS

`add_drv` returns `0` on success and `1` on failure.

FILES

`/kernel/drv`	Boot device drivers
`/usr/kernel/drv`	Other drivers that could potentially be shared between platforms
`/platform/'uname –i'/kernel/drv`	Platform-dependent drivers
`/etc/driver_aliases`	Driver aliases file
`/etc/driver_classes`	Driver classes file
`/etc/minor_perm`	Minor node permissions
`/etc/name_to_major`	Major number binding

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the `PRIV_SYS_DEVICES` privilege. This command is intended to be invoked at `ADMIN_LOW` with effective user ID `0`; if invoked by other users, the command needs the `file_dac_write` privilege

SEE ALSO

Trusted Solaris 7 Reference Manual

`drvconfig`(1M), `rem_drv`(1M)

SunOS 5.7 Reference Manual

`boot`(1M), `devlinks`(1M), `disks`(1M), `kernel`(1M), `modinfo`(1M), `ports`(1M), `tapes`(1M), `driver.conf`(4), `system`(4), `attributes`(5), `ddi_create_minor_node`(9F)

Writing Device Drivers

NOTES

Aliases may require quoting (with double-quotes) if they contain numbers.

It is possible to add a driver for a device already being managed by a different driver, where the driver being added appears in the device's `compatible` list before the current driver. In such cases, a reconfiguration boot is required (see `boot`(1M) and `kernel`(1M)). After the reconfiguration boot, device nodes in `/devices`, entries in `/dev`, and references to these files may no longer be valid (see the `–v` flag). If a reconfiguration boot would be required to complete the driver installation, `add_drv` will fail unless the `–f` option is specified. See `Example 3` in the `EXAMPLES` section.

BUGS

`add_drv` will accept a full pathname for *device_driver*. However, the kernel does not use the full pathname; it only uses the final component and searches the internal driver search path for the driver. This can lead to the kernel loading a different driver than expected.

For this reason, it is *not* recommended that you use `add_drv` with a full pathname. See `kernel`(1M) for more information on the driver search path.

NAME

adminvi – Edit text with restrictions

SYNOPSIS

adminvi *filename…*

DESCRIPTION

The admin text editor is a modified version of `vi` that provides a restricted text-editing environment. `adminvi` provides all the capabilities of `vi` except that `adminvi` does not allow the user to execute shell commands or to write any files other than the files specified on the command line.

OPTIONS

Refer to the `vi`(1) man page for a complete list of options. `adminvi` modifies the following options:

`–x`	Heuristic file encryption is not allowed.
`–C`	Forced file encryption is not allowed.
`–L`	Listing the names of files saved as the result of an editor or system crash is not allowed.
`–r` *filename*	Recovering files saved as the result of an editor or system crash is not allowed.
filename	A filename must be specified.

USAGE

Refer to the `vi`(1) man page for a complete usage description.

`adminvi` modifies `vi` commands to prevent use of the `!` operator and shell metacharacters in filenames given to commands such as `:r` and `:so`.

Commands

The actions of these commands are changed:

`:!`	The command to execute a shell command is not allowed.
`:C`	The forced-encryption command is not allowed.
`:cd`, `:chdir`	The change directory command is not allowed.
`:crypt`, `:X`	The heuristic-encryption command is not allowed.
`:e`	If the command to change the file being edited specifies a filename other than the filenames that were given on the `adminvi` command line, the file is edited in read-only mode.
`:pre`	The command to preserve the edit buffers is not allowed.
`:rec`	The command to recover preserved edit buffers is not allowed.
`:sh`	The command to run a shell is not allowed.
`:w`	This command accepts only the filenames that were given on the `adminvi` command line.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

SunOS 5.7 Reference Manual

`vi`(1), `attributes`(5)

NOTES

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

allocate – Device Allocation

SYNOPSIS

allocate [–s] [–U *uname*] *device*

allocate [–s] [–U *uname*] –g *dev –type*

allocate [–s] [–U *uname*] –F *device*

DESCRIPTION

`allocate` manages the ownership of devices through its allocation mechanism. It ensures that each device is used by only one qualified user at a time.

The *device* argument specifies the device to be manipulated. To preserve the integrity of the device's owner, the `allocate` operation is executed on all the device-special files associated with that device.

The argument *dev–type*, is the device type to be operated on. The argument *dev–type*, can only be used with the –g option.

The default allocate operation, allocates the device-special files associated with *device* to the UID of the current process.

If the –F option is specified, the device cleaning program is executed when allocation is performed. This cleaning program is found in `/etc/security/lib`. The name of this program is found in the `device_allocate`(4) entry for the device in the *dev–exec* field.

OPTIONS

–g *dev–type*	Allocate a non–allocated device with a device–type matching *dev–type*.
–s	Silent. Suppresses any diagnostic output.
–F *device*	Reallocate the device allocated to another user. This option is often used with –U to reallocate a specific device to a specific user. This option requires the `sys_devices` privilege to work.
–U *uname*	Use the user ID `uname` instead of the user ID of the current process when performing the allocate operation. This option requires the `sys_devices` privilege to work.

DIAGNOSTICS

`allocate` returns an nonzero exit status in the event of an error.

FILES

`/etc/security/device_allocate`	Mandatory access control file for devices.
`/etc/security/device_maps`	List of physical devices associated with a device name and type.
`/etc/security/dev/*`	Device storage area.
`/etc/security/lib/*`	Directory of device cleaning scripts.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The –`F` and –`U` options require the `sys_devices` privilege to work.

SEE ALSO

Trusted Solaris 7 Reference Manual

`device_allocate`(4), `device_maps`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

arp – Address resolution display and control

SYNOPSIS

arp *hostname*

arp –a

arp –d *hostname*

arp –f *filename*

arp –s *hostname ether_address* [temp] [pub] [trail]

DESCRIPTION

The `arp` program displays and modifies the Internet-to-Ethernet address translation tables used by the address resolution protocol (see `arp`(7P)).

With no flags, the program displays the current ARP entry for *hostname.* The host may be specified by name or by number, using Internet dot notation.

OPTIONS

`–a` Display all the current ARP entries. The definition for the flags in the table are:

- `P` Publish; includes IP address for the machine and the addresses that have explicitly been added by the `–s` option. ARP will respond to ARP requests for this address.
- `S` Static; not learned for the ARP protocol.
- `U` Unresolved; waiting for ARP response.
- `M` Mapping; only used for the multicast entry for '`224.0.0.0`'.

`–d` Delete an entry for the host called *hostname.* To succeed, the process must inheret the `sys_net_config` privilege.

`–f` Read the file named *filename* and set multiple entries in the ARP tables. Entries in the file should be of the form:

hostname ether_address `[temp [pub ] trail ]`

See option `–s` for argument definitions. To succeed, the process must inheret the `sys_net_config` privilege.

`–s` Create an ARP entry for the host called *hostname* with the Ethernet address *ether_address.* The Ethernet address is given as six hexadecimal bytes separated by colons. The entry will be permanent unless the word `temp` is given in the command. If the word `pub` is given, the entry will be published. For instance, this system will respond to ARP requests for *hostname* even though the hostname is not its own. The word `trail` indicates that trailer encapsulations may be sent to this host. `arp –s` can be used for a limited form of proxy ARP when a host on one of the directly attached networks is not physically present on the subnet. Another machine can then be configured to respond to ARP requests using `arp –s`. This is useful in certain SLIP or PPP configurations. To succeed, the process must inheret the `sys_net_config` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To run, options –`d`, –`f`, and –`s` need to inherit the `sys_net_config` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`ifconfig`(1M)

SunOS 5.7 Reference Manual

`arp`(7P), `attributes`(5)

NAME

atohexlabel – Convert a character-coded label to its hexadecimal equivalent

SYNOPSIS

/usr/sbin/atohexlabel [*character_coded_CMW_label*]

/usr/sbin/atohexlabel –c [*character_coded_clearance*]

/usr/sbin/atohexlabel –s [*character_coded_sensitivity_label*]

DESCRIPTION

`atohexlabel` converts a character-coded label of the type specified into its standard, formatted hexadecimal equivalent and writes the result to the standard output file. If no character-coded label is specified, one is read from standard input.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

`–c` Identifies the character-coded label as a clearance.

`–s` Identifies the character-coded label as a sensitivity label.

RETURN VALUES

`atohexlabel( )` returns:

`0` On success.

`–1` On failure, and writes diagnostics to the standard error file.

FILES

`/etc/security/tsol/label_encodings`
The label encodings file contains the classification names, words, constraints, and values for the defined labels of this system.

NOTES

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as `ADMIN_LOW`.

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]`; however, the IL component is fixed at `ADMIN_LOW`.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW`.

- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always `ADMIN_LOW`, and cannot be set on any objects.
- Options related to information labels in the `label_encodings`(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

SEE ALSO

Trusted Solaris 7 Reference Manual

`label_encodings`(4)

Trusted Solaris administrator's document set

SunOS 5.7 Reference Manual

`attributes`(5)

DIAGNOSTICS

`label translation unavailable`

The label services are currently unavailable either because the label daemon is not running or because the `label_encodings` file is incorrect or unavailable.

label `is not translatable by this process`

This process is not allowed to translate *label*.

The `sys_trans_label` privilege may be used to override this restriction.

`error in` *label* `at position` *n*

label is not a valid label. An error is noted in position *n* of the string.

NAME

audit – Control the behavior of the audit daemon

SYNOPSIS

audit –n | –s | | | –t

DESCRIPTION

The audit command is the general administrator's interface to maintaining the audit trail. The audit daemon may be notified to read the contents of the audit_control(4) file and re-initialize the current audit directory to the first directory listed in the audit_control file or to open a new audit file in the current audit directory specified in the audit_control file as last read by the audit daemon. The audit daemon may also be signaled to close the audit trail and disable auditing.

OPTIONS

–n Signal audit daemon to close the current audit file and open a new audit file in the current audit directory.

–s Signal audit daemon to read audit control file. The audit daemon stores the information internally.

–t Signal audit daemon to close the current audit trail file, disable auditing and die.

DIAGNOSTICS

The audit command will exit with 0 upon success and a positive integer upon failure.

FILES

/etc/security/audit_user	File containing user information for system audit daemon.
/etc/security/audit_control	File containing information for system audit daemon.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This functionality is active only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment. By default, the machine halts when audit files run out of disk space. The Trusted Solaris environment adds programming interfaces, audit classes, and audit events.

This command should run at ADMIN_HIGH.

SEE ALSO

Trusted Solaris 7 Reference Manual

praudit(1M), audit(2), audit_control(4), audit_user(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

This command does not modify a process' preselection mask. It only affects which audit directories are used for audit data storage and to specify the minimum size free.

NAME

auditconfig – Configure auditing

SYNOPSIS

auditconfig *option…*

DESCRIPTION

`auditconfig` provides a command line interface to get and set kernel audit parameters.

OPTIONS

–`chkconf`
Check the configuration of kernel audit event to class mappings. If the runtime class mask of a kernel audit event does not match the configured class mask, a mismatch is reported.

–`conf`
Configure kernel audit event to class mappings. Runtime class mappings are changed to match those in the audit event to class database file.

–`getfsize`
Return the maximum audit file size in bytes and the current size of the audit file in bytes.

–`setfsize` *size*
Set the maximum size of an audit file to `size` bytes. When the size limit is reached, the audit file is closed and another is started.

–`getcond`
Display the kernel audit condition. The condition displayed is the literal string `auditing` meaning auditing is enabled and turned on (the kernel audit module is constructing and queuing audit records) or `noaudit` meaning auditing is enabled but turned off (the kernel audit module is not constructing and queuing audit records), or `disabled` meaning that the audit module has not been enabled. See `auditon`(2) and `auditd`(1M) for further information.

–`setcond`[`auditing|noaudit`]
Set the kernel audit condition to the *condition* specified where *condition* is the literal string `auditing` indicating auditing should be enabled or `noaudit` indicating auditing should be disabled.

–`getclass` *event*
Display the preselection mask associated with the specified kernel audit event. *event* is the kernel event number or event name.

–`setclass` *event audit_flag*[,*audit_flag* . . .]
Map the kernel event *event* to the classes specified by *audit_flags. event* is an event number or name. An *audit_flag* is a two-character string representing an audit class. See `audit_control`(4) for further information.

–`lsevent`

Display the currently configured (runtime) kernel and user level audit event information.

–`getpinfo` *pid*
Display the audit ID, preselection mask, terminal ID and audit session ID for the specified process.

–`setpmask` *pid flags*
Set the preselection mask of the specified process. `flags` is the text representation of the flags similar to that in `audit_control`(4).

–`setsmask` *asid flags*
Set the preselection mask of all processes with the specified audit session ID.

–`setumask` *auid flags*
Set the preselection mask of all processes with the specified audit ID.

–`lspolicy`
Display the kernel audit policies with a description of each policy.

–`getpolicy`
Display the kernel audit policy.

–`setpolicy`[+|-]*policy_flag*[,*policy_flag* ...]
Set the kernel audit policy. A policy *policy_flag* is literal strings that denotes an audit policy. A prefix of + adds the policies specified to the current audit policies. A prefix of - removes the policies specified from the current audit policies. The following are the valid policy flag strings (`auditconfig` –`lspolicy` also lists the current valid audit policy flag strings):

`acl`	Include in the audit data an ACL attribute for each object accessed. Note that regardless of policy, if there is no ACL associated with an object, an attribute will not be generated. This information is not included by default.
`ahlt`	Halt the machine if an asynchronous audit event occurs that cannot be delivered because the audit queue has reached the high-water mark or because there are insufficient resources to construct an audit record. By default, records are dropped and a count is kept of the number of dropped records.
`arge`	Include the `execv`(2) system call environment arguments to the audit record. This information is not included by default.

`argv`	Include the `execv`(2) system call parameter arguments to the audit record. This information is not included by default.
`cnt`	Do not suspend processes when audit resources are exhausted. Instead, drop audit records and keep a count of the number of records dropped. By default, process are suspended until audit resources become available.
`group`	Include the supplementary group token in audit records. By default, the group token is not included.
`slabel`	Include slabels in audit records. This information is included by default.
`passwd`	Include as part of the audit record any bad authentication data encountered during a login operation. The default action is not to include the password in the audit record.
`path`	Add secondary path tokens to audit record. These are typically the pathnames of dynamically linked shared libraries or command interpreters for shell scripts. By default, they are not included.
`trail`	Include the trailer token in every audit record. By default, the trailer token is not included.
`seq`	Include the sequence token as part of every audit record. By default, the sequence token is not included. The sequence token attaches a sequence number to every audit record.
`windata_down`	Include in an audit record any downgraded data moved between windows. By default, this information is not included.
`windata_up`	Include in an audit record any upgraded data moved between windows. By default, this information is not included.

EXAMPLES

EXAMPLE 1 A sample auditconfig program

```
#
# map kernel audit event number 10 to the "fr" audit class
#
% auditconfig –setclass 10 fr

#
# turn on inclusion of exec arguments in exec audit records
#
% auditconfig –setpolicy +argv
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

`/etc/security/audit_event`	Audit event definition and class mappings.
`/etc/security/audit_class`	Audit class definitions.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This functionality is active only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment. By default, the machine halts when audit files run out of disk space. The Trusted Solaris environment adds programming interfaces, audit classes, and audit events.

These policy flags have been added to the Trusted Solaris auditing module: `acl`, `ahlt`, `slabel`, `passwd`, `windata_down`, and `windata_up`.

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as ADMIN_LOW.

Objects still have CMW labels, and CMW labels still include the IL component: IL[SL]; however, the IL component is fixed at ADMIN_LOW.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return ADMIN_LOW.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always ADMIN_LOW, and cannot be set on any objects.
- In auditing, the ilabel token is recorded as ADMIN_LOW, when it is recorded. The audit event numbers 519 (AUE_OFLOAT), 520 (AUE_SFLOAT), and 9036 (AUE_iil_change) continue to be reserved, but those events are no longer recorded.

SEE ALSO

Trusted Solaris 7 Reference Manual

auditd(1M), praudit(1M), auditon(2), execv(2), audit_class(4), audit_control(4), audit_event(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

attributes(5)

NAME

auditd – Audit daemon

SYNOPSIS

/usr/sbin/auditd

DESCRIPTION

The audit daemon controls the generation and location of audit trail files. If auditing is desired, `auditd` reads the `audit_control`(4) file to get a list of directories into which audit files can be written and the percentage limit for how much space to reserve on each filesystem before changing to the next directory.

If `auditd` receives the signal `SIGUSR1`, the current audit file is closed and another is opened. If `SIGHUP` is received, the current audit trail is closed, the `audit_control` file reread, and a new trail is opened. If `SIGTERM` is received, the audit trail is closed and auditing is terminated. The program `audit`(1M) sends these signals and is recommended for this purpose.

Each time the audit daemon opens a new audit trail file, it updates the file `audit_data`(4) to include the correct name.

Auditing Conditions

The audit daemon invokes the program `audit_warn`(1M) under the following conditions with the indicated options:

`audit_warn soft` *pathname*
: The file system upon which *pathname* resides has exceeded the minimum free space limit defined in `audit_control`(4). A new audit trail has been opened on another file system.

`audit_warn allsoft`
: All available file systems have been filled beyond the minimum free space limit. A new audit trail has been opened anyway.

`audit_warn hard` *pathname*
: The file system upon which *pathname* resides has filled or for some reason become unavailable. A new audit trail has been opened on another file system.

`audit_warn allhard` *count*
: All available file systems have been filled or for some reason become unavailable. The audit daemon will repeat this call to `audit_warn` every twenty seconds until space becomes available. *count* is the number of times that `audit_warn` has been called since the problem arose.

`audit_warn ebusy`
: There is already an audit daemon running.

`audit_warn tmpfile`
: The file `/etc/security/audit/audit_tmp` exists, indicating a fatal error.

`audit_warn nostart`

The internal system audit condition is AUC_FCHDONE. Auditing cannot be started without rebooting the system.

`audit_warn auditoff`
The internal system audit condition has been changed to not be AUC_AUDITING by someone other than the audit daemon. This causes the audit daemon to exit.

`audit_warn postsigterm`
An error occurred during the orderly shutdown of the auditing system.

`audit_warn getacdir`
There is a problem getting the directory list from /etc/security/audit/audit_control.

The audit daemon will hang in a sleep loop until this file is fixed.

FILES

`/etc/security/audit/audit_control`	File containing information for system audit daemon.
`/etc/security/audit/audit_data`	File containing current information on audit daemon.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This functionality is active only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment. By default, the machine halts when audit files run out of disk space. The Trusted Solaris environment adds programming interfaces, audit classes, and audit events.

auditd reads the audit_control(4) file rather than the audit_control(4) file and updates the audit_data(4) file rather than the audit_data(4) file.

SEE ALSO

Trusted Solaris 7 Reference Manual

audit(1M), audit_warn(1M), praudit(1M), auditon(2), auditsvc(2), audit.log(4), audit_control(4), audit_data(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

attributes(5)

NAME

auditreduce – Merge and select audit records from audit trail files

SYNOPSIS

auditreduce [*options*] [*audit-trail-file*…]

DESCRIPTION

`auditreduce` allows you to select or merge records from audit trail files. Audit files may be from one or more machines.

The merge function merges together audit records from one or more input audit trail files into a single output file. The records in an audit trail file are assumed to be sorted in chronological order (oldest first) and this order is maintained by `auditreduce` in the output file.

Unless instructed otherwise, `auditreduce` will merge the entire audit trail, which consists of all the audit trail files in the directory structure *audit_root_dir*/*/files (see `audit_control`(4) for details of the structure of the audit root). Unless stated with the –R or –S option, *audit_root_dir* defaults to `/etc/security/audit`. By using the file selection options it is possible to select some subset of these files, or files from another directory, or files named explicitly on the command line.

The select function allows audit records to be selected on the basis of numerous criteria relating to the record's content (see `audit.log`(4) for details of record content). A record must meet all of the *record-selection-option* criteria to be selected.

Audit Trail Filename Format

Any audit trail file not named on the command line must conform to the audit trail filename format. Files produced by the audit system already have this format. Output file names produced by `auditreduce` are in this format. It is:

start-time`.` *end-time*`.` *suffix*

where *start-time* is the 14-character timestamp of when the file was opened, *end-time* is the 14-character timestamp of when the file was closed, and *suffix* is the name of the machine which generated the audit trail file, or some other meaningful suffix (e.g., `all`, if the file contains a combined group of records from many machines). The *end-time* may be the literal string `not_terminated`, to indicate that the file is still being written to by the audit system. Timestamps are of the form *yyyymmddhhmmss* (year, month, day, hour, minute, second). The timestamps are in Greenwich Mean Time (GMT).

OPTIONS

File Selection Options

The file selection options indicate which files are to be processed and certain types of special treatment.

–A

All of the records from the input files will be selected regardless of their timestamp. This option effectively disables the –a, –b, and –d options. This is useful in preventing the loss of records if the –D option is used to delete

the input files after they are processed. Note, however, that if a record is *not* selected due to another option, then –A will not override that.

–C
Only process complete files. Files whose filename *end-time* timestamp is `not_terminated` are not processed (such a file is currently being written to by the audit system). This is useful in preventing the loss of records if –D is used to delete the input files after they are processed. It does not apply to files specified on the command line.

–D *suffix*
Delete input files after they are processed. The files are only deleted if the entire run is successful. If `auditreduce` detects an error while reading a file, then that file is not deleted. If –D is specified, –A, –C and –O are also implied. *suffix* is given to the –O option. This helps prevent the loss of audit records by ensuring that all of the records are written, only complete files are processed, and the records are written to a file before being deleted. Note that if both –D and –O are specified in the command line, the order of specification is significant. The *suffix* associated with the latter specification is in effect.

–M *machine*
Allows selection of records from files with *machine* as the filename suffix. If –M is not specified, all files are processed regardless of suffix. –M can also be used to allow selection of records from files that contain combined records from many machines and have a common suffix (such as `all`).

–O *suffix*
Direct output stream to a file in the current *audit_root_dir* with the indicated suffix. *suffix* may alternatively contain a full pathname, in which case the last component is taken as the suffix, ahead of which the timestamps will be placed, ahead of which the remainder of the pathname will be placed. If the –O option is not specified, the output is sent to the standard output. When `auditreduce` places timestamps in the filename, it uses the times of the first and last records in the merge as the *start-time* and *end-time*.

–Q
Quiet. Suppress notification about errors with input files.

–R *pathname*
Specify the pathname of an alternate audit root directory *audit_root_dir* to be *pathname*. Therefore, rather than using `/etc/security/audit`/*/files by default, *pathname*/*/files will be examined instead.

–S *server*
This option causes `auditreduce` to read audit trail files from a specific location (server directory). *server* is normally interpreted as the name of

a subdirectory of the audit root, therefore `auditreduce` will look in *audit_root_dir*/*server*/files for the audit trail files. But if *server* contains any '/' characters, it is the name of a specific directory not necessarily contained in the audit root. In this case, *server*/files will be consulted. This option allows archived files to be manipulated easily, without requiring that they be physically located in a directory structure like that of `/etc/security/audit`.

–v
Verbose. Display the name of each file as it is opened, and how many records total were written to the output stream.

Record Selection Options

The record selection options listed below are used to indicate which records are written to the output file produced by `auditreduce`.

Multiple arguments of the same type are not permitted.

–a *date-time*
Select records that occurred at or after *date-time*. The *date-time* argument is described under `Option Arguments`, below. *date-time* is in local time. The –a and –b options can be used together to form a range.

–b *date-time*
Select records that occurred before *date-time*.

–c *audit-classes*
Select records by audit class. Records with events that are mapped to the audit classes specified by *audit-classes* are selected. Audit class names are defined in `audit_class`(4). The *audit-classes* can be a comma separated list of `audit` *flags* like those described in `audit_control`(4). Using the `audit` *flags*, one can select records based upon success and failure criteria.

–d *date-time*
Select records that occurred on a specific day (a 24-hour period beginning at 00:00:00 of the day specified and ending at 23:59:59). The day specified is in local time. The time portion of the argument, if supplied, is ignored. Any records with timestamps during that day are selected. If any hours, minutes, or seconds are given in *time*, they are ignored. –d can not be used with –a or –b.

–e *effective-user*
Select records with the specified *effective-user*.

–f *effective-group*
Select records with the specified *effective-group*.

–g *real-group*
Select records with the specified *real-group*.

–j *subject-ID*
Select records with the specified *subject-ID* where *subject-ID* is a process ID.

–m *event*
Select records with the indicated *event*. The *event* is the literal string or the *event* number.

–o *object_type=objectID_value*
Select records by object type. A match occurs when the record contains the information describing the specified *object_type* and the object ID equals the value specified by *objectID_value*. The allowable object types and values are as follows:

`file`=*pathname* Select records containing file system objects with the specified *pathname*, where *pathname* is a comma separated list of regular expressions. If a regular expression is preceeded by a tilda (~), files matching the expression are excluded from the output. For example, the option `file="~/usr/openwin,/usr,/etc"` would select all files in `/usr` or `/etc` except those in `/usr/openwin`. The order of the regular expressions is important because `auditreduce` processes them from left to right, and stops when a file is known to be either selected or excluded. Thus the option file= `/usr`, `/etc`, `~/usr/openwin` would select all files in `/usr` and all files in `/etc`. Files in `/usr/openwin` are not excluded because the regular expression `/usr` is matched first. Care should be given in surrounding the *pathname* with quotes so as to prevent the shell from expanding any tildas.

`msgqid`=*ID* Select records containing message queue objects with the specified *ID* where *ID* is a message queue ID.

`pid`=*ID* Select records containing process objects with the specified *ID* where *ID* is a process ID. Note: Process are objects when they are receivers of signals.

`semid`=*ID* Select records containing semaphore objects with the specified *ID* where *ID* is a semaphore ID.

`shmid`=*ID* Select records containing shared memory objects with the specified *ID* where *ID* is a shared memory ID.

`sock`=*port_number* | *machine*

Select records containing socket objects with the specified *port_number* or the specified *machine* where *machine* is a machine name as defined in `hosts`(4).

–r *real-user*
Select records with the specified *real-user*.

–s *sensitivity-label*
Select records with the specified *sensitivity-label*, which may be a range as explained under Option Arguments, *sensitivity-label*.

–u *audit-user*
Select records with the specified *audit-user*. When one or more *filename* arguments appear on the command line, only the named files are processed. Files specified in this way need not conform to the audit trail filename format. However, –M, –S, and –R may not be used when processing named files. If the *filename* is "–" then the input is taken from the standard input.

Option Arguments

audit-trail-file
An audit trail file as defined in `audit.log`(4). An audit trail file not named on the command line must conform to the audit trail file name format. Audit trail files produced as output of `auditreduce` are in this format as well. The format is:

```
start-time . end-time . suffix
```

start-time is the 14 character time stamp denoting when the file was opened. *end-time* is the 14 character time stamp denoting when the file was closed. *end-time* may also be the literal string `not_terminated`, indicating the file is still be written to by the audit daemon or the file was not closed properly (a system crash or abrupt halt occurred). *suffix* is the name of the machine that generated the audit trail file (or some other meaningful suffix; e.g. `all` would be a good suffix if the audit trail file contains a combined group of records from many machines).

date-time
The *date-time* argument to –a, –b, and –d can be of two forms: An absolute *date-time* takes the form:

yyyymmdd [*hh* [*mm* [*ss*]]]

where *yyyy* specifies a year (with 1970 as the earliest value), *mm* is the month (01-12), dd is the day (01-31), *hh* is the hour (00-23), *mm* is the minute (00-59), and *ss* is the second (00-59). The default is 00 for *hh*, *mm* and *ss*.

An offset can be specified as: +*n* d|h|m| s where *n* is a number of units, and the tags d, h, m, and s stand for days, hours, minutes and seconds, respectively. An offset is relative to the starting time. Thus, this form can only be used with the –b option.

event
The literal string or ordinal event number as found in audit_event(4). If *event* is not found in the audit_event file it is considered invalid.

group
The literal string or ordinal group ID number as found in group(4). If *group* is not found in the group file it is considered invalid. *group* may be negative.

pathname
A regular expression describing a pathname.

sensitivity-label
The literal string representation of an sensitivity label or a range of two valid sensitivity labels. To specify a range, use [*x*]:[*y*] where *x* and *y* are valid sensitivity labels. Only those records that are fully bounded by *x* and *y* will be selected. If *x* or *y* is omitted, the default uses ADMIN_LOW or ADMIN_HIGH respectively.

user
The literal username or ordinal user ID number as found in passwd(4). If the username is not found in the passwd file it is considered invalid. *user* may be negative.

EXAMPLES

EXAMPLE 1 The auditreduce command.

praudit(1M) is available to display audit records in a human-readable form.

This will display the entire audit trail in a human-readable form:

```
% auditreduce | praudit
```

If all the audit trail files are being combined into one large file, then deleting the original files could be desirable to prevent the records from appearing twice:

```
% auditreduce –V –d /etc/security/audit/combined/all
```

This will print what user milner did on April 13, 1988. The output will be displayed in a human-readable form to the standard output:

```
% auditreduce –d 19880413 –u milner | praudit
```

The above example may produce a large volume of data if `milner` has been busy. Perhaps looking at only login and logout times would be simpler. The `–c` option will select records from a specified class:

```
% auditreduce –d 19880413 –u milner –c lo | praudit
```

To see `milner`'s login/logout activity for April 13, 14, and 15 the following is used. The results are saved to a file in the current working directory. Note that the name of the output file will have `milnerlo` as the *suffix*, with the appropriate timestamp prefixes. Note that the long form of the name is used for the `–c` option:

```
% auditreduce –a 19880413 –b +3d –u milner –c login_logout –o milnerlo
```

To follow `milner`'s movement about the file system on April 13, 14, and 15 the `chdir` record types could be viewed. Note that in order to get the same time range as the above example we needed to specify the `–b` time as the day after our range. This is because `19880416` defaults to midnight of that day, and records before that fall on `0415`, the end-day of the range.

```
% auditreduce –a 19880413 –b 19880416 –u milner –m AUE_CHDIR | praudit
```

In this example the audit records are being collected in summary form (the login/logout records only). The records are being written to a summary file in a different directory than the normal audit root to prevent the selected records from existing twice in the audit root.

```
% auditreduce –d 19880330 –c lo –o /etc/security/audit_summary/logins
```

If activity for user ID 9944 has been observed, but that user is not known to the system administrator, then the following example will search the entire audit trail for any records generated by that user. `auditreduce` will query the system as to the current validity of ID 9944, and print a warning message if it is not currently active:

```
% auditreduce –o /etc/security/audit_suspect/user9944 –u 9944
```

FILES

`/etc/security/audit/`*`server`*`/files/*`
Location of audit trails, when stored.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The functionality described in this man page is available only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment.

The Trusted Solaris environment has added the **-s** *sensitivity-label* record selection option to this command.

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as `ADMIN_LOW`.

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]`; however, the IL component is fixed at `ADMIN_LOW`.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW`.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always `ADMIN_LOW`, and cannot be set on any objects.
- In auditing, the `ilabel` token is recorded as `ADMIN_LOW`, when it is recorded. The audit event numbers 519 (AUE_OFLOAT), 520 (AUE_SFLOAT), and 9036 (AUE_iil_change) continue to be reserved, but those events are no longer recorded.

SEE ALSO

Trusted Solaris 7 Reference Manual

`praudit`(1M), `audit.log`(4), `audit_class`(4), `audit_control`(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`group`(4), `hosts`(4), `passwd`(4), `attributes`(5)

DIAGNOSTICS

`auditreduce` will print out error messages if there are command line errors and then exit. If there are fatal errors during the run `auditreduce` will print an explanatory message and exit. In this case the output file may be in an inconsistent state (no trailer or partially written record) and `auditreduce` will print a warning message before exiting. Successful invocation returns `0` and unsuccessful invocation returns `1`.

Since `auditreduce` may be processing a large number of input files, it is possible that the machine-wide limit on open files will be exceeded. If this happens, `auditreduce` will print a message to that effect, give information on how many file there are, and exit.

If `auditreduce` prints a record's timestamp in a diagnostic message, that time is in local time. However, when filenames are displayed, their timestamps are in GMT.

BUGS

Conjunction, disjunction, negation, and grouping of record selection options should be allowed.

NAME

audit_startup – Audit subsystem initialization script

SYNOPSIS

/etc/security/audit_startup

DESCRIPTION

The `audit_startup` script is used to initialize the audit subsystem before the audit deamon is started. This script is configurable by the security administrator, and currently consists of a series of `auditconfig`(1M) commands to set the system default policy, and download the initial event to class mapping.

SUMMARY OF TRUSTED SOLARIS CHANGES

By default, the audit module is enabled in the Trusted Solaris environment. By default, the machine halts when audit files run out of disk space. The Trusted Solaris environment adds programming interfaces, audit classes, and audit events.

SEE ALSO

Trusted Solaris 7 Reference Manual

`auditconfig`(1M), `auditd`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

auditstat – Display kernel audit statistics

SYNOPSIS

auditstat [–c *count*] [–h *numlines*] [–i *interval*] [–n] [–v]

DESCRIPTION

`auditstat` displays kernel audit statistics. To succeed, it must inherit the `sys_audit` privilege. The fields displayed are as follows:

`aud` The total number of audit records processed by the `audit`(2) system call.

`ctl` This field is obsolete.

`drop` The total number of audit records that have been dropped. Records are dropped according to the kernel audit policy. See `auditon`(2), `AUDIT_CNT` policy for details.

`enq` The total number of audit records put on the kernel audit queue.

`gen` The total number of audit records that have been constructed (not the number written).

`kern` The total number of audit records produced by user processes (as a result of system calls).

`mem` The total number of Kbytes of memory currently in use by the kernel audit module.

`nona` The total number of non-attributable audit records that have been constructed. These are audit records that are not attributable to any particular user.

`rblk` The total number of times that `auditsvc`(2) has blocked waiting to process audit data.

`tot` The total number of Kbytes of audit data written to the audit trail.

`wblk` The total number of times that user processes blocked on the audit queue at the high water mark.

`wrtn` The total number of audit records written. The difference between `enq` and `wrtn` is the number of outstanding audit records on the audit queue that have not been written.

OPTIONS

–c *count* Display the statistics a total of *count* times. If *count* is equal to zero, statistics are displayed indefinitely. A time interval must be specified.

–h *numlines* Display a header for every *numlines* of statistics printed. The default is to display the header every 20 lines. If *numlines* is equal to zero, the header is never displayed.

–i *interval*	Display the statistics every *interval* where *interval* is the number of seconds to sleep between each collection.
–n	Display the number of kernel audit events currently configured.
–v	Display the version number of the kernel audit module software.

EXIT STATUS

0	Successful completion.
1	An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The functionality described in this man page is available only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment. To succeed, this command must have the sys_audit privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

auditconfig(1M), praudit(1M), audit(2), auditon(2), auditsvc(2)

SunOS 5.7 Reference Manual

attributes(5)

NAME

audit_warn – Audit daemon warning script

SYNOPSIS

/etc/security/audit_warn [*option*[*arguments*]]

DESCRIPTION

The `audit_warn` script processes warning or error messages from the audit daemon. When a problem is encountered, the audit daemon, `auditd`(1M) calls `audit_warn` with the appropriate arguments. The *option* argument specifies the error type.

The system administrator can specify a list of mail recipients to be notified when an audit_warn situation arises by defining a mail alias called `audit_warn` in `aliases`(4). The users that make up the `audit_warn` alias are typically the administrative roles.

OPTIONS

`allhard` *count* — Indicates that the hard limit for all filesystems has been exceeded *count* times. The default action for this option is to send mail to the `audit_warn` alias only if the *count* is `1`, and to write a message to the machine console every time. It is recommended that mail *not* be sent every time as this could result in a the saturation of the file system that contains the mail spool directory.

`allsoft` — Indicates that the soft limit for all filesystems has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`auditoff` — Indicates that someone other than the audit daemon changed the system audit state to something other than `AUC_AUDITING`. The audit daemon will have exited in this case. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`ebusy` — Indicates that the audit daemon is already running. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`getacdir` *count* — Indicates that there is a problem getting the directory list from `audit_control`(4). The audit daemon will hang in a sleep loop until the file is fixed. The default action for this option is to send mail to the `audit_warn` alias only if *count* is `1`, and to write a message to the machine console every time. It is recommended that mail *not* be sent every time as this could result in a the saturation of the file system that contains the mail spool directory.

`hard` *filename* — Indicates that the hard limit for the file has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`nostart` — Indicates that auditing could not be started. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console. Some administrators may prefer to modify `audit_warn` to reboot the system when this error occurs.

`postsigterm` — Indicates that an error occurred during the orderly shutdown of the audit daemon. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`soft` *filename* — Indicates that the soft limit for *filename* has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`tmpfile` — Indicates that the temporary audit file already exists indicating a fatal error. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SUMMARY OF TRUSTED SOLARIS CHANGES

The functionality described in this man page is available only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment. By default, the machine halts when audit files run out of disk space. The Trusted Solaris environment adds programming interfaces, audit classes, and audit events.

SEE ALSO

Trusted Solaris 7 Reference Manual

`audit`(1M), `auditd`(1M), `audit.log`(4), `audit_control`(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`aliases`(4), `attributes`(5)

NAME

automount – Install automatic mount points

SYNOPSIS

/usr/sbin/automount [–t *duration*] [–v]

DESCRIPTION

The automount utility installs autofs mount points and associates an automount map with each mount point. The autofs file system monitors attempts to access directories within it and notifies the automountd(1M) daemon. The daemon uses the map to locate a file system, which it then mounts at the point of reference within the autofs file system. A map can be assigned to an autofs mount using an entry in the /etc/auto_master map or a direct map.

If the file system is not accessed within an appropriate interval (five minutes by default), the automountd daemon unmounts the file system.

The file /etc/auto_master determines the locations of all autofs mount points. By default, this file contains four entries:

```
# Master map for automounter
#
+auto_master
/net          -hosts    -nosuid
/home         auto_home
/xfn          -xfn
```

The +auto_master entry is a reference to an external NIS or NIS+ master map. If one exists, then its entries are read as if they occurred in place of the +auto_master entry. The remaining entries in the master file specify a directory on which an autofs mount will be made followed by the automounter map to be associated with it. Optional mount options may be supplied as an optional third field in the each entry. These options are used for any entries in the map that do not specify mount options explicitly. Security attributes may also follow the automounter map name. These consist of a semicolon separated list of security attributes to be associated with the map. See mount(1M) for a description of these security attributes. As with mount options, security attributes in /etc/auto_master are used for any entries in the map that do not specify security attributes explicitly. The security attribute list must be preceded by a –S flag to distinguish it from mount options. The automount command is usually run without arguments. It compares the entries /etc/auto_master with the current list of autofs mounts in /etc/mnttab and adds, removes or updates autofs mounts to bring the /etc/mnttab up to date with the /etc/auto_master. At boot time it installs all autofs mounts from the master map. Subsequently, it may be run to install autofs mounts for new entries in the master map or the direct map, or to perform unmounts for entries that have been removed from these maps.

OPTIONS

The following options are supported:

–t *duration* Specifies a *duration*, in seconds, that a file system is to remain mounted when not in use. The default is 10 minutes.

–v Verbose mode. Notifies of `autofs` mounts, unmounts, or other non-essential information.

USAGE

Map Entry Format

A simple map entry (mapping) takes the form:

```
key [ -mount-options ] [ –Sattribute-list ] location . . .
```

where `key` is the full pathname of the directory to mount when used in a direct map, or the simple name of a subdirectory in an indirect map. *mount-options* is a comma-separated list of `mount` options, and *location* specifies a file system from which the directory may be mounted. In the case of a simple NFS mount, the options that can be used are as specified in `mount_nfs`(1M), and *location* takes the form:

```
host: pathname
```

host is the name of the host from which to mount the file system, and *pathname* is the absolute pathname of the directory to mount.

Options to other file systems are documented on the other `mount_*` reference manual pages, for example, `mount_cachefs`(1M).

Replicated File Systems

Multiple *location* fields can be specified for replicated NFS file systems, in which case `automount` and the kernel will each try to use that information to increase availability. If the read-only flag is set in the map entry, `automount` mounts a list of locations that the kernel may use, sorted by several criteria. When a server does not respond, the kernel will switch to an alternate server. The sort ordering of `automount` is used to determine how the next server is chosen. If the read-only flag is not set, `automount` will mount the best single location, chosen by the same sort ordering, and new servers will only be chosen when an unmount has been possible, and a remount is done. Servers on the same local subnet are given the strongest preference, and servers on the local net are given the second strongest preference. Among servers equally far away, response times will determine the order if no weighting factors (see below) are used.

If the list includes server locations using both the NFS Version 2 Protocol and the NFS Version 3 Protocol, `automount` will choose only a subset of the server locations on the list, so that all entries will be the same protocol. It will choose servers with the NFS Version 3 Protocol so long as an NFS Version 2 Protocol server on a local subnet will not be ignored. See the *NFS Administration Guide* for additional details.

If each *location* in the list shares the same *pathname* then a single *location* may be used with a comma-separated list of hostnames:

```
hostname,hostname . . . : pathname
```

Requests for a server may be weighted, with the weighting factor appended to the server name as an integer in parentheses. Servers without a weighting are assumed to have a value of zero (most likely to be selected). Progressively higher values decrease the chance of being selected. In the example,

```
man -ro alpha,bravo,charlie(1),delta(4) : /usr/man
```

hosts `alpha` and `bravo` have the highest priority; host `delta` has the lowest.

Server proximity takes priority in the selection process. In the example above, if the server `delta` is on the same network segment as the client, but the others are on different network segments, then `delta` will be selected; the weighting value is ignored. The weighting has effect only when selecting between servers with the same network proximity.

In cases where each server has a different export point, the weighting can still be applied. For example:

```
man -ro alpha : /usr/man   bravo,charlie(1) : /usr/share/man \
delta(3) : /export/man
```

A mapping can be continued across input lines by escaping the NEWLINE with a backslash (\). Comments begin with a number sign (#) and end at the subsequent NEWLINE.

Map Key Substitution

The ampersand (&) character is expanded to the value of the `key` field for the entry in which it occurs. In this case:

```
jane sparcserver : /home/&
```

the & expands to `jane`.

Wildcard Key

The asterisk (*) character, when supplied as the `key` field, is recognized as the catch-all entry. Such an entry will match any key not previously matched. For instance, if the following entry appeared in the indirect map for `/config`:

```
*          & : /export/config/&
```

this would allow automatic mounts in `/config` of any remote file system whose location could be specified as:

```
hostname : /export/config/hostname
```

Variable Substitution

Client-specific variables can be used within an `automount` map. For instance, if `$HOST` appeared within a map, `automount` would expand it to its current value for the client's host name. Supported variables are:

`ARCH`	The application architecture is derived from the output of `uname –m`	The architecture name. For example, "sun4" on a sun4u machine.
`CPU`	The output of `uname –p`	The processor type. For example, "sparc"
`HOST`	The output of `uname –n`	The host name. For example, "biggles"
`OSNAME`	The output of `uname –s`	The OS name. For example, "SunOS"
`OSREL`	The output of `uname –r`	The OS release name. For example "5.7"
`OSVERS`	The output of `uname –v`	The OS version. For example, "beta1.0"
`NATISA`	The output of `isainfo –n`	The native instruction set architecture for the system. For example, "sparcv9"

If a reference needs to be protected from affixed characters, you can surround the variable name with curly braces (`{ }`).

Multiple Mounts

A multiple mount entry takes the form:

```
key [-mount-options] [ [mountpoint] [-mount-options] location. . . ] . . .
```

The initial `/`[*mountpoint*] is optional for the first mount and mandatory for all subsequent mounts. The optional *mountpoint* is taken as a pathname relative to the directory named by `key`. If *mountpoint* is omitted in the first occurrence, a *mountpoint* of `/` (root) is implied.

Given an entry in the indirect map for `/src`

```
BETA -RO \
/SVR1,SVR2 : /EXPORT/SRC/BETA\
/1.0SVR1,SVR2 : /EXPORT/SRC/BETA/1.0\
/1.0/MANSVR1,SVR2 : /EXPORT/SRC/BETA/1.0/MAN
```

All offsets must exist on the server under `beta`. `automount` will automatically mount `/src/beta`, `/src/beta/1.0`, and `/src/beta/1.0/man`, as needed, from either `svr1` or `svr2`, whichever host is nearest and responds first.

Other File System Types

The automounter assumes NFS mounts as a default file system type. Other file system types can be described using the `fstype` mount option. Other mount options specific to this file system type can be combined with the `fstype` option. The location field must contain information specific to the file system type. If the location field begins with a slash, a colon character must be prepended, for instance, to mount a CD file system:

```
cdrom -fstype=hsfs,ro   :/dev/sr0
```

or to perform an `autofs` mount:

```
src   –fstype=autofs    auto_src
```

Note: Use this procedure only if you are not using Volume Manager.

Mounts using CacheFS are most useful when applied to an entire map as map defaults. The following entry in the master map describes cached home directory mounts. It assumes the default location of the cache directory, `/cache`.

```
/home auto_home       –fstype=cachefs,backfstype=nfs
```

See the `NOTES` section for information on option inheritance.

Indirect Maps

An indirect map allows you to specify mappings for the subdirectories you wish to mount under the `directory` indicated on the command line. In an indirect map, each `key` consists of a simple name that refers to one or more file systems that are to be mounted as needed.

Direct Maps

Entries in a direct map are associated directly with `autofs` mount points. Each `key` is the full pathname of an `autofs` mount point. The direct map as a whole is not associated with any single directory.

Included Maps

The contents of another map can be included within a map with an entry of the form

```
+mapname
```

If *mapname* begins with a slash, it is assumed to be the pathname of a local file. Otherwise, the location of the map is determined by the policy of the name service switch according to the entry for the automounter in `/etc/nsswitch.conf`, such as

```
automount: files nis
```

If the name service is `files`, then the name is assumed to be that of a local file in `/etc`. If the key being searched for is not found in the included map, the search continues with the next entry.

Special Maps

There are three special maps available: `–hosts`, `–xfn`, and `–null`. The `–hosts` map is used with the `/net` directory and assumes that the map key is the hostname of an NFS server. The `automountd` daemon dynamically constructs a map entry from the server's list of exported file systems. For instance, a reference to `/net/hermes/usr` would initiate an automatic mount of all exported file systems from `hermes` that are mountable by the client. References to a directory under `/net/hermes` will refer to the corresponding directory relative to `hermes` root.

The `–xfn` map is used to mount the initial context of the Federated Naming Service (FNS) namespace under the `/xfn` directory. For more information on FNS, see `fns`(5), `fns_initial_context`(5), `fns_policies`(5), and the Federated Naming Service Guide.

The `–null` map, when indicated on the command line, cancels a previous map for the directory indicated. This is most useful in the `/etc/auto_master` for cancelling entries that would otherwise be inherited from the `+auto_master` include entry. To be effective, the `–null` entries must be inserted before the included map entry.

Executable Maps

Local maps that have the execute bit set in their file permissions will be executed by the automounter and provided with a key to be looked up as an argument. The executable map is expected to return the content of an automounter map entry on its stdout or no output if the entry cannot be determined. A direct map cannot be made executable.

Configuration and the auto_master Map

When initiated without arguments, `automount` consults the master map for a list of `autofs` mount points and their maps. It mounts any `autofs` mounts that are not already mounted, and unmounts `autofs` mounts that have been removed from the master map or direct map.

The master map is assumed to be called `auto_master` and its location is determined by the name service switch policy. Normally the master map is located initially as a local file `/etc/auto_master`.

Browsing

The Solaris 7 release supports browsability of indirect maps. This allows all of the potential mount points to be visible, whether or not they are mounted. The –nobrowse option can be added to any indirect autofs map to disable browsing. For example:

```
/net     -hosts      -nosuid,nobrowse
/home    auto_home
```

In this case, any *hostnames* would only be visible in /net after they are mounted, but all potential mount points would be visible under /home. The –browse option enables browsability of autofs file systems. This is the default for all indirect maps.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 An error occurred.

FILES

/etc/auto_master	master automount map.
/etc/auto_home	map to support automounted home directories.
/etc/nsswitch.conf	the name service switch configuration file.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

Security attributes can be specified in auto_master and in autofs map entries with the –S option. If security attributes are not specified in either auto_master or an autofs map entry, but an entry for the mount point is in /etc/security/tsol/vfstab_adjunct, then security attributes in the vfstab_adjunct file are used.

automount must be started as root, with a process sensitivity label of ADMIN_LOW, and a clearance of ADMIN_HIGH. It must have the PAF_TRUSTED_PATH process attribute, and must inherit the following privileges: file_mac_read, file_mac_write, file_dac_read, file_dac_write, and sys_mount.

SEE ALSO
Trusted Solaris 7 Reference Manual

automountd(1M), mount(1M), vfstab_adjunct(4)

SunOS 5.7 Reference Manual

`attributes`(5), `fns`(5), `fns_initial_context`(5), `fns_policies`(5)

NFS Administration Guide

NOTES

`autofs` mount points must not be hierarchically related. `automount` does not allow an `autofs` mount point to be created within another `autofs` mount.

Since each direct map entry results in a new `autofs` mount such maps should be kept short.

Entries in both direct and indirect maps can be modified at any time. The new information is used when `automountd` next uses the map entry to do a mount.

New entries added to a master map or direct map will not be useful until the automount command is run to install them as new `autofs` mount points. New entries added to an indirect map may be used immediately.

As of the Solaris 7 release, a listing (see `ls`(1)) of the `autofs` directory associated with an indirect map shows all potential mountable entries. The attributes associated with the potential mountable entries are temporary. The real file system attributes will only be shown once the file system has been mounted.

Default mount options can be assigned to an entire map when specified as an optional third field in the master map. These options apply only to map entries that have no mount options. Note that map entities with options override the default options, as at this time, the options do not concatenate. The concatenation feature is planned for a future release.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same.

NAME

automountd – Autofs mount/unmount daemon

SYNOPSIS

automountd [–Tvn] [–D *name=value*]

DESCRIPTION

automountd is an RPC server that answers filesystem mount and unmount requests from the autofs filesystem. It uses local files or name service maps to locate filesystems to be mounted. These maps are described with the automount(1M) command.

The automountd daemon is automatically invoked in run level 2.

OPTIONS

–T Trace. Expand each RPC call and display it on the standard output.

–v Verbose. Log status messages to the console.

–n Turn off browsing for all autofs mount points. This option overrides the –browse autofs map option on the local host.

–D *name=value* Assign *value* to the indicated automount map substitution variable. These assignments cannot be used to substitute variables in the master map auto_master.

USAGE

See largefile(5) for the description of the behavior of automountd when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/auto_master Master map for automounter

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

automountd must be started as root, with a process sensitivity label of ADMIN_LOW, and a clearance of ADMIN_HIGH. It must have the PAF_TRUSTED_PATH process attribute, and must inherit the following privileges: file_mac_read, file_mac_write, file_upgrade_sl, file_upgrade_il, net_mac_read, net_privaddr, net_upgrade_sl, net_upgrade_il, proc_audit_tcb, proc_setsl, proc_setil, sys_mount, and sys_trans_label.

SEE ALSO

Trusted Solaris 7 Reference Manual

automount(1M)

SunOS 5.7 Reference Manual

attributes(5)

NAME

autopush – Configures lists of automatically pushed STREAMS modules

SYNOPSIS

autopush –f *filename*

autopush –g –M *major* –m *minor*

autopush –r –M *major* –m *minor*

DESCRIPTION

The autopush command configures the list of modules to be automatically pushed onto the stream when a device is opened. It can also be used to remove a previous setting or get information on a setting.

OPTIONS

The following options are supported:

–f *filename* Sets up the autopush configuration for each driver according to the information stored in *filename.* An autopush file consists of lines of four or more fields, separated by spaces as shown below:

major minor last-minor module1 module2 … modulen

The first field is a string that specifies the *major* device name, as listed in the /kernel/drv directory. The next two fields are integers that specify the *minor* device number and *last-minor* device number. The fields following represent the names of modules. If *minor* is –1, then all minor devices of a major driver specified by *major* are configured, and the value for *last-minor* is ignored. If *last-minor* is 0, then only a single minor device is configured. To configure a range of minor devices for a particular major, *minor* must be less than *last-minor*.

The last fields of a line in the autopush file represent the list of module names. The maximum number of modules that can be automatically pushed on a stream is eight. The modules are pushed in the order they are specified. Comment lines start with a # sign.The sys_devices privilege is required for this command to succeed.

–g Gets the current configuration setting of a particular *major* and *minor* device number specified with the –M and –m options respectively and displays the autopush modules associated with it. It will also return the starting minor device number if the request corresponds to a setting of a range (as described with the –f option).

–M *major* Specifies the major device number.

–m *minor* Specifies the minor device number.

`–r` Removes the previous configuration setting of the particular *major* and *minor* device number specified with the `–M` and `–m` options respectively. If the values of *major* and *minor* correspond to a previously established setting of a range of minor devices, where *minor* matches the first minor device number in the range, the configuration would be removed for the entire range.

EXAMPLES

EXAMPLE 1 Using the `autopush`command.

The following example gets the current configuration settings for the *major* and *minor* device numbers as indicated and displays the `autopush` modules associated with them for the character-special device `/dev/term/a:`

```
example# autopush –g –M 29 –m 0
Major Minor Lastminor Modules
29 0 1 ldterm ttcompat
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The `sys_devices` privilege is required for this command to succeed.

FILES

`/etc/iu.ap` Autopush configuration file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`sad`(7D)

SunOS 5.7 Reference Manual

`bdconfig`(1M), `ttymon`(1M), `attributes`(5), `streamio`(7I), `ldterm`(7M), `ttcompat`(7M)

STREAMS Programming Guide

NAME

rpc.bootparamd, bootparamd – Boot parameter server

SYNOPSIS

/usr/sbin/rpc.bootparamd [–d]

DESCRIPTION

`rpc.bootparamd` is a server process that provides information from a `bootparams` database to diskless clients at boot time. See `bootparams`(4)

The source for the `bootparams` database is determined by the `nsswitch.conf`(4) file (on the machine running the `rpc.bootparamd` process).

The `rpc.bootparamd` program can be invoked either by `inetd`(1M) or directly from the command line.

OPTIONS

–d Display debugging information.

SUMMARY OF TRUSTED SOLARIS CHANGES

`rpc.bootparamd` requires the trust path attribute with a UID of `0` , and the sensitivity label `ADMIN_LOW` .

FILES

`/etc/bootparams` Boot parameter database.

`/etc/nsswitch.conf` Configuration file for the name-service switch.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`bootparams`(4) , `attributes`(5)

NOTES

A diskless client requires service from at least one `rpc.bootparamd` process running on a server that is on the same IP subnetwork as the diskless client.

Some routines that compare hostnames use case-sensitive string comparisons; some do not. If an incoming request fails, verify that the case of the hostname in the file to be parsed matches the case of the hostname called for, and attempt the request again.

NAME

bsmconv, bsmunconv – Enable or disable the Basic Security Module (BSM)

SYNOPSIS

/etc/security/bsmconv [*rootdir*…]

/etc/security/bsmunconv [*rootdir*…]

DESCRIPTION

In the Solaris environment, the `bsmconv` and `bsmunconv` scripts are used to enable or disable the BSM features, auditing and device protection.

In the Trusted Solaris environment, the `bsmconv` and `bsmunconv` scripts are *not* used to enable or disable auditing, or to protect devices.

SUMMARY OF TRUSTED SOLARIS CHANGES

Enabling and disabling auditing in the Trusted Solaris environment does *not* use the `bsmconv` and `bsmunconv` commands. See *Trusted Solaris Audit Administration* for the procedure to disable and enable auditing. Devices are always protected in the Trusted Solaris environment.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M) , `auditd`(1M) , `audit_startup`(1M) , `audit.log`(4) , `audit_control`(4) , `device_allocate`(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

bsmconv, bsmunconv – Enable or disable the Basic Security Module (BSM)

SYNOPSIS

/etc/security/bsmconv [*rootdir*…]

/etc/security/bsmunconv [*rootdir*…]

DESCRIPTION

In the Solaris environment, the `bsmconv` and `bsmunconv` scripts are used to enable or disable the BSM features, auditing and device protection.

In the Trusted Solaris environment, the `bsmconv` and `bsmunconv` scripts are *not* used to enable or disable auditing, or to protect devices.

SUMMARY OF TRUSTED SOLARIS CHANGES

Enabling and disabling auditing in the Trusted Solaris environment does *not* use the `bsmconv` and `bsmunconv` commands. See *Trusted Solaris Audit Administration* for the procedure to disable and enable auditing. Devices are always protected in the Trusted Solaris environment.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M) , `auditd`(1M) , `audit_startup`(1M) , `audit.log`(4) , `audit_control`(4) , `device_allocate`(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

chk_encodings – Check the label encodings file syntax

SYNOPSIS

/usr/sbin/chk_encodings [–a] [–c *maxclass*] [*pathname*]

DESCRIPTION

`chk_encodings` checks the syntax of the label-encodings file specified by *pathname*; with the –a option, `chk_encodings` also prints a semantic analysis of the label-encodings file specified by *pathname*. If *pathname* is not specified, `chk_encodings` checks and analyzes `/etc/security/tsol/label_encodings`.

If label-encodings file analysis was requested, whatever analysis can be provided is written to the standard output file even if errors were found.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–a Provide a semantic analysis of the label encodings file.

–c *maxclass* Accept a maximum classification value of *maxclass* (default 255) in the label encodings file `CLASSIFICATIONS` section.

ERRORS

When successful, `chk_encodings` returns an exit status of `0` (true) and writes to the standard output file a confirmation that no errors were found in *pathname*. Otherwise, `chk_encodings` returns an exit status of nonzero (false) and writes an error diagnostic to the standard output file.

FILES

`/etc/security/tsol/label_encodings`
The label encodings file contains the classification names, words, constraints, and values for the defined labels of this system.

SEE ALSO

Trusted Solaris 7 Reference Manual

`label_encodings`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

chroot – Change root directory for a command

SYNOPSIS

/usr/sbin/chroot *newroot command*

DESCRIPTION

The `chroot` utility causes *command* to be executed relative to *newroot*. The meaning of any initial slashes (`/`) in the pathnames is changed to *newroot* for *command* and any of its child processes. Upon execution, the initial working directory is *newroot*.

Notice that redirecting the output of *command* to a file, as in the following example:

```
example# chroot newroot command >x
```

will create the file *x* relative to the original root of *command*, not the new one.

The new root pathname is always relative to the current root. Even if a `chroot` is currently in effect, the *newroot* argument is relative to the current root of the running process.

The `proc_chroot` privilege is required to run this command.

RETURN VALUES

The exit status of `chroot` is the return value of *command*.

EXAMPLES

EXAMPLE 1 Using the `chroot` utility.

The `chroot` utility provides an easy way to extract `tar` files (see `tar`(1)) written with absolute filenames to a different location:

```
example# cp /usr/sbin/static/tar /tmp
example# dd if=/dev/nrst0 | chroot /tmp tar xvf -
```

Note that `tar` is statically linked, so it is not necessary to copy any shared libraries to the *newroot* filesystem.

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the `proc_chroot` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`chroot`(2)

SunOS 5.7 Reference Manual

`cd`(1), `ttyname`(3C), `attributes`(5)

NOTES

Exercise extreme caution when referencing device files in the new root file system.

References by routines such as `ttyname`(3C) to stdin, stdout, and stderr will find that the device associated with the file descriptor is unknown after `chroot` is run.

NAME

pfsh, clist – Profile shell

SYNOPSIS

pfsh [–acefhiknprstuvx] [*argument*…]

DESCRIPTION

The profile shell is a modified version of the Bourne shell, sh(1) . Based on the user's profiles, pfsh restricts the commands that can be executed. Based on the profile definitions, pfsh determines which privileges, user ID (UID), and group ID (GID) to use in executing commands.

Usage

Refer to the sh(1) man page for a complete usage description. pfsh adds the clist command.

Commands

clist	Displays a list of the commands that are permitted for the user.
[–-hpniu]	
–h	Includes a hexadecimal list of the privileges assigned to each command in the command list.
–p	Includes a list of the privileges assigned to each command in the command list. The list is in text form.
–n	Includes a comma-separated decimal list of the privileges assigned to each command in the command list.
–i	Includes the UID and GID assigned to each command in the command list.
–u	Lists only those commands that are are unusable because the profile assigned privileges that pfsh did not inherit. (See WARNINGS .)

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

tsolprof(4) , tsoluser(4)

SunOS 5.7 Reference Manual

sh(1) , attributes(5)

WARNINGS

`pfsh` must inherit privileges in order to run commands with those privileges. Privileges for a command that are defined in a profile may not be inherited when `pfsh` runs that command. If such a command is executed, a warning message is printed and the command is run with no privileges.

Profiles are searched in the order specified in the user's `tsoluser` entry. If the same command appears in more than one profile, `pfsh` uses the first entry whose label range includes the sensitivity label of the process.

When it is executed, `pfsh` builds the list of allowable commands by reading the user's profiles. If any changes are made to the profiles while `pfsh` is running, the changes will not take effect until the shell is restarted.

NOTES

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

cron – Clock daemon

SYNOPSIS

/usr/sbin/cron

DESCRIPTION

The `cron` command starts a process that executes commands at specified dates and times. Regularly scheduled commands can be specified according to instructions found in `crontab` files in the directory `/var/spool/cron/crontabs`. Users can submit their own `crontab` file using the `crontab`(1) command. Commands which are to be executed only once may be submitted using the `at`(1) command.

`cron` only examines `crontab` or `at` command files during its own process initialization phase and when the `crontab` or `at` command is run. This reduces the overhead of checking for new or changed files at regularly scheduled intervals.

Since `cron` never exits, it should be executed only once. This is done routinely through `/etc/rc2.d/S75cron` at system boot time. The file `/etc/cron.d/FIFO` is used (among other things) as a lock file to prevent the execution of more than one instance of `cron`.

`cron` captures the output of the job's `stdout` and `stderr` streams, and, if it is non-empty, mails the output to the user. If the job does not produce output, no mail is sent to the user (unless the job is an `at`(1) job and the `–m` option was specified when the job was submitted).

Setting cron Defaults

To keep a log of all actions taken by `cron`, `CRONLOG=YES` (by default) must be specified in the `/etc/default/cron` file. If `CRONLOG=NO` is specified, no logging is done. Keeping the log is a user configurable option since `cron` usually creates huge log files.

The `PATH` for `user` cron jobs can be set using `PATH=` in `/etc/default/cron`. The `PATH` for `root` cron jobs can be set using `SUPATH=` in `/etc/default/cron`. The security implications of setting `PATH` and `SUPATH` should be carefully considered.

Example `/etc/default/cron` file:

```
CRONLOG=YES
PATH=/usr/bin:/usr/ucb:
```

This example enables logging and sets the default `PATH` used by non-root jobs to /usr/bin:/usr/ucb:. Root jobs will continue to use `/usr/sbin:/usr/bin`.

`/etc/cron.d/logchecker` is a script that checks to see if the log file has exceeded the system ulimit. If so, the log file is moved to `/var/cron/olog`.

SUMMARY OF TRUSTED SOLARIS CHANGES

The job directories `/var/spool/cron/crontabs` and `/var/spool/cron/atjobs` are multilevel directories (MLDs). The MLD job directory provides for the separation of job files at different sensitivity labels. Hence, there can be multiple `crontab` files for a single user within the `crontabs` directory, but each `crontab` file is at a different sensitivity label. In addition, a user can have multiple `atjob` files at different sensitivity labels.

Each `crontab` file in the `crontabs` MLD and each `atjob` file in the `atjobs` MLD has an ancillary file containing information used by `cron` to set up a job. The `crontab` ancillary files are named `username.ad`, and the `atjobs` ancillary files are name `jobname.ad`.

The clock daemon must be started with the root userid, must have the `PAF_TRUSTED_PATH` process attribute, and it must inherit the following privileges: `file_mac_write`, `net_mac_read`, `proc_setid`, `proc_setsl`, `proc_setil`, `proc_setclr`, `sys_audit`, `proc_audit_tcb`, `file_dac_read`, and `file_owner`.

If the clock daemon has the `PAF_PRIV_DEBUG` process attribute, it passes the attribute on to the job to be executed. Because the daemon never has the `PAF_TOKMAPPER`, `PAF_DISKLESS_BOOT`, and `PAF_SELAGNT` process attributes, these attributes will not be passed on to the job to be executed.

The clock daemon creates the `/var/cron/log` file at the `ADMIN_HIGH` sensitivity label.

In the default Trusted Solaris environment, there are two pairs of `crontab` and its ancillary file for the root userid: one pair at the `ADMIN_HIGH` sensitivity label, and the other pair at the `ADMIN_LOW` sensitivity label.

FILES

`/etc/cron.d`	main `cron` directory
`/etc/cron.d/FIFO`	used as a lock file
`/etc/default/cron`	contains cron default settings
`/var/cron/log`	`cron` history information
`/var/spool/cron`	spool area
`/etc/cron.d/logchecker`	moves log file to `/var/cron/olog` if log file exceeds system ulimit.
`/etc/cron.d/queuedefs`	queue description file for `at`, `batch`, and `cron`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`at`(1), `crontab`(1)

SunOS 5.7 Reference Manual

`sh`(1), `queuedefs`(4), `attributes`(5)

DIAGNOSTICS

A history of all actions taken by `cron` is stored in `/var/cron/log` and (possibly) `/var/cron/olog`.

NAME

deallocate – Device deallocation

SYNOPSIS

deallocate [–s] *device*

deallocate [–s] [–F] *device*

deallocate [–s] –I

deallocate [–s] –R [*device*]

DESCRIPTION

`deallocate` deallocates a device allocated to the evoking user. *device* can be a device defined in `device_allocate`(4) or one of the device special files associated with the device. It resets the ownership and the permission on all device special files associated with *device*, disabling the user's access to that device. This option can be used by a privileged user to remove access to the device by another user.

When deallocation or forced deallocation is performed, the appropiate device cleaning program is executed, based on the contents of `device_allocate`(4). These cleaning programs are normally stored in `/etc/security/lib`. `deallocate` requires the `file_chown`, `file_dac_read`, `file_mac_read`, `file_setdac`, and `sys_audit` privileges to be successful. In addition, certain options require the trusted path attribute to be successful.

OPTIONS

device	Deallocate the device associated with the device special file specified by *device*.
–s	Silent. Suppress any diagnostic output.
–F *device*	Force deallocation of the device associated with the file specified by *device*. Only the super user is permitted to use this option.
–I	Force deallocation of all allocatable devices. This option requires the trusted path attribute to be successful. This option should only be used at system initialization.
–R *device*	Reset the specified device to be allocatable. All associated physical device nodes listed in the `device_maps` file for the specified *device* will be reset to the deallocated mode and label. Intended as a means for reclaiming a device from a state of error, this option requires the trusted path attribute to be successful. If the specified device is allocated or if the device is a nonallocatable device, this option will fail. If no device is specified, the command is applied to all allocatable devices.

FILES

`/etc/security/device_allocate`	Mandatory access control file for devices.

`/etc/security/device_maps`	List of physical devices associated with a device name and type.
`/etc/security/dev/*`	Device storage area.
`/etc/security/lib/*`	Directory of device cleaning scripts.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To run successfully, `deallocate` requires the `file_chown`, `file_dac_read`, `file_mac_read`, `file_setdac`, and `sys_audit` privileges. In addition, the `–F` option and the new `–R` option require the trusted path attribute.

DIAGNOSTICS

`deallocate` returns a non-zero exit status in the event of an error.

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M), `device_allocate`(4), `device_maps`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

device_clean – Device clean programs

SYNOPSIS

/etc/security/lib/*device-clean-program* *character-media-label-string* [–A | –D]

DESCRIPTION

An allocatable device may optionally have a device clean program. Device clean programs are specified in the *device-clean* field in the `device_allocate`(4) file. Device clean programs are invoked by `allocate`(1M) and `deallocate`(1M) to clean device states, registers, and any residual information in the device before it is allocated to a user as required by the *object reuse* policy, and also to ensure proper media labeling by asking the user to confirm the correct labeled media is inserted in the device on allocation and by asking the user to confirm removal of the media and affix correct label on the media.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

OPTIONS

character-media-label-string

Provide CMW Label of the device. This information is used by most device clean programs in a prompt to remind the user to affix a correct label to the removable media.

–A The device clean program is invoked from `allocate`(1M) command before the device is allocated to a user.

–D The device clean program is invoked from `deallocate`(1M) command after the device is deallocated from a user.

FILES

`/etc/security/device_allocate` Mandatory access control file for devices

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M), `deallocate`(1M), `device_allocate`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

devpolicy – Configure device policy

SYNOPSIS

devpolicy [–s | –v][–f *policyfile*] [–r *rootdir*]

DESCRIPTION

devpolicy reads the /etc/security/tsol/device_policy file and, for each device node in the /devices tree, constructs device policy information and downloads the information to the kernel.

To be successful, devpolicy requires the trusted path attribute and the sys_devices privilege. If device policy has been downloaded by an earlier invocation of the command, devpolicy will fail. If a device has two or more device nodes that are assigned different policies in the device_policy file, devpolicy displays a warning.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsr

OPTIONS

–s	Silent mode; suppresses non-critical warning messages.
–v	Verbose mode; displays all warning messages, including messages for unknown devices.
–f *policyfile*	Read *policyfile* instead of /etc/security/tsol/device_policy.
–r *rootdir*	Find devices under *rootdir* instead of /devices.

EXIT STATUS

0	Successful.
>0	An error occurred.

FILES

/etc/security/tsol/device_policy	Security policy configuration file for devices

SEE ALSO

Trusted Solaris 7 Reference Manual

drvconfig(1M), device_policy(4)

SunOS 5.7 Reference Manual

attributes(5)

NAME

dfmounts – Display mounted resource information

SYNOPSIS

dfmounts [–F *FSType*] [–h] [–o *specific_options*] [*restriction*…]

DESCRIPTION

`dfmounts` shows the local resources shared through a distributed file system *FSType* along with a list of clients that have the resource mounted. If *restriction* is not specified, `dfmounts` shows file systems that are currently shared on any NFS server. *specific_options* as well as the availability and semantics of *restriction* are specific to particular distributed file system types.

If `dfmounts` is entered without arguments, all remote resources currently mounted on the local system are displayed, regardless of file system type.

dfmounts Output

The output of `dfmounts` consists of an optional header line (suppressed with the –h flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

resource server pathname clients ...

where:

resource Specifies the resource name that must be given to the `mount`(1M) command.

server Specifies the system from which the resource was mounted.

pathname Specifies the pathname that must be given to the `share`(1M) command.

clients Is a comma-separated list of systems that have mounted the resource. Clients are listed in the form *domain.*, *domain.system*, or *system*, depending on the file system type.

A field may be null. Each null field is indicated by a hyphen (–) unless the remainder of the fields on the line are also null; in which case, the hyphen may be omitted.

Fields with whitespace are enclosed in quotation marks (" ").

OPTIONS

–F *FSType* Specify filesystem type. Defaults to the first entry in `/etc/dfs/fstypes`. *Note*: currently the only valid *FSType* is `nfs`.

–h Suppress header line in output.

–o *specific_options* Specify options specific to the filesystem provided by the –F option. *Note*: currently no options are supported.

FILES

`/etc/dfs/fstypes` file system types

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The output fields show the resource and pathname that must be given to the Trusted Solaris versions of the `mount` and `share` commands.

SEE ALSO

Trusted Solaris 7 Reference Manual

`dfshares`(1M), `mount`(1M), `share`(1M), `unshare`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

dfshares – list available resources from remote or local systems

SYNOPSIS

dfshares [–F *FSType*] [–h] [–o *specific_options*] [*server*…]

DESCRIPTION

`dfshares` provides information about resources available to the host through a distributed file system of type *FSType*. *specific_options* as well as the semantics of *server* are specific to particular distributed file systems.

If `dfshares` is entered without arguments, all resources currently shared on the local system are displayed, regardless of file system type.

The output of `dfshares` consists of an optional header line (suppressed with the –h flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

resource server access transport

where

resource Specifies the resource name that must be given to the `mount`(1M) command.

server Specifies the name of the system that is making the resource available.

access Specifies the access permissions granted to the client systems, either `ro` (for read-only) or `rw` (for read/write). If `dfshares` cannot determine access permissions, a hyphen (–) is displayed.

transport Specifies the transport provider over which the resource is shared.

A field may be null. Each null field is indicated by a hyphen (–) unless the remainder of the fields on the line are also null; in which case, the hyphen may be omitted.

OPTIONS

–F *FSType* Specify filesystem type. Defaults to the first entry in `/etc/dfs/fstypes`.

–h Suppress header line in output.

–o *specific_options* Specify options specific to the filesystem provided by the –F option.

FILES

`/etc/dfs/fstypes`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO dfmounts(1M), mount(1M), share(1M), unshare(1M), attributes(5)

NAME

dispadmin – Process scheduler administration

SYNOPSIS

dispadmin –l

dispadmin –c *class* –g [–r *res*]

dispadmin –c *class* –s *file*

DESCRIPTION

The dispadmin command displays or changes process scheduler parameters while the system is running.

dispadmin does limited checking on the values supplied in *file* to verify that they are within their required bounds. The checking, however, does not attempt to analyze the effect that the new values have on the performance of the system. Inappropriate values can have a negative effect on system performance. (See *System Administration Guide, Volume I.*)

OPTIONS

–l Lists the scheduler classes currently configured in the system.

–c *class* Specifies the class whose parameters are to be displayed or changed. Valid *class* values are: RT for the real-time class, TS for the time-sharing class, and IA for the inter-active class. The time-sharing and inter-active classes share the same scheduler, so changes to the scheduling parameters of one will change those of the other.

–g Gets the parameters for the specified class and writes them to the standard output. Parameters for the real-time class are described in rt_dptbl(4). Parameters for the time-sharing and inter-active classes are described in ts_dptbl(4).

–r *res* When using the –g option you may also use the –r option to specify a resolution to be used for outputting the time quantum values. If no resolution is specified, time quantum values are in milliseconds. If *res* is specified it must be a positive integer between 1 and 1000000000 inclusive, and the resolution used is the reciprocal of *res* in seconds. For example, a *res* value of 10 yields time quantum values expressed in tenths of a second; a *res* value of 1000000 yields time quantum values expressed in microseconds. If the time quantum cannot be expressed as an integer in the specified resolution, it is rounded up to the next integral multiple of the specified resolution.

–s *file* Sets scheduler parameters for the specified class using the values in *file*. These values overwrite the current values in memory—they become the parameters that control scheduling of processes in the specified class. The values in *file* must be in the format output by the –g option. Moreover, the values must describe a table that is the same size (has same number of priority levels) as the table being

overwritten. The `sys_config` privilege is required for the `–s` option to succeed.

Note: The `–g` and `–s` options are mutually exclusive: you may not retrieve the table at the same time you are overwriting it.

EXAMPLES

EXAMPLE 1 Retrieving the current scheduler parameters for the real-time class.

The following command retrieves the current scheduler parameters for the real-time class from kernel memory and writes them to the standard output. Time quantum values are in microseconds.

```
dispadmin –c RT –g –r 1000000
```

EXAMPLE 2 Overwriting the current scheduler parameters for the real-time class.

The following command overwrites the current scheduler parameters for the real-time class with the values specified in `rt.config`.

```
dispadmin –c RT –s rt.config
```

EXAMPLE 3 Retrieving the current scheduler parameters for the time-sharing class.

The following command retrieves the current scheduler parameters for the time-sharing class from kernel memory and writes them to the standard output. Time quantum values are in nanoseconds.

```
dispadmin –c TS –g –r 1000000000
```

EXAMPLE 4 Overwriting the current scheduler parameters for the time-sharing class.

The following command overwrites the current scheduler parameters for the time-sharing class with the values specified in `ts.config`.

```
dispadmin –c TS –s ts.config
```

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed with the `–s` option, this command needs the `sys_config` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`priocntl`(2)

SunOS 5.7 Reference Manual

`priocntl`(1), `rt_dptbl`(4), `ts_dptbl`(4), `attributes`(5)

DIAGNOSTICS

`dispadmin` prints an appropriate diagnostic message if it fails to overwrite the current scheduler parameters due to lack of required permissions or a problem with the specified input file.

NAME

dl_booting, dl_restore – Inform the kernel that a machine is in the state of disklessly booting or in the normal state

SYNOPSIS

/usr/sbin/dl_booting [*hostname* | *ip_address*]

/usr/sbin/dl_restore [*hostname* | *ip_address*]

DESCRIPTION

`dl_booting` informs the kernel that the machine specified by *hostname* or *ip_address* is in the state of booting disklessly. Hence, until the kernel is notified that the machine has reverted to the normal state, it must be viewed as an unlabeled host, and only processes with the `PAF_DISKLESS_BOOT` process attribute can communicate with the machine while it is in the booting state. In the normal state, packets exchanged are properly labeled.

`dl_restore` informs the kernel that the machine specified by the hostname or IP address is now in the normal state.

To succeed, both `dl_booting` and `dl_restore` must inherit the `sys_net_config` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, both `dl_booting` and `dl_restore` must inherit the `sys_net_config` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`chstate`(2)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

dl_booting, dl_restore – Inform the kernel that a machine is in the state of disklessly booting or in the normal state

SYNOPSIS

/usr/sbin/dl_booting [*hostname* | *ip_address*]

/usr/sbin/dl_restore [*hostname* | *ip_address*]

DESCRIPTION

`dl_booting` informs the kernel that the machine specified by *hostname* or *ip_address* is in the state of booting disklessly. Hence, until the kernel is notified that the machine has reverted to the normal state, it must be viewed as an unlabeled host, and only processes with the `PAF_DISKLESS_BOOT` process attribute can communicate with the machine while it is in the booting state. In the normal state, packets exchanged are properly labeled.

`dl_restore` informs the kernel that the machine specified by the hostname or IP address is now in the normal state.

To succeed, both `dl_booting` and `dl_restore` must inherit the `sys_net_config` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, both `dl_booting` and `dl_restore` must inherit the `sys_net_config` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`chstate`(2)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

dminfo – Report information about a device entry in a device maps file

SYNOPSIS

dminfo [–v] [–a] [–f *pathname*]

dminfo [–v] [–a] [–f *pathname*] –n *dev –name…*

dminfo [–v] [–a] [–f *pathname*] –d *dev –path…*

dminfo [–v] [–a] [–f *pathname*] –t *dev –type…*

dminfo [–v] [–f *pathname*] –u *dm -entry*

DESCRIPTION

`dminfo` reports and updates information about the `device_maps`(4) file.

OPTIONS

–v	Verbose. Print the requested entry or entries, one line per entry, on the standard output. If no entries are specified, all are printed.
–a	Succeed if any of the requested entries are found. If used with –v, all entries that match the requested case(s) are printed.
–f *pathname*	Use a device_maps file with *pathname* instead of `/etc/security/device_maps`.
–n *dev–name*	Search by *dev–name.* Search `device_maps`(4) for a *device_name* field matching *dev–name.* This option cannot be used with –d, –t, or –u.
–d *dev–path*	Search by *dev–path.* Search `device_maps`(4) for a device special pathname in the *device_list* field matching the *dev–path* argument. This option cannot be used with –n, –t, or –u.
–t *dev–type*	Search by *dev–type.* Search `device_maps`(4) for a *device_type* field matching the given *dev–type.* This option cannot be used with –d, –n, or –u.
–u *dm–entry*	Update the `device_maps`(4) file. This option is provided to add entries to the `device_maps`(4) file. The *dm–entry* must be a complete `device_maps` file entry. The *dm–entry* has fields, as in the `device_maps` file. It uses the colon (`:`) as a field separator, and white space as the *device_list* subfield separators. The *dm–entry* is not made if any fields are missing, or if the *dm–entry* would be a duplicate. This option requires the trusted path and write access to the `device_maps` file.

DIAGNOSTICS

`dminfo` returns an exit code of `0` if successful, `1` if the request failed, and `2` if the invocation syntax was incorrect.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –u option requires the trusted path and write access to the `/etc/security/device_maps` file. The calling process may use the privileges `file_mac_write` and `file_dac_write` to override access restrictions.

FILES

`/etc/security/device_maps`	List of physical devices associated with a device name and type.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`device_maps`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

drvconfig – Configure the /devices directory

SYNOPSIS

drvconfig [–bn] [–a *alias_name*] [–c *class_name*] [–i *drivername*] [–m *major_num*] [–r *rootdir*]

DESCRIPTION

The default operation of `drvconfig` is to create the `/devices` directory tree that describes, in the filesystem namespace, the hardware layout of a particular machine. Hardware devices present on the machine and powered on as well as pseudo-drivers are represented under `/devices`. Normally this command is run automatically after a new driver has been installed (with `add_drv`(1M)) and the system has been rebooted.

/etc/minor_perm file

`drvconfig` reads the `/etc/minor_perm` file to obtain permission information and applies the permissions only to nodes that it has just created. It does not change permissions on already existing nodes. The format of the `/etc/minor_perm` file is as follows:

name:*minor_name permissions owner group*

minor_name may be the actual name of the minor node, or contain shell metacharacters to represent several minor nodes (see `sh`(1)).

For example:

```
sd:* 0640 root sys
zs:[a-z],cu 0600 uucp uucp
mm:kmem 0640 root bin
```

The first line sets all devices exported by the `sd` node to `0640` permissions, owned by `root`, with group `sys`. In the second line, devices such as `a,cu` and `z,cu` exported by the `zs` driver are set to `0600` permission, owned by `uucp`, with group `uucp`. In the third line the `kmem` device exported by the `mm` driver is set to `0640` permission, owned by `root`, with group `bin`.

/etc/security/tsol/minor_perm.adjunct file

`drvconfig` reads the `/etc/security/tsol/minor_perm.adjunct` file to obtain label information and applies the labels to nodes that it has just created. `drvconfig` does not change labels on already existing nodes. The format of the file is:

name:*minor_name [SL]*

minor_name is the name of the minor node; shell metacharacters may be used to represent several minor nodes (see `sh`(1)). Labels can be represented in hex format which `add_drv`(1M) converts when an entry is added to the file. For readability in the example shown below, the first two lines would be entered as a single line, as would the last two lines.

```
SD:* [0x7fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff \
ffffffff]
mm:kmem [0x7fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff \
fffffffffff]
```

The above example sets all devices exported by the `sd` node to have a sensitivity label of `ADMIN_HIGH`. The `kmem` device exported by the `mm` driver is set to have a sensitivity label of `ADMIN_HIGH`.

OPTIONS

The following options may be of use to system administrators and driver developers:

`–i` *drivername* — Only configure the devices for the named driver. The following options are used by the implementation of `add_drv`(1M) and `rem_drv`(1M), and may not be supported in future versions of the Solaris and Trusted Solaris environments.

`–b` — Add a new major number to name binding into the kernel's internal `name_to_major` tables. This option is not normally used directly, but is used by other utilities such as `add_drv`(1M). Use of the `–b` option requires that `–i` and `–m` be used also. No `/devices` entries are created.

`–n` — Do not try to load and attach any drivers, or if the `–i` option is given, do not try to attach the driver named *drivername*.

`–a` *alias_name* — Add the name *alias_name* to the list of aliases that this driver is known by. This option, if used, must be used with the `–m` *major_num*, the `–b` and the `–i` *drivername* options.

`–c` *class_name* — The driver being added to the system exports the class *class_name*. This option is not normally used directly, but is used by other utilities. It is only effective when used with the `–b` option.

`–m` *major_num* — Specify the major number *major_num* for this driver to add to the kernel's `name_to_major` binding tables.

`–r` *rootdir* — Build the device tree under the directory specified by *rootdir* instead of the default `/devices` directory.

EXIT STATUS

`0` — Successful completion.

non-zero — An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `/etc/security/tsol/minor_perm.adjunct` file is used to record the sensitivity label of devices.

FILES

`/devices`	Device nodes directory
`/etc/minor_perm`	Minor mode permissions
`/etc/security/tsol/minor_perm.adjunct` Default label	
`/etc/name_to_major`	Major number binding
`/etc/driver_classes`	Driver class binding file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`add_drv`(1M), `modload`(1M), `modunload`(1M), `rem_drv`(1M)

SunOS 5.7 Reference Manual

`sh`(1), `devlinks`(1M), `disks`(1M), `modinfo`(1M), `ports`(1M), `tapes`(1M), `path_to_inst`(4) `attributes`(5)

NOTES

This document does not constitute an API. `/etc/minor_perm`, `/etc/security/tsol/minor_perm.adjunct`, `/etc/name_to_major`, `/etc/driver_classes`, and `/devices` may not exist or may have different contents or interpretations in a future release. The existence of this notice does not imply that any other documentation that lacks this notice constitutes an API.

NAME

du – Summarize disk usage

SYNOPSIS

/usr/bin/du [–adkr] [–s | –o][–M] [*file*...]

/usr/xpg4/bin/du [–a | –s][–krx] [*file*...]

DESCRIPTION

The du utility writes to standard output the size of the file space allocated to, and the size of the file space allocated to each subdirectory of, the file hierarchy rooted in each of the specified files. The size of the file space allocated to a file of type directory is defined as the sum total of space allocated to all files in the file hierarchy rooted in the directory plus the space allocated to the directory itself.

Files with multiple links will be counted and written for only one entry. The directory entry that is selected in the report is unspecified. By default, file sizes are written in 512-byte units, rounded up to the next 512-byte unit.

/usr/xpg4/bin/du

When du cannot obtain file attributes or read directories (see stat(2)), it will report an error condition and the final exit status will be affected.

OPTIONS

The following options are supported for /usr/bin/du and /usr/xpg4/bin/du:

–k Write the files sizes in units of 1024 bytes, rather than the default 512-byte units.

–s Instead of the default output, report only the total sum for each of the specified files.

/usr/bin/du

The following options are supported for /usr/bin/du only:

–a In addition to the default output, report the size of each file not of type directory in the file hierarchy rooted in the specific file.

–d Do not cross filesystem boundaries. For example, du –d / reports usage only on the root partition.

–L Process symbolic links by using the file or directory which the symbolic link references, rather than the link itself.

–o Do not add child directories' usage to a parent's total. Without this option, the usage listed for a particular directory is the space taken by the files in that directory, as well as the files in all directories beneath it. This option does nothing if –s is used.

–r Generate messages about directories that cannot be read, files that cannot be opened, and so forth, rather than being silent (the default).

–M Process all accessible single-level directories while descending multilevel directories.

/usr/xpg4/bin/du

The following options are supported for /usr/xpg4/bin/du only:

–a In addition to the default output, report the size of each file not of type directory in the file hierarchy rooted in the specified file. Regardless of

the presence of the –a option, non-directories given as *file* operands will always be listed.

–r By default, generate messages about directories that cannot be read, files that cannot be opened, and so forth.

–x When evaluating file sizes, evaluate only those files that have the same device as the file specified by the *file* operand.

OPERANDS

The following operand is supported:

file The path name of a file whose size is to be written. If *file* is not specified, the current directory is used.

OUTPUT

The output from du consists of the amount of the space allocated to a file and the name of the file.

USAGE

See largefile(5) for the description of the behavior of du when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ENVIRONMENT VARIABLES

See environ(5) for descriptions of the following environment variables that affect the execution of du: LC_CTYPE, LC_MESSAGES, and NLSPATH.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

/usr/bin/du

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
CSI	enabled

/usr/xpg4/bin/du

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWxcu4
CSI	enabled

SUMMARY OF TRUSTED SOLARIS CHANGES

The –M option processes SLDs when descending multilevel directories.

SEE ALSO

Trusted Solaris 7 Reference Manual

`stat`(2)

SunOS 5.7 Reference Manual

`ls`(1), `attributes`(5), `environ`(5)

NOTES

A file with two or more links is counted only once. If, however, there are links between files in different directories where the directories are on separate branches of the file system hierarchy, `du` will count the excess files more than once.

Files containing holes will result in an incorrect block count.

NAME

eeprom – EEPROM Display and Load Utility

SYNOPSIS

SPARC

/usr/platform/ *platform-name* /sbin/eeprom [–] [–f *device*] [*parameter* [=*value*] …]

x86

/usr/platform/ *platform-name* /sbin/eeprom [–] [–f *device*] [–I] [*mmu-modlist*] [*parameter* [=*value*] …]

DESCRIPTION

`eeprom` displays or changes the values of parameters in the EEPROM. It processes parameters in the order given. When processing a *parameter* accompanied by a *value*, `eeprom` makes the indicated alteration to the EEPROM; otherwise it displays the *parameter*'s value. When given no parameter specifiers, `eeprom` displays the values of all EEPROM parameters. A '–' (hyphen) flag specifies that parameters and values are to be read from the standard input (one *parameter* or *parameter=value* per line).

`eeprom` verifies the EEPROM checksums and complains if they are incorrect.

platform-name is the name of the platform implementation and can be found using the –`i` option of `uname`(1).

SPARC

SPARC based systems implement firmware password protection with `eeprom` using the `security-mode`, `security-password`, and `security-#badlogins` properties.

x86

EEPROM storage is simulated using a file residing in the platform specific boot area. The `/platform/`*platform-name*`/boot/solaris/bootenv.rc` file simulates EEPROM storage.

Because x86 based systems typically implement password protection in the sytem BIOS, there is no support for password protection in the `eeprom` program. While it is possible to set the `security-mode`, `security-password`, and `security-#badlogins` properties on x86 based systems, these properties have no special meaning or behavior on x86 based systems.

OPTIONS

–`f` *device* Use *device* as the EEPROM device.

x86 Only

–`I` Initialize boot properties on an x86 based system. Only `init`(1M) run-level initialization scripts should use this option.

OPERANDS

x86 Only

mmu-modlist A colon-separated list of candidate modules that implement memory management. If *mmu-modlist* is defined, it overrides the default list derived from the memory configuration on x86 based systems. Instead, the first module in the list that is found in `/platform/`*platform-name*`/kernel/mmu` is used.

NVRAM CONFIGURATION PARAMETERS

Not all OpenBoot systems support all parameters. Defaults may vary depending on the system and the PROM revision.

`auto-boot?` If `true`, boot automatically after power-on or reset. The default value is `true`.

`ansi-terminal?` Configuration variable used to control the behavior of the terminal emulator. The value `false` makes the terminal emulator stop interpreting ANSI escape sequences, instead just echoing them to the output device. The default value is `true`.

`boot-command` Command executed if `auto-boot?` is `true`. The default value is `boot`.

`boot-device` Device from which to boot. *boot-device* may contain 0 or more device specifiers separated by spaces. Each device specifier may be either a prom device alias or a prom device path. The boot prom will attempt to open each successive device specifier in the list beginning with the first device specifier. The first device specifier which opens successfully will be used as the device to boot from. The default value is `disk net`.

`boot-file` File to boot (an empty string lets the secondary booter choose default). The default is an empty string.

`boot-from` Boot device and file (OpenBoot PROM version 1.*x* only). The default value is `vmunix`.

`boot-from-diag` Diagnostic boot device and file (OpenBoot PROM version 1.*x* only). The default value is `le( )unix`.

`com`*X*`-noprobe` Where *X* is the number of the serial port, prevents device probe on serial port *X*.

`diag-device` Diagnostic boot source device. The default value is `net`.

`diag-file` File from which to boot in diagnostic mode. The default is an empty string.

`diag-level` Diagnostics level. Values include `off`, `min`, `max`, and `menus`. There may be additional platform-specific values. When set to `off`, POST is not called. If POST is called, the value

	is made available as an argument to, and is interpretted by POST. The default value is `platform-dependent`.
`diag-switch?`	If `true`, run in diagnostic mode. The default value is `true`.
`fcode-debug?`	If `true`, include name parameter for plug-in device FCodes. The default value is `false`.
`hardware-revision`	System version information.
`input-device`	Input device used at power-on (usually `keyboard`, `ttya`, or `ttyb`). The default is `keyboard`.
`keyboard-click?`	If `true` enable keyboard click. The default value is `false`.
`keymap`	Keymap for custom keyboard.
`last-hardware-update`	System update information.
`load-base`	Default load address for client programs. The default value is `16384`.
`local-mac-address?`	If `true`, network drivers use their own MAC address, not system's. The default value is `false`.
`mfg-mode`	Manufacturing mode argument for POST. Possible values include `off` or `chamber`. The value is passed as an argument to POST. The default value is `off`.
`mfg-switch?`	If `true`, repeat system self-tests until interrupted with **`STOP-A`**. The default value is `false`.
`nvramrc`	Contents of NVRAMRC. The default value is `empty`.
`oem-banner`	Custom OEM banner (enabled by setting `oem-banner?` to `true`). The default is an empty string.
`oem-banner?`	If `true`, use custom OEM banner. The default value is `false`.
`oem-logo`	Byte array custom OEM logo (enabled by setting `oem-logo?` to `true`). Displayed in hexadecimal.

`oem-logo?`	If `true`, use custom OEM logo (else, use Sun logo). The default value is `false`.
`output-device`	Output device used at power-on (usually `screen`, `ttya`, or `ttyb`). The default value is `screen`.
`sbus-probe-list`	Which SBus slots are probed and in what order. The default is `0123`.
`screen-#columns`	Number of on-screen columns (characters/line). The default is `80`.
`screen-#rows`	Number of on-screen rows (lines). The default is `34`.
`scsi-initiator-id`	SCSI bus address of host adapter, range 0-7. The default is `7`.
`sd-targets`	Map SCSI disk units (OpenBoot PROM version 1.*x* only). The default is `31204567`, which means that unit 0 maps to target `3`, unit 1 maps to target `1`, and so on.
`security-#badlogins`	Number of incorrect security password attempts. This property has no special meaning or behavior on x86 based systems.
`security-mode`	Firmware security level (options: `none`, `command`, or `full`). If set to `command` or `full`, system will prompt for PROM security password. The default is `none`. This property has no special meaning or behavior on x86 based systems.

`security-password`	Firmware security password (never displayed). Can be set only when `security-mode` is set to `command` or `full`.
	This property has no special meaning or behavior on x86 based systems.
	`example# eeprom security-password=`
	`Changing PROM password:`
	`New password:`
	`Retype new password:`

`selftest-#megs`	Megabytes of RAM to test. Ignored if `diag-switch?` is `true`. The default is `1`.
`skip-vme-loopback?`	If `true`, POST does not do VMEbus loopback tests. The default is `false`.
`st-targets`	Map SCSI tape units (OpenBoot PROM version 1.*x* only). The default is `45670123`, which means that unit 0 maps to target `4`, unit 1 maps to target `5`, and so on.
`sunmon-compat?`	If `true`, display Restricted Monitor prompt (>). The default is `false`.
`testarea`	One-byte scratch field, available for read/write test. The default is `0`.
`tpe-link-test?`	Enable 10baseT link test for built-in twisted pair Ethernet. The default is `true`.
`ttya-mode`	TTYA (baud rate, #bits, parity, #stop, handshake). The default is `9600,8,n,1,–`. Fields, in left-to-right order, are: baud rate: 110, 300, 1200, 4800, 9600... data bits: 5, 6, 7, 8 parity: n(none), e(even), o(odd), m(mark), s(space) stop bits: 1, 1.5, 2 handshake: –(none), h(hardware:rts/cts), s(software:xon/xoff)
`ttyb-mode`	TTYB (baud rate, #bits, parity, #stop, handshake). The default is `9600,8,n,1,–`. Fields, in left-to-right order, are: baud rate: 110, 300, 1200, 4800, 9600... data bits: 5, 6, 7, 8 stop bits: 1, 1.5, 2 parity: n(none), e(even), o(odd), m(mark), s(space)

	handshake: –(none), h(hardware:rts/cts), s(software:xon/xoff)
`ttya-ignore-cd`	If `true`, operating system ignores carrier-detect on TTYA. The default is `true`.
`ttyb-ignore-cd`	If `true`, operating system ignores carrier-detect on TTYA. The default is `true`.
`ttya-rts-dtr-off`	If `true`, operating system does not assert DTR and RTS on TTYA. The default is `false`.
`ttyb-rts-dtr-off`	If `true`, operating system does not assert DTR and RTS on TTYB. The default is `false`.
`use-nvramrc?`	If `true`, execute commands in NVRAMRC during system start-up. The default is `false`.
`version2?`	If `true`, hybrid (1.*x*/2.*x*) PROM comes up in version 2.*x*. The default is `true`.
`watchdog-reboot?`	If `true`, reboot after watchdog reset. The default is `false`.

EXAMPLES

EXAMPLE 1 Changing the number of megabytes of RAM.

The following example demonstrates the method for changing from one to two the number of megabytes of RAM that the system will test.

```
example# eeprom selftest-#megs
selftest-#megs=1

example# eeprom selftest-#megs=2

example# eeprom selftest-#megs
selftest-#megs=2
```

EXAMPLE 2 Setting the `auto-boot?` parameter to `true`.

The following example demonstrates the method for setting the `auto-boot?` parameter to `true`.

```
example# eeprom auto-boot?=true
```

When the `eeprom` command is executed in user mode, the parameters with a trailing question mark (?) need to be enclosed in double quotation marks (" ") to prevent the shell from interpreting the question mark.

```
example% eeprom "auto-boot?"=true
```

Preceding the question mark with an escape character (\) will also prevent the shell from interpreting the question mark.

SUMMARY OF TRUSTED SOLARIS CHANGES

For administrative users who alter the EEPROM contents, this command must be invoked with effective user ID of `0`.

FILES

`/dev/openprom`
device file

`/usr/platform/`*platform-name*`/sbin/eeprom`
Platform-specific version of `eeprom`. Use `uname -i`. to obtain *platform-name*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`uname`(2)

SunOS 5.7 Reference Manual

`passwd`(1), `sh`(1), `attributes`(5)

NAME

format – Disk partitioning and maintenance utility

SYNOPSIS

format [–f *command-file*] [–l *log-file*] [–x *data-file*] [–d *disk-name*] [–t *disk-type*] [–p *partition-name*] [–s] [–m] [–M] [–e] [*disk-list*]

DESCRIPTION

`format` enables you to format, label, repair and analyze disks on your system. Unlike previous disk maintenance programs, `format` runs under SunOS. Because there are limitations to what can be done to the system disk while the system is running, `format` is also supported within the memory-resident system environment. For most applications, however, running `format` under SunOS is the more convenient approach.

`format` first uses the disk list defined in *data-file* if the –x option is used. `format` then checks for the `FORMAT_PATH` environment variable, a colon-separated list of filenames and / or directories. In the case of a directory, `format` searches for a file named `format.dat` in that directory; a filename should be an absolute pathname, and is used without change. `format` adds all disk and partition definitions in each specified file to the working set. Multiple identical definitions are silently ignored. If `FORMAT_PATH` is not set, the path defaults to `/etc/format.dat`.

disk-list is a list of disks in the form `c?t?d?` or `/dev/rdsk/c?t?d?s?`. With the latter form shell wildcard specifications are supported. For example, specifying `/dev/rdsk/c2*` will cause `format` to work on all drives connected to controller `c2` only. If no *disk-list* is specified, `format` lists all the disks present in the system.

OPTIONS

The following options are supported:

–d *disk-name* Specify which disk should be made current upon entry into the program. The disk is specified by its logical name (for instance, –d `c0t1d0`). This can also be accomplished by specifying a single disk in the disk list.

–e Enable SCSI expert menu. Note this option is not recommended for casual use.

–f *command-file* Take command input from *command-file* rather than the standard input. The file must contain commands that appear just as they would if they had been entered from the keyboard. With this option, `format` does not issue `continue?` prompts; there is no need to specify y(es) or n(o) answers in the *command-file*. In non-interactive mode, `format` does not initially expect the input of a disk selection number. The user must specify the current working disk with the –d *disk-name* option when format is invoked,

	or specify `disk` and the disk selection number in the *command-file*.
–`l` *log-file*	Log a transcript of the `format` session to the indicated *log-file*, including the standard input, the standard output and the standard error.
–`m`	Enable extended messages. Provides more detailed information in the event of an error.
–`M`	Enable extended and diagnostic messages. Provides extensive information on the state of a SCSI device's mode pages, during formatting.
–`p` *partition-name*	Specify the partition table for the disk which is current upon entry into the program. The table is specified by its name as defined in the data file. This option can only be used if a disk is being made current, and its type is either specified or available from the disk label.
–`t` *disk-type*	Specify the type of disk which is current upon entry into the program. A disk's type is specified by name in the data file. This option can only be used if a disk is being made current as described above.
–`s`	Silent. Suppress all of the standard output. Error messages are still displayed. This is generally used in conjunction with the –`f` option.
–`x` *data-file*	Use the list of disks contained in *data-file*.

USAGE

The `format` utility's main menu items allow you to do the following tasks:

`analyze`	Run read, write, and compare tests.
`backup`	Search for backup labels.
`current`	Display the device name, the disk geometry, and the pathname to the disk device.
`defect`	Retrieve and print defect lists.
`disk`	Choose the disk that will be used in subsequent operations (known as the current disk).
`fdisk`	Run the `fdisk`(1M) program to create a `fdisk` partition for Solaris software (x86 based systems only).
`format`	Format and verify the current disk.

`inquiry`	Display the vendor, product name, and revision level of the current drive.
`label`	Write a new label to the current disk.
`partition`	Create and modify slices.
`quit`	Exit the format menu.
`repair`	Repair a specific block on the disk.
`save`	Save new disk and slice information.
`type`	Select (define) a disk type.
`verify`	Read and display labels. Print information such as the number of cylinders, alternate cylinders, heads, sectors, and the partition table.
`volname`	Label the disk with a new eight character volume name.

ENVIRONMENT VARIABLES

`FORMAT_PATH`	A colon-separated list of filenames and/or directories of disk and partition definitions. If a directory is specified, `format` searches for the file `format.dat` in that directory.

FILES

`/etc/format.dat` Default data file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, the `format` command requires the `sys_devices` privilege and a UID of `0`.

SEE ALSO

SunOS 5.7 Reference Manual

`fmthard`(1M), `prtvtoc`(1M), `format.dat`(4), `attributes`(5), `sd`(7D)

See *Disk Management* in *System Administration Guide, Volume I*

x86 Only

`fdisk`(1M)

WARNINGS

When the `format` function is selected to format the Maxtor 207MB disk, the following message displays:

```
Mode sense page(4) reports rpm value as 0, adjusting it to 3600.
```

This is a drive bug that may also occur with older third-party drives.

The above message is not an error; the drive will still function correctly.

NOTES

`format` provides a help facility you can use whenever `format` is expecting input. You can request help about what information is expected by simply entering a question mark (?) and `format` prints a brief description of what type of input is needed. If you enter a `?` at the menu prompt, a list of available commands is displayed.

For SCSI disks, formatting is done with both Primary and Grown defects list by default. However, if only Primary list is extracted in defect menu before formatting, formatting will be done with Primary list only.

NAME

fsdb_ufs – ufs File System Debugger

SYNOPSIS

fsdb –F ufs [*generic_options*] [*specific_options*] *special*

DESCRIPTION

The `fsdb_ufs` command is an interactive tool that can be used to patch up a damaged UFS file system. It has conversions to translate block and i-numbers into their corresponding disk addresses. Also included are mnemonic offsets to access different parts of an inode. These greatly simplify the process of correcting control block entries or descending the file system tree.

`fsdb` contains several error-checking routines to verify inode and block addresses. These can be disabled if necessary by invoking `fsdb` with the `–o` option or by the use of the `o` command.

`fsdb` reads a block at a time and will therefore work with raw as well as block I/O devices. A buffer management routine is used to retain commonly used blocks of data in order to reduce the number of read system calls. All assignment operations result in an immediate write-through of the corresponding block. Note that in order to modify any portion of the disk, `fsdb` must be invoked with the `w` option.

Wherever possible, `adb`-like syntax was adopted to promote the use of `fsdb` through familiarity.

OPTIONS

The following option is supported:

`–o` Specify UFS file system specific options. These options can be any combination of the following separated by commas (with no intervening spaces). The options available are:

`?`	Display usage
`o`	Override some error conditions
`p='string'`	Set prompt to string
`w`	Open for write

USAGE

Numbers are considered hexadecimal by default. However, the user has control over how data is to be displayed or accepted. The `base` command will display or set the input/output base. Once set, all input will default to this base and all output will be shown in this base. The base can be overridden temporarily for input by preceding hexadecimal numbers with '`0x`', preceding decimal numbers with '`0t`', or octal numbers with '`0`'. Hexadecimal numbers beginning with `a-f` or `A-F` must be preceded with '`0x`' to distinguish them from commands.

Disk addressing by `fsdb` is at the byte level. However, `fsdb` offers many commands to convert a desired inode, directory entry, block, superblock and

so forth to a byte address. Once the address has been calculated, `fsdb` will record the result in dot (`.`).

Several global values are maintained by `fsdb`:

- the current base (referred to as `base`),
- the current address (referred to as `dot`),
- the current inode (referred to as `inode`),
- the current count (referred to as `count`),
- and the current type (referred to as `type`).

Most commands use the preset value of `dot` in their execution. For example,

```
> 2:inode
```

will first set the value of `dot` to 2, ':', will alert the start of a command, and the `inode` command will set `inode` to 2. A count is specified after a ',', Once set, `count` will remain at this value until a new command is encountered which will then reset the value back to 1 (the default). So, if

```
> 2000,400/X
```

is typed, 400 hex longs are listed from 2000, and when completed, the value of `dot` will be `2000 + 400 * sizeof (long)`. If a RETURN is then typed, the output routine will use the current values of `dot`, `count`, and `type` and display 400 more hex longs. A '*' will cause the entire block to be displayed.

End of fragment, block and file are maintained by `fsdb`. When displaying data as fragments or blocks, an error message will be displayed when the end of fragment or block is reached. When displaying data using the `db`, `ib`, `directory`, or `file` commands an error message is displayed if the end of file is reached. This is mainly needed to avoid passing the end of a directory or file and getting unknown and unwanted results.

An example showing several commands and the use of RETURN would be:

```
> 2:ino; 0:dir?d
     or
> 2:ino; 0:db:block?d
```

The two examples are synonymous for getting to the first directory entry of the root of the file system. Once there, any subsequent RETURN (or +, -) will advance to subsequent entries. Note that

```
> 2:inode; :ls
      or
> :ls /
```

is again synonymous.

Expressions

The symbols recognized by `fsdb` are:

RETURN — update the value of `dot` by the current value of `type` and display using the current value of `count`.

— numeric expressions may be composed of +, -, *, and % operators (evaluated left to right) and may use parentheses. Once evaluated, the value of `dot` is updated.

`,` *count* — count indicator. The global value of `count` will be updated to `count`. The value of `count` will remain until a new command is run. A count specifier of '*' will attempt to show a *blocks's* worth of information. The default for `count` is 1.

`?` *f* — display in structured style with format specifier *f*. See `FormattedOutput`.

/ f — display in unstructured style with format specifier *f* See `FormattedOutput`.

`.` — the value of `dot`.

`+`*e* — increment the value of `dot` by the expression *e*. The amount actually incremented is dependent on the size of `type`:

```
dot = dot + e * sizeof (type)
```

The default for *e* is `1`.

`-`*e* — decrement the value of `dot` by the expression *e*. See `+`.

`*`*e* — multiply the value of `dot` by the expression *e*. Multiplication and division don't use `type`. In the above calculation of `dot`, consider the `sizeof(type)` to be `1`.

%e — divide the value of `dot` by the expression *e*. See `*`.

`<` *name* — restore an address saved in register *name*. *name* must be a single letter or digit.

`>` *name* — save an address in register *name*. *name* must be a single letter or digit.

`=` *f* display indicator. If *f* is a legitimate format specifier. then the value of `dot` is displayed using the format specifier *f*. See `Formatted`Output. Otherwise, assignment is assumed See `=`.

`= [`*s*`] [`*e*`]` assignment indicator. The address pointed to by `dot` has its contents changed to the value of the expression *e* or to the ASCII representation of the quoted (") string *s*. This may be useful for changing directory names or ASCII file information.

`=+` *e* incremental assignment. The address pointed to by `dot` has its contents incremented by expression *e*.

`=-` *e* decremental assignment. The address pointed to by `dot` has its contents decremented by expression *e*.

Commands

A command must be prefixed by a ':' character. Only enough letters of the command to uniquely distinguish it are needed. Multiple commands may be entered on one line by separating them by a SPACE, TAB or ';'.

In order to view a potentially unmounted disk in a reasonable manner, `fsdb` offers the `cd`, `pwd`, `ls` and `find` commands. The functionality of these commands substantially matches those of its UNIX counterparts. See individual commands for details. The '*', '?', and '[-]' wild card characters are available.

`base=b` display or set base. As stated above, all input and output is governed by the current `base`. If the `=b` is omitted, the current `base` is displayed. Otherwise, the current `base` is set to *b*. Note that this is interpreted using the old value of `base`, so to ensure correctness use the '0', '0t', or '0x' prefix when changing the `base`. The default for `base` is hexadecimal.

`block` convert the value of `dot` to a block address.

`cd` *dir* change the current directory to directory *dir*. The current values of `inode` and `dot` are also updated. If no *dir* is specified, then change directories to inode `2` ("/").

`cg` convert the value of `dot` to a cylinder group.

`directory` If the current `inode` is a directory, then the value of `dot` is converted to a directory slot offset in that directory and `dot` now points to this entry.

`file` the value of `dot` is taken as a relative block count from the beginning of the file. The value of `dot` is updated to the first byte of this block.

`find` *dir* [`–name` *n*] [`–inum` *i*]	find files by name or i-number. `find` recursively searches directory `dir` and below for filenames whose i-number matches *i* or whose name matches pattern *n*. Note that only one of the two options (-name or -inum) may be used at one time. Also, the -print is not needed or accepted.
`fill=`*p*	fill an area of disk with pattern *p*. The area of disk is delimited by `dot` and `count`.
`fragment`	convert the value of *dot* to a fragment address. The only difference between the `fragment` command and the `block` command is the amount that is able to be displayed.
`inode`	convert the value of *dot* to an inode address. If successful, the current value of `inode` will be updated as well as the value of *dot*. As a convenient shorthand, if ':inode' appears at the beginning of the line, the value of *dot* is set to the current `inode` and that inode is displayed in inode format.
`log_chk`	run through the valid log entries without printing any information and verify the layout.
`log_delta`	count the number of deltas into the log, using the value of dot as an offset into the log. No checking is done to make sure that offset is within the head / tail offsets.
`log_head`	display the header information about the file system logging. This shows the block allocation for the log and the data structures on the disk.
`log_otodb`	return the physical disk block number, using the value of dot as an offset into the log.
`log_show`	display all deltas between the beginning of the log (BOL) and the end of the log (EOL).
`ls`	[`–R`] [`–l`] *pat1 pat2*... list directories or files. If no file is specified, the current directory is assumed. Either or both of the options may be used (but, if used, *must* be specified before the filename specifiers). Also, as stated above, wild card characters are available and multiple arguments may be given. The long listing shows only the i-number and the name; use the `inode` command with '?i' to get more information.
`override`	toggle the value of override. Some error conditions may be overriden if override is toggled on.
`prompt` *p*	change the `fsdb` prompt to *p*. *p* must be surrounded by (")s.

`pwd` display the current working directory.

`quit` quit `fsdb`.

`sb` the value of *dot* is taken as a cylinder group number and then converted to the address of the superblock in that cylinder group. As a shorthand, ':sb' at the beginning of a line will set the value of *dot* to *the* superblock and display it in superblock format.

`shadow` if the current inode is a shadow inode, then the value of *dot* is set to the beginning of the shadow inode data.

`!` escape to shell

Inode Commands

In addition to the above commands, there are several commands that deal with inode fields and operate directly on the current `inode` (they still require the ':'). They may be used to more easily display or change the particular fields. The value of *dot* is only used by the '`:db`' and '`:ib`' commands. Upon completion of the command, the value of *dot* is changed to point to that particular field. For example,

```
> :ln=+1
```

would increment the link count of the current `inode` and set the value of *dot* to the address of the link count field.

`at` access time.

`bs` block size.

`ct` creation time.

`db` use the current value of *dot* as a direct block index, where direct blocks number from 0 - 11. In order to display the block itself, you need to 'pipe' this result into the `block` or `fragment` command. For example,

```
> 1:db:block,20/X
```

would get the contents of data block field 1 from the inode and convert it to a block address. 20 longs are then displayed in hexadecimal. See `Formatted`Output`.`

`gid` group id.

`ib` use the current value of *dot* as an indirect block index where indirect blocks number from 0 - 2. This will only get the indirect block itself (the block containing the pointers to the actual blocks). Use the `file` command and start at block 12 to get to the actual blocks.

`ln` link count.

`mt` modification time.

`md` mode.

`maj` major device number.

`min` minor device number.

`nm` although listed here, this command actually operates on the directory name field. Once poised at the desired directory entry (using the *directory* command), this command will allow you to change or display the directory name. For example,

```
> 7:dir:nm="foo"
```

will get the 7th directory entry of the current `inode` and change its name to foo. Note that names cannot be made larger than the field is set up for. If an attempt is made, the string is truncated to fit and a warning message to this effect is displayed.

`si` shadow inode.

`sz` file size.

`uid` user id.

Formatted Output

There are two styles and many format types. The two styles are structured and unstructured. Structured output is used to display inodes, directories, superblocks and the like. Unstructured displays raw data. The following shows the different ways of displaying:

`?`	`c`	display as cylinder groups
	`i`	display as inodes
	`d`	display as directories
	`s`	display as superblocks
	`S`	display as shadow inode data
`/`	`b`	display as bytes
	`c`	display as characters
	`o O`	display as octal shorts or longs
	`d D`	display as decimal shorts or longs
	`x X`	display as hexadecimal shorts or longs

The format specifier immediately follows the '/' or '?' character. The values displayed by '/b' and all '?' formats are displayed in the current `base`. Also, `type` is appropriately updated upon completion.

EXAMPLES

`> 2000+400%(20+20)=D`
will display `2010` in decimal (use of `fsdb` as a calculator for complex arithmetic).

`> 386:ino?i`
display i-number `386` in an inode format. This now becomes the current `inode`.

`> :ln=4`
changes the link count for the current `inode` to `4`.

`> :ln=+1`
increments the link count by `1`.

`> :ct=X`
display the creation time as a hexadecimal long.

`> :mt=t`
display the modification time in time format.

`> 0:file/c`
displays, in ASCII, block zero of the file associated with the current `inode`.

`> 2:ino,*?d`
displays the first blocks worth of directory entries for the root inode of this file system. It will stop prematurely if the EOF is reached.

`> 5:dir:inode; 0:file,*/c`
changes the current inode to that associated with the `5`th directory entry (numbered from zero) of the current `inode`. The first logical block of the file is then displayed in ASCII.

`> :sb`
displays the superblock of this file system.

`> 1:cg?c`
displays cylinder group information and summary for cylinder group `1`.

`> 2:inode; 7:dir=3`
changes the i-number for the seventh directory slot in the root directory to `3`.

`> 2:db:block,*?d`
displays the third block of the current `inode` as directory entries.

`> 7:dir:nm="name"`
changes the name field in the directory slot to *name*.

`> 3c3:fragment,20:fill=0x20`
get fragment `3c3` and fill `20` `type` elements with `0x20`.

`> 2050=0xffff`
set the contents of address `2050` to `0xffffffff`. `0xffffffff` may be truncated depending on the current `type`.

`> 1c92434="this is some text"`
will place the ASCII for the string at `1c92434`.

`> 2:ino:si:ino;0:shadow,*?S`
displays all of the shadow inode data in the shadow inode associated with the root inode of this file system.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The following are added to handle shadow inodes: the `shadow` command, the `si` inode command, and the `S` format type.

SEE ALSO

SunOS 5.7 Reference Manual

`clri`(1M), `fsck_ufs`(1M), `dir_ufs`(4), `fs_ufs`(4), `attributes`(5)

WARNINGS

Since `fsdb` reads the disk raw, extreme caution is advised in determining its availability of `fsdb` on the system. Suggested permissions are 600 and owned by bin.

NOTES

The old command line syntax for clearing i-nodes using the ufs-specific `'-z i-number'` option is still supported by the new debugger, though it is obsolete and will be removed in a future release. Use of this flag will result in correct operation, but an error message will be printed warning of the impending obsolesence of this option to the command. The equivalent functionality is available using the more flexible `clri`(1M) command.

NAME

in.ftpd, ftpd – File transfer protocol server

SYNOPSIS

in.ftpd [–dl] [–t *timeout*]

DESCRIPTION

in.ftpd is the Internet File Transfer Protocol (FTP) server process. The server is invoked by the Internet daemon, inetd(1M) , each time a connection to the FTP service (see services(4)) is made.

OPTIONS

–d	Debugging information is logged to the system log daemon syslogd(1) .
–l	Each FTP session is logged to the system log daemon syslogd(1) .
–t *timeout*	Set the inactivity timeout period to *timeout* seconds. The FTP server will timeout an inactive session after 15 minutes.

Requests

The FTP server currently supports the following FTP requests; case is not distinguished.

ABOR	abort previous command
ACCT	specify account (ignored)
ALLO	allocate storage (vacuously)
APPE	append to a file
CDUP	change to parent of current working directory
CWD	change working directory
DELE	delete a file
HELP	give help information
LIST	give list files in a directory (ls –lg)
MKD	make a directory
MODE	specify data transfer *mode*
NLST	give name list of files in directory (ls)
NOOP	do nothing
PASS	specify password
PASV	prepare for server-to-server transfer
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session

RETR retrieve a file

RMD remove a directory

RNFR specify rename-from file name

RNTO specify rename-to file name

STOR store a file

STOU store a file with a unique name

STRU specify data transfer *structure*

TYPE specify data transfer type

USER specify user name

XCUP change to parent of current working directory

XCWD change working directory

XMKD make a directory

XPWD print the current working directory

XRMD remove a directory

The remaining FTP requests specified in RFC 959 are recognized, but not implemented.

The FTP server will abort an active file transfer only when the ABOR command is preceded by a Telnet "Interrupt Process" (IP) signal and a Telnet "Synch" signal in the command Telnet stream, as described in RFC 959. in.ftpd interprets file names according to the "globbing" conventions used by sh(1) . This allows users to utilize the metacharacters:

```
* ?  [ ] { } ~
```

in.ftpd 's umask (which it uses to create files during PUT operations) may be adjusted by adding the line

```
UMASK=nnn
```

to /etc/default/ftpd .

The banner returned by in.ftpd in the parenthetical portion of its greeting is configurable. The default is equivalent to "'uname -sr'" and will be used if no banner is set in /etc/default/ftpd . To set the banner, add a line of the form

```
BANNER="..."
```

to /etc/default/ftpd . Nonempty banner strings are fed to shells for evaluation.

The default banner may also be obtained by

```
BANNER="‘uname -s‘ ‘uname -r‘"
```

and no banner will be printed if /etc/default/ftpd contains

```
BANNER="
```

in.ftpd authenticates users according to five rules.

First, the user name must be in the password data base, /etc/passwd , and have a password that is not NULL . A password must always be provided by the client before any file operations may be performed. The PAM framework (see SECURITY below) is used to verify that the correct password was entered.

Second, if the user name appears in the file /etc/ftpusers , ftp access is denied.

Third, ftp access is denied if the user's shell (from /etc/passwd) is not listed in the file /etc/shells . If the file /etc/shells does not exist, then the user's shell must be one of the following:

/usr/bin/sh	/usr/bin/csh	/usr/bin/ksh
/usr/bin/jsh	/bin/sh	/bin/csh
/bin/ksh	/bin/jsh	/sbin/sh
/sbin/jsh		

Fourth, if the user name is "anonymous" or "ftp", an entry for the user name ftp must be present in the password and shadow files. The user is then allowed to log in by specifying any password — by convention this is given as the user's e-mail address (such as user@host.Sun.COM). Do not specify a valid shell in the password entry of the ftp user, and do not give it a valid password (use NP in the encrypted password field of the shadow file).

Fifth, access is denied unless a user has the remote login authorization. If the /etc/nologin file exists, access is denied.

For anonymous ftp users, in.ftpd takes special measures to restrict the client's access privileges. The server performs a chroot(2) command to the home directory of the "ftp" user. In order that system security is not breached, it is recommended that the "ftp" subtree be constructed with care; the following rules are suggested.

~ftp
Make the home directory owned by root and unwritable by anyone.

`~ftp/bin`
Make this directory owned by root and unwritable by anyone. Make this a symbolic link to `~ftp/usr/bin` . The program `ls`(1) must be present to support the list commands. This program should have mode 111.

`~ftp/usr/lib`
Make this directory owned by root and unwritable by anyone. Copy the following shared libraries from `/usr/lib` into this directory:

ld.so.1*
libc.so.1*
libdl.so.1*
libmp.so.2*
libnsl.so.1*
libsocket.so.1*
nss_compat.so.1*
nss_dns.so.1*
nss_files.so.1*
nss_nis.so.1*
nss_nisplus.so.1*
nss_xfn.so.1*
straddr.so*
straddr.so.2*

`~ftp/etc`
Make this directory owned by root and unwritable by anyone. Copies of the files `passwd`(4) , `group`(4) , and `netconfig`(4) must be present for the `ls`(1) command to work properly. These files should be mode 444.

`~ftp/pub`
Make this directory mode 755 and owned by root. Users should then place files which are to be accessible via the anonymous account in this directory.

`~ftp/dev`
Make this directory owned by root and unwritable by anyone. First perform `ls –lL` on the device files listed below to determine their major and minor numbers, then use `mknod` to create them in this directory.

```
/dev/zero/dev/tcp/dev/udp/dev/ticotsord
```

Set the read and write mode on these nodes to 666 so that passive `ftp` will not fail with “permission denied” errors.

`~ftp/usr/share/lib/zoneinfo`
Make this directory mode 555 and owned by root. Copy its contents from `/usr/share/lib/zoneinfo` . This enables `ls –l` to display time and date stamps correctly.

SECURITY

`in.ftpd` uses `pam`(3) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the module to be used for `in.ftpd`. Here is a partial `pam.conf` file with entries for the `in.ftpd` command using the UNIX authentication, account management, and session management module.

ftp	auth	required	/usr/lib/security/pam_unix.so.1
ftp	account	required	/usr/lib/security/pam_unix.so.1
ftp	session	required	/usr/lib/security/pam_unix.so.1

If there are no entries for the ftp service, then the entries for the "other" service will be used. Unlike `login`, `passwd`, and other commands, the ftp protocol will only support a single password. Using multiple modules will prevent `in.ftpd` from working properly.

EXAMPLES

EXAMPLE 1 Setting Up An Anonymous Ftp

To set up anonymous ftp, add the following entry to the `/etc/passwd` file. In this example, `/export/ftp` was chosen to be the anonymous ftp area, and the shell is the non-existent file `/nosuchshell`. This prevents users from logging in as the ftp user.

ftp:x:30000:30000:Anonymous FTP:/export/ftp:/nosuchshell

Add the following entry to the `/etc/shadow` file:

ftp:NP:6445::::::

The following shell script sets up the anonymous ftp area. It presumes that names are resolved using NIS .

```
#!/bin/sh
      # script to setup anonymous ftp area
      #

      # verify you are root
      /usr/bin/id | grep -w 'uid=0' >/dev/null 2>&1
      if [ "$?" != "0" ]; then
         echo
         exit 1
      fi

      # handle the optional command line argument
```

```
case $# in

   # the default location for the anon ftp comes from the passwd
   # file
   0) ftphome="`getent passwd ftp | cut -d: -f6`"
      ;;

   1) if [ "$1" = "start" ]; then
         ftphome="`getent passwd ftp | cut -d: -f6`"
      else
         ftphome=$1
      fi
      ;;

   *) echo "Usage: $0 [anon-ftp-root]"
      exit 1
      ;;
esac

if [ -z "${ftphome}" ]; then
   echo "$0: ftphome must be non-null"
   exit 2
fi

case ${ftphome} in
   /*) # ok
       ;;

   *) echo "$0: ftphome must be an absolute pathname"
      exit 1
      ;;
esac

# This script assumes that ftphome is neither / nor /usr so ...
if [ -z "${ftphome}" -o "${ftphome}" = "/" -o "${ftphome}" = "/usr" ]; then
   echo "$0: ftphome must be non-null and neither / or /usr"
   exit 2
fi

# If ftphome does not exist but parent does, create ftphome
if [ ! -d ${ftphome} ]; then
    # lack of -p below is intentional
    mkdir ${ftphome}
fi
chown root ${ftphome}
chmod 555 ${ftphome}

echo Setting up anonymous ftp area ${ftphome}

# Ensure that the /usr directory exists
if [ ! -d ${ftphome}/usr ]; then
    mkdir -p ${ftphome}/usr
fi
# Now set the ownership and modes to match the man page
chown root ${ftphome}/usr
```

```
      chmod 555 ${ftphome}/usr

      # Ensure that the /usr/bin directory exists
      if [ ! -d ${ftphome}/usr/bin ]; then
          mkdir -p ${ftphome}/usr/bin
      fi
      # Now set the ownership and modes to match the man page
      chown root ${ftphome}/usr/bin
      chmod 555 ${ftphome}/usr/bin

      # this may not be the right thing to do
      # but we need the bin -> usr/bin link
      rm -f ${ftphome}/bin
      ln -s usr/bin ${ftphome}/bin

      # Ensure that the /usr/lib and /etc directories exist
      if [ ! -d ${ftphome}/usr/lib ]; then
          mkdir -p ${ftphome}/usr/lib
      fi
      chown root ${ftphome}/usr/lib
      chmod 555 ${ftphome}/usr/lib

      if [ ! -d ${ftphome}/usr/lib/security ]; then
          mkdir -p ${ftphome}/usr/lib/security
      fi
      chown root ${ftphome}/usr/lib/security
      chmod 555 ${ftphome}/usr/lib/security

      if [ ! -d ${ftphome}/etc ]; then
          mkdir -p ${ftphome}/etc
      fi
      chown root ${ftphome}/etc
      chmod 555 ${ftphome}/etc


      # a list of all the commands that should be copied to
      # ${ftphome}/usr/bin.
      # /usr/bin/ls is needed at a minimum.
      ftpcmd="
         /usr/bin/ls
      "

      # ${ftphome}/usr/lib needs to have all the libraries needed by the above
      # commands, plus the runtime linker, and some name service libraries
      # to resolve names. We just take all of them here.

      ftplib="`ldd $ftpcmd | nawk '$3 ~ /lib/ { print $3 }' | sort | uniq`"
      ftplib="$ftplib /usr/lib/nss_* /usr/lib/straddr* /usr/lib/libmp.so*"
      ftplib="$ftplib /usr/lib/libnsl.so.1 /usr/lib/libsocket.so.1 \\
      /usr/lib/ld.so.1"
      ftplib="`echo $ftplib | tr ' ' '\
' | sort | uniq`"

      cp ${ftplib} ${ftphome}/usr/lib
      chmod 555 ${ftphome}/usr/lib/*
```

```
cp /usr/lib/security/* ${ftphome}/usr/lib/security
chmod 555 ${ftphome}/usr/lib/security/*

cp ${ftpcmd} ${ftphome}/usr/bin
chmod 111 ${ftphome}/usr/bin/*

# you also might want to have separate minimal versions of passwd
# and group
cp /etc/passwd /etc/group /etc/netconfig /etc/pam.conf ${ftphome}/etc
chmod 444 ${ftphome}/etc/*

# need /etc/default/init for timezone to be correct
if [ ! -d ${ftphome}/etc/default ]; then
   mkdir ${ftphome}/etc/default
fi
chown root ${ftphome}/etc/default
chmod 555 ${ftphome}/etc/default
cp /etc/default/init ${ftphome}/etc/default
chmod 444 ${ftphome}/etc/default/init

# Copy timezone database
mkdir -p ${ftphome}/usr/share/lib/zoneinfo
(cd ${ftphome}/usr/share/lib/zoneinfo
  (cd /usr/share/lib/zoneinfo; find . -print |
   cpio -o) 2>/dev/null | cpio -imdu 2>/dev/null
  find . -print | xargs chmod 555
  find . -print | xargs chown root
)

# Ensure that the /dev directory exists
if [ ! -d ${ftphome}/dev ]; then
    mkdir -p ${ftphome}/dev
fi

# make device nodes. ticotsord and udp are necessary for
# 'ls' to resolve NIS names.

for device in zero tcp udp ticotsord ticlts
do
   line=`ls -lL /dev/${device} | sed -e 's/,//'`
   major=`echo $line | awk '{print $5}'`
   minor=`echo $line | awk '{print $6}'`
   rm -f ${ftphome}/dev/${device}
   mknod ${ftphome}/dev/${device} c ${major} ${minor}
done

chmod 666 ${ftphome}/dev/*

## Now set the ownership and modes
chown root ${ftphome}/dev
chmod 555 ${ftphome}/dev

# uncomment the below if you want a place for people to store
# things, but beware the security implications
```

```
#if [ ! -d ${ftphome}/pub ]; then
#   mkdir -p ${ftphome}/pub
#fi
#chown root ${ftphome}/pub
#chmod 1755 ${ftphome}/pub
```

After running this script, edit the files in `~ftp/etc` to make sure all non-public information is removed.

ATTRIBUTES

See `attributes` (5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

FILES

`/etc/default/ftpd`

SUMMARY OF TRUSTED SOLARIS CHANGES

Login is not allowed unless the user has the remote login authorization. If the `/etc/nologin` file exists, the user is not allowed to log in.

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `chroot`(2) , `getsockopt`(3N) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`ftp`(1) , `ls`(1) , `sh`(1) , `aset`(1M) , `mknod`(1M) , `syslogd`(1M) , `group`(4) , `netconfig`(4) , `netrc`(4) , `passwd`(4) , `services`(4) , `attributes`(5)

Postel, Jon, and Joyce Reynolds, *File Transfer Protocol (FTP)* , RFC 959, Network Information Center, SRI International, Menlo Park, CA, October 1985.

DIAGNOSTICS

`in.ftpd` logs various errors to `syslogd` , with a facility code of `daemon` . These messages are logged only if the `–l` flag is specified.

Info Severity

`FTPD: connection from` *host* at *time*
A connection was made to `ftpd` from the host *host* at the date and time *time* .

`FTPD: User` *user* `timed out after` *timeout* `seconds at` *time*
The user *user* was logged out because they had not entered any commands after *timeout* seconds; the logout occurred at the date and time *time* .

Debug Severity

These messages are logged only if the `–d` flag is specified.

`FTPD: command: command`
A command line containing `command` was read from the FTP client.

`lost connection`
The FTP client dropped the connection.

<––– replycode

<—— *replycode*–
A reply was sent to the FTP client with the reply code *replycode* . The next message logged will include the message associated with the reply. If a – follows the reply code, the reply is continued on later lines.

NOTES

The anonymous ftp account is inherently dangerous and should be avoided when possible.

The name service caching daemon `/usr/sbin/nscd` may interfere with some of the functionality of anonymous ftp . The *sublogin* feature does not work unless caching for `passwd` is disabled in `/etc/nscd.conf` .

The server must run as the superuser to create sockets with privileged port numbers. It maintains an effective user ID of the logged in user, reverting to the superuser only when binding addresses to sockets. The possible security holes have been extensively scrutinized, but may be incomplete.

`/etc/ftpusers` contains a list of users who cannot access the system; the format of the file is one user name per line.

NAME

fuser – Identify processes using a file or file structure

SYNOPSIS

/usr/sbin/fuser [– [c | f]ku] *files* [[– [c | f]ku] *files*] …

DESCRIPTION

`fuser` displays the process IDs of the processes that are using the *files* specified as arguments.

Each process ID is followed by a letter code. These letter codes are interpreted as follows: if the process is using the file as

`c` Indicates that the process is using the file as its current directory.

`m` Indicates that the process is using a file mapped with `mmap( )`-2. See `mmap`(2) for details.

`o` Indicates that the process is using the file as an open file.

`r` Indicates that the process is using the file as its root directory.

`t` Indicates that the process is using the file as its text file.

`y` Indicates that the process is using the file as its controlling terminal.

For block special devices with mounted file systems, all processes using any file on that device are listed. For all types of files (text files, executables, directories, devices, and so forth), only the processes using that file are reported.

If more than one group of files are specified, the options may be respecified for each additional group of files. A lone dash cancels the options currently in force.

The process IDs are printed as a single line on the standard output, separated by spaces and terminated with a single new line. All other output is written on standard error.

To succeed, this command requires the `sys_mount` privilege. Any user with permission to read `/dev/kmem` and `/dev/mem` can use fuser.

OPTIONS

The following options are supported:

`–c` Reports on files that are mount points for file systems, and any files within that mounted file system.

`–f` Print a report for the named file, not for files within a mounted file system.

`–k` Sends the `SIGKILL` signal to each process. Since this option spawns kills for each process, the kill messages may not show up immediately. (See `kill`(2)). The `–k` option requires the `sys_mountproc_owner` privilege if it is to terminate another user's process.

`–u` Displays the user login name in parentheses following the process ID.

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `fuser`: `LANG`, `LC_ALL`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command requires the `sys_mount` privilege. With the –`k` option, the command also needs the `proc_owner` privilege to terminate another user's process.

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `kill`(2)

SunOS 5.7 Reference Manual

`ps`(1), `signal`(3C), `attributes`(5)

NOTES

Because `fuser` works with a snapshot of the system image, it may miss processes that begin using a file while `fuser` is running. Also, processes reported as using a file may have stopped using it while `fuser` was running. These factors should discourage the use of the –`k` option.

NAME

getfsattr – Display file system security attributes

SYNOPSIS

getfsattr [–a | –f | –i | –I | –l | –L | –m | –p | –P | –s | –S]
[–F fstype] [–o option] [*pathname*…]

DESCRIPTION

`getfsattr` displays the specified security attributes of the file system on which *pathname* resides. If no option is specified, all the file system security attributes are displayed. If no *pathname* is given, the attributes of all mounted filesystems are displayed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–a	Display the file system access ACL.
–f	Display the file system attribute flags.
–l	Display the filesystem sensitivity level range in short form.
–L	Display the file system sensitivity level range in long form.
–m	Display the file system MLD prefix.
–p	Display the file system allowed privilege set.
–P	Display the file system forced privilege set.
–s	Display the file system sensitivity label in short form.
–S	Display the file system sensitivity label in long form.
–F *fstype*	Restrict the output to file systems of type *fstype*.
–o *option*	Specifies filesystem specific options. See the individual filesystem variants of `getfsattr` for details.

DIAGNOSTICS

`getfsattr` exits with one of the following values:

0	Success
1	Usage error
2	Failure, error message is the system error number from `getfsattr`(2).

NOTES

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as `ADMIN_LOW`.

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]`; however, the IL component is fixed at `ADMIN_LOW`.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW`.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always `ADMIN_LOW`, and cannot be set on any objects.

SEE ALSO

Trusted Solaris 7 Reference Manual

`getfsattr_ufs`(1M), `getfsattr`(2), `vfstab_adjunct`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

getfsattr_ufs – Display file system security attributes

SYNOPSIS

getfsattr –F ufs [*generic_options*] –o {b | c] }{[*mount_point*} { | *device_name*]…}

DESCRIPTION

`getfsattr` displays the security attributes specified by *generic_options* for the file system mounted on *mount_point* or on the device *device_name*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

generic_options — Options supported by the generic `getfsattr` command. See `getfsattr`(1M) for a description of these options.

`–o` — Select the device type to read. A `b` requests use of the block device, while a `c` requests use of the character device.

DIAGNOSTICS

`getfsattr` exits with one of the following values:

`0` Success

`1` Usage error

`2` Unable to access the specified file system

SEE ALSO

Trusted Solaris 7 Reference Manual

`getfsattr`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

halt, poweroff – Stop the processor

SYNOPSIS

/usr/sbin/halt [–lnqy]

/usr/sbin/poweroff [–lnqy]

DESCRIPTION

`halt` and `poweroff` write out any pending information to the disks and then stop the processor. `poweroff` will have the machine remove power, if possible.

`halt` and `poweroff` normally log the system shutdown to the system log daemon, `syslogd`(1M) , and place a shutdown record in the login accounting file `/var/adm/wtmp` . These actions are inhibited if the –n or –q options are present.

OPTIONS

The following options are supported:

–l Suppress sending a message about who executed `halt` to the system log daemon, `syslogd`(1M) .

–n Prevent the `sync`(4) before stopping.

–q Quick halt. No graceful shutdown is attempted.

–y Halt the system, even from a dialup terminal.

FILES

`/var/adm/wtmp` login accounting file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This command requires the `PRIV_SYS_BOOT` privilege and an effective uid of `0` in order to run.

SEE ALSO

Trusted Solaris 7 Reference Manual

`init`(1M) , `reboot`(1M)

SunOS 5.7 Reference Manual

`shutdown`(1M) , `sync`(1M) , `syslogd`(1M) , `attributes`(5)

NOTES

Unlike `shutdown`(1M) and `init`(1M) , `halt` does not execute the `rc0` scripts.

`poweroff` is equivalent to `init 5` .

NAME

hextoalabel – Convert a hexadecimal label to its character-coded equivalent

SYNOPSIS

/usr/sbin/hextoalabel [*hexadecimal_CMW_label*]

/usr/sbin/hextoalabel –c [*hexadecimal_clearance*]

/usr/sbin/hextoalabel –s [*hexadecimal_sensitivity_label*]

DESCRIPTION

`hextoalabel` converts a hexadecimal label of the type specified into its standard formatted character-coded equivalent and writes the result to the standard output file. If no hexadecimal label is specified, one is read from standard input.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

`–c` Identifies the hexadecimal label as a clearance.

`–s` Identifies the hexadecimal label as a sensitivity label.

RETURN VALUES

`hextoalabel( )` returns:

`0` On success.

`–1` On failure, and writes diagnostics to the standard error file.

FILES

`/etc/security/tsol/label_encodings`
The label encodings file contains the classification names, words, constraints, and values for the defined labels of this system.

NOTES

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as `ADMIN_LOW`.

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]`; however, the IL component is fixed at `ADMIN_LOW`.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW`.

- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always `ADMIN_LOW`, and cannot be set on any objects.
- Options related to information labels in the `label_encodings`(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

SEE ALSO

Trusted Solaris 7 Reference Manual

`label_encodings`(4)

Trusted Solaris administrator's document set

SunOS 5.7 Reference Manual

`attributes`(5)

DIAGNOSTICS

`label translation unavailable`

or

hexadecimal_label `not translatable by this process`

The label services are unavailable at this time for one of these reasons: either the label daemon is not running, or the `label_encodings` file is incorrect or unavailable, or this process is not allowed to translate *hexadecimal_label*. The `sys_trans_label` privilege may be used to override this last restriction.

`unable to translate` *hexadecimal_label* `as type specified`

hexadecimal_label does not match the hexadecimal format for the specified type.

NAME

ifconfig – Configure network interface parameters

SYNOPSIS

/sbin/ifconfig *interface* [*address_family*] [*address* [*dest_address*]] [up] [down] [auto-revarp] [netmask *mask*] [broadcast *address*] [metric *n*] [mtu *n*] [trailers | –`trailers`][private | –`private`][arp | –`arp`][plumb] [unplumb]

/usr/sbin/ifconfig *interface* [*address_family*] [*address* [*dest_address*]] [up] [down] [auto-revarp] [netmask *mask*] [broadcast *address*] [metric *n*] [mtu *n*] [trailers | –`trailers`][private | –`private`][arp | –`arp`][plumb] [unplumb]

/sbin/ifconfig *interface* {auto-dhcp | dhcp }[primary] [wait *seconds*] drop | extend | ping | release | start | status

/usr/sbin/ifconfig *interface* {auto-dhcp | dhcp }[primary] [wait *seconds*] drop | extend | ping | release | start | status

DESCRIPTION

The command `ifconfig` is used to assign an address to a network interface or to configure network interface parameters, or both. `ifconfig` must be used at boot time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters. If no option is specified, `ifconfig` displays the current configuration for a network interface. If an address family is specified, `ifconfig` reports only the details specific to that address family. `ifconfig` needs the `sys_net_config` privilege in order to modify the configuration of a network interface. Options appearing within braces ({ }) indicate that one of the options must be specified.

The two versions of `ifconfig`, `/sbin/ifconfig` and `/usr/sbin/ifconfig`, behave differently with respect to name services. The order in which names are looked up by `/sbin/ifconfig` when the system is booting is fixed and cannot be changed. In contrast, changing `/etc/nsswitch.conf` may affect the behavior of `/usr/sbin/ifconfig`. The system administrator may configure the source and lookup order in the tables via the name service switch. See `nsswitch.conf`(4) for more information.

DHCP Configuration

The third and fourth forms of this command are used to control DHCP (Dynamic Host Configuration Protocol) configuring of the interface. DHCP is only available on interfaces whose address family is `inet`. In this mode, `ifconfig` is used to control operation of `dhcpagent`(1M), the DHCP client daemon. Once an interface is placed under DHCP control (by using the `start` operand), `ifconfig` should not, in normal operation, be used to modify the address or characteristics of the interface. If the address of an interface under DHCP is changed, the agent will implicitly drop the interface from its control, although

this will not occur until dhcpagent wakes up to conduct another DHCP operation on the interface.

OPTIONS

Options that need to open network devices readable only by root and protected at ADMIN_HIGH (ether, auto-revarp, and plumb) are intended to be invoked at ADMIN_HIGH with effective user ID 0. These restrictions may be overridden by the file_dac_read, file_dac_write, and file_mac_read privileges.

The following options are supported:

arp — Enable the use of the Address Resolution Protocol (ARP) in mapping between network level addresses and link level addresses (default). This is currently implemented for mapping between TCP/IP addresses and 10Mb/s Ethernet addresses.

–arp — Disable the use of the Address Resolution Protocol ARP.

auto-dhcp — Use the Dynamic Host Configuration Protocol (DHCP) to automatically acquire an address for this interface. This option has a completely equivalent alias called dhcp.

primary — Defines the interface as the "primary". The interface is defined as the preferred one for the delivery of client-wide configuration data. See dhcpagent(1M) and dhcpinfo(1) for details. Only one interface can be the primary at any given time. If another interface is subsequently selected as the primary, it replaces the previous one. Nominating an interface as the primary one will not have much significance once the client work station has booted, as many applications will already have started and been configured with data read from the previous primary interface.

	`wait` *seconds*	`ifconfig` will wait until the operation either completes or for the interval specified, whichever is the sooner. If no wait interval is given, and the operation is one that cannot complete immediately, `ifconfig` will exit immediately but the requested operation will continue. The exit status of `ifconfig` in this case will indicate merely the validity of the request, not whether that request was actually successful. The symbolic value `forever` may be used in place of a numeric, with obvious meaning.
	`drop`	The specified interface will be removed from the control of `dhcpagent`.
	`extend`	`extend dhcpagent` will try to extend the lease on the interface's IP address. This is not required, as the agent will automatically extend the lease well before it expires.
	`ping`	Checks whether the interface given is under DHCP control. An exit status of `0` means yes.
	`release`	The IP address on the interface is relinquished, and the interface marked as "down".
	`start`	DHCP will be started on the interface.
	`status`	Display the DHCP configuration status of the interface.
`auto-revarp`	Use the Reverse Address Resolution Protocol (RARP) to automatically acquire an address for this interface.	
broadcast *address*	(`inet` only.) Specify the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all 1's. A "+" (plus sign) given for the broadcast value causes the broadcast address to be reset to a default appropriate for the (possibly new) address and netmask. `Note`: The arguments of `ifconfig` are interpreted left to right. Therefore	

```
ifconfig -a netmask + broadcast +
```

and

```
ifconfig -a broadcast + netmask +
```

may result in different values being assigned for the broadcast addresses of the interfaces.

`dhcp` This option is an alias for option `auto-dhcp`

`down` Mark an interface "down". When an interface is marked "down", the system does not attempt to transmit messages through that interface. If possible, the interface is reset to disable reception as well. This action does not automatically disable routes using the interface.

`metric` *n* Set the routing metric of the interface to *n*. If no value is specified, the default is `0`. The routing metric is used by the routing protocol. Higher metrics have the effect of making a route less favorable; metrics are counted as addition hops to the destination network or host.

`mtu` *n* Set the maximum transmission unit of the interface to *n*. For many types of networks, the mtu has an upper limit, for example, `1500` for Ethernet.

`netmask` *mask* (`inet` only.) Specify how much of the address to reserve for subdividing networks into sub-networks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask contains 1's for the bit positions in the 32-bit address which are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion. The mask can be specified in one of four ways:

1. with a single hexadecimal number with a leading 0x,
2. with a dot-notation address,
3. with a "+" (plus sign) address, or
4. with a pseudo host name/pseudo network name found in the network database `networks`(4).

If a "+" (plus sign) is given for the netmask value, the mask is looked up in the `netmasks`(4) database. This lookup finds the longest matching netmask in the database by starting with the interface's IP address as the key and iteratively masking off more and more low order bits of the address. This iterative lookup ensures that the `netmasks`(4) database can be used to specify the netmasks when variable length subnetmasks are used within a network number.

If a pseudo host name/pseudo network name is supplied as the netmask value, netmask data may be located in the `hosts` or `networks` database. Names are looked up by first using `gethostbyname`(3N). If not found there, the names are looked up in `getnetbyname`(3N). These interfaces may in turn use `nsswitch.conf`(4) to determine what data store(s) to use to fetch the actual value.

`plumb` Open the device associated with the physical interface name and set up the streams needed for TCP/IP to use the device. Before this is done, the interface will not show up in the output of `ifconfig –a`.

`unplumb` Destroy any streams associated with this device and close the device. After this command is executed, the device name should not show up in the output of `ifconfig –a`.

`private` Tells the `in.routed` routing daemon that the interface should not be advertised.

`–private` Specify unadvertised interfaces.

`trailers` This flag previously caused a non-standard encapsulation of `inet` packets on certain link levels. Drivers supplied with this release no longer use this flag. It is provided for compatibility, but is ignored.

`–trailers` Disable the use of a "trailer" link level encapsulation.

`up` Mark an interface "up". This happens automatically when setting the first address on an interface. The `up` option enables an interface after an `ifconfig down`, which reinitializes the hardware.

OPERANDS

The *interface* operand, as well as address parameters that affect it, are described below.

interface A string of the form, *name physical-unit*, for example, `le0` or `ie1`; or of the form *name physical-unit*`:`*logical-unit*, for example, `le0:1`. Five special interface names, `–a`, `–ad`, `–au`,

–adD, and –auD, are reserved and refer to all or a subset of the interfaces in the system. If one of these interface names is given, the commands following it are applied to all of the interfaces that match:

–a Apply the commands to all interfaces in the system.

–ad Apply the commands to all "down" interfaces in the system.

–adD Like –ad, but only apply the commands if the interface is not under DHCP (Dynamic Host Configuration Protocol) control.

–au Apply the commands to all "up" interfaces in the system.

–auD Like –au, but only apply the commands if the interface is not under DHCP control.

address_family Since an interface may receive transmissions in differing protocols, each of which may require separate naming schemes, the parameters and addresses are interpreted according to the rules of some address family, specified by the *address_family* parameter. The address families currently supported are `ether` and `inet`. If no address family is specified, `inet` is assumed.

address For the TCP/IP family (`inet`), the *address* is either a host name present in the host name data base [see `hosts`(4)] or in the Network Information Service (NIS) map `hosts`, or a TCP/IP address expressed in the Internet standard "dot notation". Typically, an Internet address specified in dot notation consists of your system's network number and the machine's unique host number. A typical Internet address is `192.9.200.44`, where `192.9.200` is the network number and `44` is the machine's host number.

For the `ether` address family, the address is an Ethernet address represented as *x*:*x*:*x*: *x*:*x*:*x* where *x* is a hexadecimal number between `0` and `FF`.

Some, though not all, of the Ethernet interface cards have their own addresses. To use cards that do not have their own addresses, refer to section 3.2.3(4) of the IEEE 802.3 specification for a definition of the locally administered address space. The use of interface groups should be

restricted to those cards with their own addresses (see INTERFACE GROUPS).

dest_address If the *dest_address* parameter is supplied in addition to the *address* parameter, it specifies the address of the correspondent on the other end of a point-to-point link.

LOGICAL INTERFACES

Solaris TCP/IP allows multiple logical interfaces to be associated with a physical network interface. This allows a single machine to be assigned multiple IP addresses, even though it may have only one network interface. Physical network interfaces have names of the form *driver-name physical-unit-number*, while logical interfaces have names of the form *driver-name physical-unit-number*:*logical-unit-number*. A physical interface is configured into the system using the `plumb` sub-command. For example:

```
ifconfig le0 plumb
```

Logical interfaces do not need to be "plumbed". Once a physical interface has been "plumbed", logical interfaces associated with the physical interface can be configured by naming them in subsequent `ifconfig` commands. However, only root can create or delete a logical interface. For example, when executed by root the command:

```
ifconfig le0:1
```

allocates a logical interface associated with the physical interface `le0` and reports its status. When executed by a non-privileged user, `ifconfig` will report the status of the interface if it already exists, or give an error message if it does not exist.

A logical interface can be configured with parameters (`address`, `netmask`, and so on) different from the physical interface with which it is associated. Logical interfaces that are associated with the same physical interface can be given different parameters as well. Each logical interface must be associated with a physical interface. So, for example, the logical interface `le0:1` can only be configured after the physical interface `le0` has been plumbed.

To delete a logical interface, simply name the interface specifying an address of `0`, after ensuring that the interface has been marked as "down". For example, the command:

```
ifconfig le0:1 0 down
```

will delete the logical interface `le0:1`.

INTERFACE GROUPS

If an interface (logical or physical) shares an IP prefix with another interface, these interfaces are collected into an *interface group*. IP uses an interface group to rotate source address selection when the source address is unspecified, and in the case of multiple physical interfaces in the same group, to scatter traffic across different IP addresses on a per-IP-destination basis. See `netstat`(1M) for per-IP-destination information.

This feature may be disabled by using `ndd`(1M).

EXAMPLES

EXAMPLE 1 Using the `ifconfig` Command

If your workstation is not attached to an Ethernet, the `le0` interface should be marked "down" as follows:

```
example% ifconfig le0 down
```

To print out the addressing information for each interface, use the following command:

```
example% ifconfig a
```

To reset each interface's broadcast address after the netmasks have been correctly set, use the next command:

```
example% ifconfig a broadcast +
```

To change the Ethernet address for interface `le0`, use the following command:

```
example% ifconfig le0 ether aa:1:2:3:4:5
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The `ifconfig` command needs the `sys_net_config` privilege to configure network interfaces; without privilege, `ifconfig` displays the status of network interfaces. The `ether`, `auto-revarp`, and `plumb` options need to open `ADMIN_HIGH` network devices readable only by root; these options are intended to be invoked at `ADMIN_HIGH` with an effective user ID `0`. Alternately, `file_dac_read`, `file_dac_write`, and `file_mac_read` privileges may be used to override these restrictions.

FILES

`/etc/netmasks` netmask data

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

/usr/sbin

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

/sbin

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

Trusted Solaris 7 Reference Manual

`in.routed`(1M), `netstat`(1M), `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`ethers`(3N), `hosts`(4), `netmasks`(4), `networks`(4), `attributes`(5), `arp`(7P)

DIAGNOSTICS

`ifconfig` sends messages that indicate if:

- the specified interface does not exist
- the requested address is unknown
- the user is not privileged and tried to alter an interface's configuration

NOTES

It is recommended that the names `broadcast`, `down`, `private`, `trailers`, `up`, and the other possible option names not be selected when choosing host names. Choosing any one of these names as host names will cause bizarre problems that can be extremely difficult to diagnose.

NAME

inetd – Internet services daemon

SYNOPSIS

inetd [–d] [–s] [–t] [–r *count interval*] [*configuration-file*]

DESCRIPTION

`inetd` is the server process for the Internet standard services. It is usually started up at system boot time. The *configuration-file* lists the services that `inetd` is to provide. If no *configuration-file* is given on the command line, `inetd` reads its configuration information from the file `/etc/inetd.conf`. See `inetd.conf`(4) for more information on the format of this file. `inetd` listens for service requests on the TCP or UDP ports associated with each of the service listed in the configuration file. When a request arrives, `inetd` executes the server program associated with the service.

A service can be configured to be "single-threaded", in which case `inetd` waits for the server process to exit before starting a second server process. RPC services can also be started by `inetd`.

`inetd` provides a number of simple Internet services internally. These include `echo`, `discard`, `chargen` (character generator), `daytime` (human-readable time), and `time` (machine-readable time, in the form of the number of seconds since midnight, January 1, 1900).

`inetd` rereads its configuration file once when it is started and again whenever it receives a hangup signal, `SIGHUP`. New services can be activated, and existing services deleted or modified by editing the configuration file, then sending `inetd` a `SIGHUP` signal.

OPTIONS

–d Runs `inetd` in the foreground and enables debugging output.

–s Allows you to run `inetd` "standalone," outside the Service Access Facility (SAF). If the –s option is omitted, `inetd` will attempt to contact the service access controller (SAC) and will exit if SAC is not already running. See `sac`(1M).

–t Instructs `inetd` to trace the incoming connections for all of its TCP services. It does this by logging the client's IP address and TCP port number, along with the name of the service, using the `syslog`(3) facility. UDP services can not be traced. When tracing is enabled, `inetd` uses the syslog facility code "daemon" and "notice" priority level.

–r Allows `inetd` to detect and then suspend "broken" connectionless datagram services servers, for example, UDP, and RPC/CLTS. Without this detection, a buggy server that fails before consuming the service request will be continuously restarted and will tax system resources too much. The –r flag has the form:

–r *count interval*

count and *interval* are decimal numbers that represent the maximum *count* of invocations per *interval* of seconds a service may be started before the service is considered "broken."

Once considered "broken," a server is suspended for ten minutes. After ten minutes, `inetd` again enables service, hoping the server operates correctly.

If the `–r` flag is not specified, `inetd` behaves as though `–r40 60` was specified.

OPERANDS

configuration-file Lists the services `inetd` is to provide.

EXIT STATUS

`inetd` does not return an Exit Status.

SUMMARY OF TRUSTED SOLARIS CHANGES

`inetd` starts servers at the correct sensitivity label based upon the sensitivity label of the client request.

A number of new configuration options are defined in `inetd.conf`(4). See that man page for more detail.

`inetd` registers RPC servers as multilevel servers with `rpcbind`.

If there is an entry for a server in the `inetd` profile and that entry specifies privileges, the server will inherit the specified privileges from `inetd`. To support this inheritance, `inetd` must have all privileges.

If there is an entry for a server in the `inetd` profile entry and that entry specifies minimum and maximum sensitivity labels, `inetd` will verify that the sensitivity label of the client is within the specified min / max range. If the label is not, the server will not be executed.

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as ADMIN_LOW.

Objects still have CMW labels, and CMW labels still include the IL component: IL[SL]; however, the IL component is fixed at ADMIN_LOW.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return ADMIN_LOW.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always ADMIN_LOW, and cannot be set on any objects.
- Options related to information labels in the label_encodings(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

in.ftpd(1M), in.rexecd(1M), in.rshd(1M), in.tftpd(1M), inetd.conf(4)

SunOS 5.7 Reference Manual

sac(1M), attributes(5)

Postel, Jon, "Echo Protocol," RFC 862, Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon, "Discard Protocol," RFC 863, Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon, "Character Generator Protocol," RFC 864, Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon, "Daytime Protocol," RFC 867, Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon, and Ken Harrenstien, “Time Protocol,” RFC 868, Network Information Center, SRI International, Menlo Park, CA, May 1983.

WARNINGS

Do not configure `udp` services as `nowait`. This will cause a race condition where the `inetd` program selects on the socket and the server program reads from the socket. Many server programs will be forked and performance will be severely compromised.

NOTES

For RPC services, `inetd` listens on all the transports (not only `tcp` and `udp`) as specified for each service in the `inetd.conf`(4) file.

NAME

in.ftpd, ftpd – File transfer protocol server

SYNOPSIS

in.ftpd [–dl] [–t *timeout*]

DESCRIPTION

in.ftpd is the Internet File Transfer Protocol (FTP) server process. The server is invoked by the Internet daemon, inetd(1M) , each time a connection to the FTP service (see services(4)) is made.

OPTIONS

–d	Debugging information is logged to the system log daemon syslogd(1) .
–l	Each FTP session is logged to the system log daemon syslogd(1) .
–t *timeout*	Set the inactivity timeout period to *timeout* seconds. The FTP server will timeout an inactive session after 15 minutes.

Requests

The FTP server currently supports the following FTP requests; case is not distinguished.

ABOR	abort previous command
ACCT	specify account (ignored)
ALLO	allocate storage (vacuously)
APPE	append to a file
CDUP	change to parent of current working directory
CWD	change working directory
DELE	delete a file
HELP	give help information
LIST	give list files in a directory (ls –lg)
MKD	make a directory
MODE	specify data transfer *mode*
NLST	give name list of files in directory (ls)
NOOP	do nothing
PASS	specify password
PASV	prepare for server-to-server transfer
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session

RETR retrieve a file

RMD remove a directory

RNFR specify rename-from file name

RNTO specify rename-to file name

STOR store a file

STOU store a file with a unique name

STRU specify data transfer *structure*

TYPE specify data transfer type

USER specify user name

XCUP change to parent of current working directory

XCWD change working directory

XMKD make a directory

XPWD print the current working directory

XRMD remove a directory

The remaining FTP requests specified in RFC 959 are recognized, but not implemented.

The FTP server will abort an active file transfer only when the ABOR command is preceded by a Telnet "Interrupt Process" (IP) signal and a Telnet "Synch" signal in the command Telnet stream, as described in RFC 959. in.ftpd interprets file names according to the "globbing" conventions used by sh(1) . This allows users to utilize the metacharacters:

```
* ?  [ ] { } ~
```

in.ftpd 's umask (which it uses to create files during PUT operations) may be adjusted by adding the line

```
UMASK=nnn
```

to /etc/default/ftpd .

The banner returned by in.ftpd in the parenthetical portion of its greeting is configurable. The default is equivalent to "`uname -sr`" and will be used if no banner is set in /etc/default/ftpd . To set the banner, add a line of the form

```
BANNER="..."
```

to /etc/default/ftpd . Nonempty banner strings are fed to shells for evaluation.

The default banner may also be obtained by

```
BANNER="‘uname -s‘ ‘uname -r‘"
```

and no banner will be printed if /etc/default/ftpd contains

```
BANNER="
```

in.ftpd authenticates users according to five rules.

First, the user name must be in the password data base, /etc/passwd , and have a password that is not NULL . A password must always be provided by the client before any file operations may be performed. The PAM framework (see SECURITY below) is used to verify that the correct password was entered.

Second, if the user name appears in the file /etc/ftpusers , ftp access is denied.

Third, ftp access is denied if the user's shell (from /etc/passwd) is not listed in the file /etc/shells . If the file /etc/shells does not exist, then the user's shell must be one of the following:

/usr/bin/sh	/usr/bin/csh	/usr/bin/ksh
/usr/bin/jsh	/bin/sh	/bin/csh
/bin/ksh	/bin/jsh	/sbin/sh
/sbin/jsh		

Fourth, if the user name is "anonymous" or "ftp", an entry for the user name ftp must be present in the password and shadow files. The user is then allowed to log in by specifying any password — by convention this is given as the user's e-mail address (such as user@host.Sun.COM). Do not specify a valid shell in the password entry of the ftp user, and do not give it a valid password (use NP in the encrypted password field of the shadow file).

Fifth, access is denied unless a user has the remote login authorization. If the /etc/nologin file exists, access is denied.

For anonymous ftp users, in.ftpd takes special measures to restrict the client's access privileges. The server performs a chroot(2) command to the home directory of the "ftp" user. In order that system security is not breached, it is recommended that the "ftp" subtree be constructed with care; the following rules are suggested.

~ftp
Make the home directory owned by root and unwritable by anyone.

`~ftp/bin`
Make this directory owned by root and unwritable by anyone. Make this a symbolic link to `~ftp/usr/bin` . The program `ls`(1) must be present to support the list commands. This program should have mode 111.

`~ftp/usr/lib`
Make this directory owned by root and unwritable by anyone. Copy the following shared libraries from `/usr/lib` into this directory:

ld.so.1*
libc.so.1*
libdl.so.1*
libmp.so.2*
libnsl.so.1*
libsocket.so.1*
nss_compat.so.1*
nss_dns.so.1*
nss_files.so.1*
nss_nis.so.1*
nss_nisplus.so.1*
nss_xfn.so.1*
straddr.so*
straddr.so.2*

`~ftp/etc`
Make this directory owned by root and unwritable by anyone. Copies of the files `passwd`(4) , `group`(4) , and `netconfig`(4) must be present for the `ls`(1) command to work properly. These files should be mode 444.

`~ftp/pub`
Make this directory mode 755 and owned by root. Users should then place files which are to be accessible via the anonymous account in this directory.

`~ftp/dev`
Make this directory owned by root and unwritable by anyone. First perform `ls –lL` on the device files listed below to determine their major and minor numbers, then use `mknod` to create them in this directory.

`/dev/zero/dev/tcp/dev/udp/dev/ticotsord`

Set the read and write mode on these nodes to 666 so that passive `ftp` will not fail with “permission denied” errors.

`~ftp/usr/share/lib/zoneinfo`
Make this directory mode 555 and owned by root. Copy its contents from `/usr/share/lib/zoneinfo` . This enables `ls –l` to display time and date stamps correctly.

SECURITY

`in.ftpd` uses `pam`(3) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the module to be used for `in.ftpd`. Here is a partial `pam.conf` file with entries for the `in.ftpd` command using the UNIX authentication, account management, and session management module.

ftp	auth	required	/usr/lib/security/pam_unix.so.1
ftp	account	required	/usr/lib/security/pam_unix.so.1
ftp	session	required	/usr/lib/security/pam_unix.so.1

If there are no entries for the ftp service, then the entries for the "other" service will be used. Unlike `login`, `passwd`, and other commands, the ftp protocol will only support a single password. Using multiple modules will prevent `in.ftpd` from working properly.

EXAMPLES

EXAMPLE 1 Setting Up An Anonymous Ftp

To set up anonymous ftp, add the following entry to the `/etc/passwd` file. In this example, `/export/ftp` was chosen to be the anonymous ftp area, and the shell is the non-existent file `/nosuchshell`. This prevents users from logging in as the ftp user.

ftp:x:30000:30000:Anonymous FTP:/export/ftp:/nosuchshell

Add the following entry to the `/etc/shadow` file:

ftp:NP:6445::::::

The following shell script sets up the anonymous ftp area. It presumes that names are resolved using NIS.

```
#!/bin/sh
      # script to setup anonymous ftp area
      #

      # verify you are root
      /usr/bin/id | grep -w 'uid=0' >/dev/null 2>&1
      if [ "$?" != "0" ]; then
         echo
         exit 1
      fi

      # handle the optional command line argument
```

```
case $# in

   # the default location for the anon ftp comes from the passwd
   # file
   0) ftphome="`getent passwd ftp | cut -d: -f6`"
      ;;

   1) if [ "$1" = "start" ]; then
         ftphome="`getent passwd ftp | cut -d: -f6`"
      else
         ftphome=$1
      fi
      ;;

   *) echo "Usage: $0 [anon-ftp-root]"
      exit 1
      ;;
esac

if [ -z "${ftphome}" ]; then
   echo "$0: ftphome must be non-null"
   exit 2
fi

case ${ftphome} in
   /*) # ok
       ;;

   *) echo "$0: ftphome must be an absolute pathname"
      exit 1
      ;;
esac

# This script assumes that ftphome is neither / nor /usr so ...
if [ -z "${ftphome}" -o "${ftphome}" = "/" -o "${ftphome}" = "/usr" ]; then
   echo "$0: ftphome must be non-null and neither / or /usr"
   exit 2
fi

# If ftphome does not exist but parent does, create ftphome
if [ ! -d ${ftphome} ]; then
    # lack of -p below is intentional
    mkdir ${ftphome}
fi
chown root ${ftphome}
chmod 555 ${ftphome}

echo Setting up anonymous ftp area ${ftphome}

# Ensure that the /usr directory exists
if [ ! -d ${ftphome}/usr ]; then
    mkdir -p ${ftphome}/usr
fi
# Now set the ownership and modes to match the man page
chown root ${ftphome}/usr
```

```
        chmod 555 ${ftphome}/usr

        # Ensure that the /usr/bin directory exists
        if [ ! -d ${ftphome}/usr/bin ]; then
            mkdir -p ${ftphome}/usr/bin
        fi
        # Now set the ownership and modes to match the man page
        chown root ${ftphome}/usr/bin
        chmod 555 ${ftphome}/usr/bin

        # this may not be the right thing to do
        # but we need the bin -> usr/bin link
        rm -f ${ftphome}/bin
        ln -s usr/bin ${ftphome}/bin

        # Ensure that the /usr/lib and /etc directories exist
        if [ ! -d ${ftphome}/usr/lib ]; then
            mkdir -p ${ftphome}/usr/lib
        fi
        chown root ${ftphome}/usr/lib
        chmod 555 ${ftphome}/usr/lib

        if [ ! -d ${ftphome}/usr/lib/security ]; then
            mkdir -p ${ftphome}/usr/lib/security
        fi
        chown root ${ftphome}/usr/lib/security
        chmod 555 ${ftphome}/usr/lib/security

        if [ ! -d ${ftphome}/etc ]; then
            mkdir -p ${ftphome}/etc
        fi
        chown root ${ftphome}/etc
        chmod 555 ${ftphome}/etc


        # a list of all the commands that should be copied to
        # ${ftphome}/usr/bin.
        # /usr/bin/ls is needed at a minimum.
        ftpcmd="
           /usr/bin/ls
        "

        # ${ftphome}/usr/lib needs to have all the libraries needed by the above
        # commands, plus the runtime linker, and some name service libraries
        # to resolve names. We just take all of them here.

        ftplib="`ldd $ftpcmd | nawk '$3 ~ /lib/ { print $3 }' | sort | uniq`"
        ftplib="$ftplib /usr/lib/nss_* /usr/lib/straddr* /usr/lib/libmp.so*"
        ftplib="$ftplib /usr/lib/libnsl.so.1 /usr/lib/libsocket.so.1 \\
        /usr/lib/ld.so.1"
        ftplib="`echo $ftplib | tr ' ' '\
    ' | sort | uniq`"

        cp ${ftplib} ${ftphome}/usr/lib
        chmod 555 ${ftphome}/usr/lib/*
```

```
cp /usr/lib/security/* ${ftphome}/usr/lib/security
chmod 555 ${ftphome}/usr/lib/security/*

cp ${ftpcmd} ${ftphome}/usr/bin
chmod 111 ${ftphome}/usr/bin/*

# you also might want to have separate minimal versions of passwd
# and group
cp /etc/passwd /etc/group /etc/netconfig /etc/pam.conf ${ftphome}/etc
chmod 444 ${ftphome}/etc/*

# need /etc/default/init for timezone to be correct
if [ ! -d ${ftphome}/etc/default ]; then
   mkdir ${ftphome}/etc/default
fi
chown root ${ftphome}/etc/default
chmod 555 ${ftphome}/etc/default
cp /etc/default/init ${ftphome}/etc/default
chmod 444 ${ftphome}/etc/default/init

# Copy timezone database
mkdir -p ${ftphome}/usr/share/lib/zoneinfo
(cd ${ftphome}/usr/share/lib/zoneinfo
  (cd /usr/share/lib/zoneinfo; find . -print |
   cpio -o) 2>/dev/null | cpio -imdu 2>/dev/null
  find . -print | xargs chmod 555
  find . -print | xargs chown root
)

# Ensure that the /dev directory exists
if [ ! -d ${ftphome}/dev ]; then
    mkdir -p ${ftphome}/dev
fi

# make device nodes. ticotsord and udp are necessary for
# 'ls' to resolve NIS names.

for device in zero tcp udp ticotsord ticlts
do
   line=`ls -lL /dev/${device} | sed -e 's/,//'`
   major=`echo $line | awk '{print $5}'`
   minor=`echo $line | awk '{print $6}'`
   rm -f ${ftphome}/dev/${device}
   mknod ${ftphome}/dev/${device} c ${major} ${minor}
done

chmod 666 ${ftphome}/dev/*

## Now set the ownership and modes
chown root ${ftphome}/dev
chmod 555 ${ftphome}/dev

# uncomment the below if you want a place for people to store
# things, but beware the security implications
```

```
#if [ ! -d ${ftphome}/pub ]; then
#   mkdir -p ${ftphome}/pub
#fi
#chown root ${ftphome}/pub
#chmod 1755 ${ftphome}/pub
```

After running this script, edit the files in `~ftp/etc` to make sure all non-public information is removed.

ATTRIBUTES

See `attributes` (5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

FILES

`/etc/default/ftpd`

SUMMARY OF TRUSTED SOLARIS CHANGES

Login is not allowed unless the user has the remote login authorization. If the `/etc/nologin` file exists, the user is not allowed to log in.

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `chroot`(2) , `getsockopt`(3N) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`ftp`(1) , `ls`(1) , `sh`(1) , `aset`(1M) , `mknod`(1M) , `syslogd`(1M) , `group`(4) , `netconfig`(4) , `netrc`(4) , `passwd`(4) , `services`(4) , `attributes`(5)

Postel, Jon, and Joyce Reynolds, *File Transfer Protocol (FTP)* , RFC 959, Network Information Center, SRI International, Menlo Park, CA, October 1985.

DIAGNOSTICS

`in.ftpd` logs various errors to `syslogd` , with a facility code of `daemon` . These messages are logged only if the –`l` flag is specified.

Info Severity

`FTPD: connection from` *host* at *time*
A connection was made to `ftpd` from the host *host* at the date and time *time* .

`FTPD: User` *user* `timed out after` *timeout* `seconds at` *time*
The user *user* was logged out because they had not entered any commands after *timeout* seconds; the logout occurred at the date and time *time* .

Debug Severity

These messages are logged only if the –`d` flag is specified.

`FTPD: command: command`
A command line containing `command` was read from the FTP client.

`lost connection`
The FTP client dropped the connection.

<––– *replycode*

<—– *replycode*–
A reply was sent to the FTP client with the reply code *replycode* . The next message logged will include the message associated with the reply. If a – follows the reply code, the reply is continued on later lines.

NOTES

The anonymous ftp account is inherently dangerous and should be avoided when possible.

The name service caching daemon `/usr/sbin/nscd` may interfere with some of the functionality of anonymous ftp . The *sublogin* feature does not work unless caching for `passwd` is disabled in `/etc/nscd.conf` .

The server must run as the superuser to create sockets with privileged port numbers. It maintains an effective user ID of the logged in user, reverting to the superuser only when binding addresses to sockets. The possible security holes have been extensively scrutinized, but may be incomplete.

`/etc/ftpusers` contains a list of users who cannot access the system; the format of the file is one user name per line.

NAME

init, telinit – Process control initialization

SYNOPSIS

/sbin/init [0123456abcQqSs]

/etc/telinit [0123456abcQqSs]

DESCRIPTION

`init` is a general process spawner. Its primary role is to create processes from information stored in the file `/etc/inittab` .

Run Level Defined

At any given time, the system is in one of eight possible run levels. A run level is a software configuration under which only a selected group of processes exists. Processes spawned by `init` for each of these run levels are defined in `/etc/inittab` . `init` can be in one of eight run levels, `0`–`6` and `S` or `s` (`S` and `s` are identical). The run level changes when a privileged user runs `/sbin/init` . This sends appropriate signals to the original `init` spawned by the operating system at boot time, saying which run level to invoke.

init and System Booting

When the system is booted, `init` is invoked and the following occurs. First, it reads `/etc/default/init` to set environment variables. This is typically where `TZ` (time zone) and locale-related environments such as `LANG` or `LC_CTYPE` get set.

`init` then looks in `/etc/inittab` for the `initdefault` entry [see `inittab`(4)]. If the `initdefault` entry:

exists — `init` usually uses the run level specified in that entry as the initial run level to enter.

does not exist — `/etc/inittab` , `init` asks the user to enter a run level from the system console.

`S` or `s` — `init` goes to the single-user state. In this state, the system console device (`/dev/console`) is opened for reading and writing and the command `/sbin/su` , [see `su`(1M)], is invoked. Use either `init` or `telinit` to change the run level of the system. Note that if the shell is terminated (using an end-of-file), `init` only re-initializes to the single-user state if `/etc/inittab` does not exist.

`0-6` — `init` enters the corresponding run level. Run levels `0` , `5` , and `6` are reserved states for shutting the system down. Run levels `2` , `3` , and `4` are available as multi-user operating states.

If this is the first time since power up that `init` has entered a run level other than single-user state, `init` first scans `/etc/inittab` for `boot` and `bootwait` entries [see `inittab`(4)]. These entries are performed before any other processing of `/etc/inittab` takes place, providing that the run level entered

matches that of the entry. In this way any special initialization of the operating system, such as mounting file systems, can take place before users are allowed onto the system. `init` then scans `/etc/inittab` and executes all other entries that are to be processed for that run level.

To spawn each process in `/etc/inittab`, `init` reads each entry and for each entry that should be respawned, it forks a child process. After it has spawned all of the processes specified by `/etc/inittab`, `init` waits for one of its descendant processes to die, a `powerfail` signal, or a signal from another `init` or `telinit` process to change the system's run level. When one of these conditions occurs, `init` re-examines `/etc/inittab`.

inittab Additions

New entries can be added to `/etc/inittab` at any time; however, `init` still waits for one of the above three conditions to occur before re-examining `/etc/inittab`. To get around this, `init Q` or `init q` command wakes `init` to re-examine `/etc/inittab` immediately.

When `init` comes up at boot time and whenever the system changes from the single-user state to another run state, `init` sets the `ioctl`(2) states of the console to those modes saved in the file `/etc/ioctl.syscon`. `init` writes this file whenever the single-user state is entered.

Run Level Changes

When a run level change request is made, `init` sends the warning signal (`SIGTERM`) to all processes that are undefined in the target run level. `init` waits five seconds before forcibly terminating these processes by sending a kill signal (`SIGKILL`).

When `init` receives a signal telling it that a process it spawned has died, it records the fact and the reason it died in `/var/adm/utmp` and `/var/adm/wtmp` if it exists [see `who`(1)]. A history of the processes spawned is kept in `/var/adm/wtmp`.

If `init` receives a `powerfail` signal (`SIGPWR`) it scans `/etc/inittab` for special entries of the type `powerfail` and `powerwait`. These entries are invoked (if the run levels permit) before any further processing takes place. In this way `init` can perform various cleanup and recording functions during the powerdown of the operating system.

/etc/defaults/init File

Default values can be set for the following flags in `/etc/default/init`. For example: `TZ` =US/Pacific

`TZ`	Either specifies the timezone information [see `ctime`(3C)] or the name of a timezone information file `/usr/share/lib/zoneinfo`.
`LC_CTYPE`	Character characterization information.
`LC_MESSAGES`	Message translation.

LC_MONETARY	Monetary formatting information.
LC_NUMERIC	Numeric formatting information.
LC_TIME	Time formatting information.
LC_ALL	If set, all other LC_* environmental variables take-on this value.
LANG	If LC_ALL is not set, and any particular LC_* is also not set, the value of LANG is used for that particular environmental variable.

telinit

telinit , which is linked to /sbin/init , is used to direct the actions of init . It takes a one-character argument and signals init to take the appropriate action.

SECURITY

init uses pam(3) for session management. The PAM configuration policy, listed through /etc/pam.conf , specifies the session management module to be used for init . Here is a partial pam.conf file with entries for init using the UNIX session management module.

```
init    session    required    /usr/lib/security/pam_unix.so.1
```

If there are no entries for the init service, then the entries for the "other" service will be used.

OPTIONS

0	Go into firmware.
1	Put the system in system administrator mode. All local file systems are mounted. Only a small set of essential kernel processes are left running. This mode is for administrative tasks such as installing optional utility packages. All files are accessible and no users are logged in on the system.
2	Put the system in multi-user mode. All multi-user environment terminal processes and daemons are spawned. This state is commonly referred to as the multi-user state.
3	Extend multi-user mode by making local resources available over the network.
4	Is available to be defined as an alternative multi-user environment configuration. It is not necessary for system operation and is usually not used.
5	Shut the machine down so that it is safe to remove the power. Have the machine remove power, if possible.

6	Stop the operating system and reboot to the state defined by the `initdefault` entry in `/etc/inittab` .
a, b, c	Process only those `/etc/inittab` entries having the `a` , `b` , or `c` run level set. These are pseudo-states, which may be defined to run certain commands, but which do not cause the current run level to change.
Q, q	Re-examine `/etc/inittab` .
S, s	Enter single-user mode. This is the only run level that doesn't require the existence of a properly formatted `/etc/inittab` file. If this file does not exist, then by default, the only legal run level that `init` can enter is the single-user mode. When in single-user mode, the filesystems required for basic system operation will be mounted. When the system comes down to single-user mode, these file systems will remain mounted (even if provided by a remote file server), and any other local filesystems will also be left mounted. During the transition down to single-user mode, all processes started by `init` or `init.d` scripts that should only be running in multi-user mode are killed. In addition, any process that has a `utmp` entry will be killed. This last condition insures that all port monitors started by the SAC are killed and all services started by these port monitors, including `ttymon` login services, are killed.

FILES

`/etc/inittab`	Controls process dispatching by `init` .
`/var/adm/utmp`	Accounting information.
`/var/adm/wtmp`	History of all logins since file was last created.
`/etc/ioctl.syscon`	System console states.
`/dev/console`	System console device.
`/etc/default/init`	Environment variables.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`init` requires privilege to run in the Trusted Solaris environment.

SEE ALSO

Trusted Solaris 7 Reference Manual

`login`(1) , `kill`(2) , `inittab`(4)

SunOS 5.7 Reference Manual

`sh`(1) , `stty`(1) , `who`(1) , `shutdown`(1M) , `su`(1M) , `ttymon`(1M) , `ioctl`(2) , `ctime`(3C) , `pam`(3) , `pam.conf`(4) , `utmp`(4) , `utmpx`(4) , `attributes`(5) , `pam_unix`(5) , `termio`(7I)

DIAGNOSTICS

If `init` finds that it is respawning an entry from `/etc/inittab` more than ten times in two minutes, assumes that there is an error in the command string in the entry, and generates an error message on the system console. It will then refuse to respawn this entry until either five minutes has elapsed or it receives a signal from a user-spawned `init` or `telinit` . This prevents `init` from eating up system resources when someone makes a typographical error in the `inittab` file, or a program is removed that is referenced in `/etc/inittab` .

NOTES

`init` and `telinit` can be run only by a privileged user.

The `S` or `s` state must not be used indiscriminately in `/etc/inittab` . When modifying this file, it is best to avoid adding this state to any line other than `initdefault` .

If a default state is not specified in the `initdefault` entry in `/etc/inittab` , state `6` is entered. Consequently, the system will loop by going to firmware and rebooting continuously.

If the `utmp` file cannot be created when booting the system, the system will boot to state “ `s` ” regardless of the state specified in the `initdefault` entry in `/etc/inittab` . This can occur if the `/var` file system is not accessible.

NAME

in.named, named – Internet domain name server

SYNOPSIS

in.named [–d *debuglevel*] [–q] [–r] [–f] [–p *remote/local-port*] [–w *dirname*] [[–b | –c] *configfile*]

DESCRIPTION

`in.named` is the Internet domain name server. `in.named` spawns the `named-xfer` process whenever it needs to perform a zone transfer. See `named-xfer`(1M) .

The `in.named` name service is used by hosts on the Internet to provide access to the Internet distributed naming database. See *RFC 1034* and *RFC 1035* for more information on the Internet domain name system.

With no arguments, `in.named` reads the default configuration file `/etc/named.conf` for any initial data, and listens for queries. Any additional arguments beyond those shown in the `SYNOPSIS` section are interpreted as the names of boot files. If multiple boot files are specified, only the last is used.

The name server reads the boot file to obtain instructions on where to find its initial data.

In a Trusted Solaris system, `in.named` listens for input requests on a multilevel port (MLP) and sends responses to the DNS client at the sensitivity label of the client's request. Thus, though `in.named` runs at the sensitivity label `ADMIN_LOW` , it can accept requests at any sensitivity label. `in.named` can also serve DNS clients and communicate with other DNS name servers on either Trusted Solaris hosts or non-trusted hosts.

The DNS name server running on a Trusted Solaris machine is viewed as a supplier of public information, and the name database that it maintains is considered trusted. `in.named` requires the trusted path attribute, and it requires that the `/etc/named.boot` file, zone files, and other configuration files that it uses be at the sensitivity label `ADMIN_LOW` . As part of the name database, these files and their contents are also considered trusted; thus `in.named` can query any DNS name server specified in the files. The DNS name servers specified in these files may reside on either Trusted Solaris hosts or non-trusted hosts.

OPTIONS

`–w` *dirname* — Change the current working directory of `in.named` to *dirname* .

`–b` *bootfile* — Use bootfile rather than `/etc/named.conf` . This options allows filenames to begin with a leading dash.

`–c` *bootfile* — Use bootfile rather than `/etc/named.conf` . This options allows filenames to begin with a leading dash.

–d *level*
: Print debugging information. *level* is a number indicating the level of messages printed.

–p *remote/local-port*
: Use different, port numbers. The default is the standard port number as returned by `getservbyname`(3N) for service domain. The –p argument can specify up to two port numbers. The specification of two port numbers requires a ‘ / ’ (slash) separator. In this case, the first port is used when contacting remote servers, and the second one is the service port bound by the local instance of `in.named` . This option is used mostly for debugging purposes.

–q
: Trace all incoming queries. Note: this option is ignored in favor of the boot file directive, `options query-log` , when both options are used.

–r
: Turns recursion off in the server. Answers can come only from local (primary or secondary) zones. This option can be used on root servers. Note: This option will probably be eventually abandoned in favor of the boot file directive, `options no-recursion` .

USAGE

/etc/named.conf File Directives

The following is a simple configuration file `/etc/named.conf` containing directives to guide the `in.named` process at startup time.

```
options {
            directory  "/usr/local/adm/named";
            pid-file   "/var/named/named.pid";
            named-xfer "/usr/sbin/named-xfer";
            forwarders       {
                     10.0.0.78;
                     10.2.0.78;
            };
            transfers-in     10;
            forward only;
            fake-iquery yes;
            pollfd-chunk-size 20;
      };

      logging {
              category lame-servers { null; };
              category cname  { null; };
      };
```

```
zone "." in {
        type hint;
        file "root.cache";
};

zone "cc.berkeley.edu" in {
        type slave;
        file "128.32.137.3";
        masters { 128.32.137.8; };
};

zone "6.32.128.in-addr.arpa" in {
        type slave;
        file "128.32.137.3";
        masters { 128.32.137.8; };
};
zone "0.0.127.in-addr.arpa" in {
        type master;
        file "master/db.127";
};

zone "berkeley.edu" in {
        type master;
        file "berkeley.edu.zone";
};

zone "32.128.in-addr.arpa" in {
        type master;
        file "ucbhosts.rev";
};
```

The configuration file consists of sections and comments. Sections end with a ' `;` ' and contain statements which are enclosed in ' `{ }` ' and may span multiple lines. The following sections are supported: `options` , `zone` , `server` , `logging` , `acl` , `include` , and `key` .

Comments Syntax

The following are examples of comments syntax in BIND 8.1:

```
/* This is a BIND comment as in C */
// This is a BIND comment as in C++
#  This is a BIND comment as in common Unix shells and perl
```

WARNING: you cannot use the semicolon character (`;`) to start a comment.

Options Section

The sytax of the options section is as follows:

```
options {
       [ directory path_name; ]
       [ named-xfer path_name; ]
       [ pid-file path_name; ]
       [ auth-nxdomain yes_or_no; ]
```

```
        [ fake-iquery yes_or_no; ]
        [ fetch-glue yes_or_no; ]
        [ multiple-cnames yes_or_no; ]
        [ notify yes_or_no; ]
        [ recursion yes_or_no; ]
        [ forward ( only | first ); ]
        [ forwarders { [ in_addr ; [ in_addr ; ... ] ] }; ]
        [ check-names ( master | slave | response ) ( warn | fail | ignore); ]
        [ allow-query { address_match_list }; ]
        [ allow-transfer { address_match_list }; ]
        [ listen-on [ port ip_port ] { address_match_list }; ]
        [ query-source [ address ( ip_addr | * ) ] [ port ( ip_port | * ) ] ; ]
        [ max-transfer-time-in number; ]
        [ transfer-format ( one-answer | many-answers ); ]
        [ transfers-in  number; ]
        [ transfers-out number; ]
        [ transfers-per-ns number; ]
        [ coresize size_spec ; ]
        [ datasize size_spec ; ]
        [ files size_spec ; ]
        [ stacksize size_spec ; ]
        [ clean-interval number; ]
        [ interface-interval number; ]
        [ scan-interval number; ]
        [ topology { address_match_list }; ]
      };
```

Definitions and Use of Options

The `options` section sets up global options to be used by BIND. This section may appear at only once in a configuration file; if more than one occurrence is found, the first occurrence determines the actual options used, and a warning will be generated. If there is no `options` section, an `options` block with each `option` set to its default will be used.

Pathnames

directory — The working directory of the server. Any non-absolute pathnames in the configuration file will be taken as relative to this directory. The default location for most server output files (for example, " `named.run` ") is this directory. If a directory is not specified, the working directory defaults to " `.` ", the directory from which the server was started. The directory specified should be an absolute path.

named-xfer — The pathname to the `named-xfer` program that the server uses for inbound zone transfers. If not specified, the default is operating system dependent, for example, “ `/usr/sbin/named-xfer` ”).

pid-file — The pathname of the file the server writes its process ID in. If not specified, the default is operating system dependent, but is usually " `/var/run/named.pid` " or " `/etc/named.pid`

". The pid-file is used by programs like " ndc " that want to send signals to the running nameserver.

Boolean Options

auth-nxdomain — If yes , then the AA bit is always set on NXDOMAIN responses, even if the server is not actually authoritative. The default is yes . Do not turn off auth-nxdomain unless you are sure you know what you are doing, as some older software will not like it.

fake-iquery — If yes , the server will simulate the obsolete DNS query type IQUERY . The default is no .

fetch-glue — If yes (the default), the server will fetch "glue" resource records it does not have when constructing the additional data section of a response. fetch-glue no can be used in conjunction with recursion no to prevent the server's cache from growing or becoming corrupted (at the cost of requiring more work from the client).

multiple-cnames — If yes, then multiple CNAME resource records will be allowed for a domain name. The default is no . Allowing multiple CNAME records is against standards and is not recommended. Multiple CNAME support is available because previous versions of BIND allowed multiple CNAME records, and these records have been used for load balancing by a number of sites.

notify — If yes (the default), DNS NOTIFY messages are sent when a zone the server is authoritative for changes. The use of NOTIFY speeds convergence between the master and its slaves. Slave servers that receive a NOTIFY message and understand it will contact the master server for the zone and see if they need to do a zone transfer, and if they do, they will initiate it immediately. The notify option may also be specified in the zone section, in which case it overrides the options notify statement.

recursion — If yes , and a DNS query requests recursion , then the server will attempt to do all the work required to answer the query. If recursion is not on, the server will return a referral to the client if it doesn't know the answer. The default is yes . See also fetch-glue above.

Forwarding

The forwarding facility can be used to create a large sitewide cache on a few servers, reducing traffic over links to external name servers. It can also be used to allow queries by servers that do not have direct access to the Internet, but wish to look up exterior names anyway. Forwarding occurs only on those queries for which the server is not authoritative, and it does not have the answer in its cache.

forward — This option is only meaningful if the `forwarders` list is not empty. A value of first, the default, causes the server to query the `forwarders` first, and if that doesn't answer the question, the server will then look for the answer itself. If `only` is specified, the server will only query the `forwarders` .

forwarders — Specifies the IP addresses to be used for forwarding. The default is the empty list (no forwarding).

Future versions of BIND 8 will provide a more powerful forwarding system. The syntax described above will continue to be supported.

Name Checking

The server can check domain names based upon their expected client contexts. For example, a domain name used as a hostname can be checked for compliance with the valid hostnames defined in the RFC s. Three checking methods are available:

ignore — No checking is done.

warn — Names are checked against their expected client contexts. Invalid names are logged, but processing continues normally.

fail — Names are checked against their expected client contexts. Invalid names are logged, and the offending data is rejected.

The server can check names in three areas: master zone files, slave zone files, and in responses to queries the server has initiated. If `check-names response fail` has been specified, and answering the client's question would require sending an invalid name to the client, the server will send a `REFUSED` response code to the client.

The defaults are:

```
check-names master fail;
check-names slave warn;
check-names response ignore;
```

`check-names` may also be specified in the zone section, in which case it overrides the options `check-names` statement. When used in a zone section, the area is not specified (because it can be deduced from the zone type).

Access Control

Access to the server can be restricted based on the IP address of the requesting system. See address_match_list for details on how to specify IP address lists.

`allow-query` — Specifies which hosts are allowed to ask ordinary questions. `allow-query` may also be specified in the zone section, in which case it overrides the options `allow-query` statement. If not specified, the default is to allow queries from all hosts.

`allow-transfer` — Specifies which hosts are allowed to receive zone transfers from the server. `allow-transfer` may also be specified in the zonesection, in which case it overrides the options `allow-transfer` statement. If not specified, the default is to allow transfers from all hosts.

Interfaces

The interfaces and ports that the server will answer queries from may be specified using the `listen-on` option. `listen-on` takes an optional port, and an `address_match_list` . The server will listen on all interfaces allowed by the address match list. If a port is not specified, port 53 will be used.

Multiple listen-on statements are allowed. For example,

```
listen-on { 5.6.7.8; };
listen-on port 1234 { !1.2.3.4; 1.2/16; };
```

If no `listen-on` is specified, the server will listen on port 53 on all interfaces.

Query Address

If the server does not know the answer to a question, it will query other name servers. `query-source` specifies the address and port used for such queries. If address is `*` or is omitted, a wildcard IP address (`INADDR_ANY`) will be used. If port is `*` or is omitted, a random unprivileged port will be used. The default is:

```
query-source address * port *;
```

Note: `query-source` currently applies only to UDP queries; TCP queries always use a wildcard IP address and a random unprivileged port.

Zone Transfers

max-transfer-time-in — Inbound zone transfers (`named-xfer` processes) running longer than this many minutes will be terminated. The default is `120` minutes.

transfer-format — The server supports two zone transfer methods. `one-answer` uses one DNS message per resource record transferred. `many-answers` packs as many resource records as possible into a message.

`many-answers` is more efficient, but is only known to be understood by BIND 8.1 and patched versions of BIND 4.9.5. The default is `one-answer` . `transfer-format` may be overridden on a per-server basis by using the server section.

transfers-in — The maximum number of inbound zone transfers that can be running concurrently. The default value is 10. Increasing `transfers-in` may speed up the convergence of slave zones, but it also may increase the load on the local system.

transfers-out — This option will be used in the future to limit the number of concurrent outbound zone transfers. It is checked for syntax, but is otherwise ignored.

transfers-per-ns — The maximum number of inbound zone transfers (`named-xfer` processes) that can be concurrently transferring from a given remote name server. The default value is 2. Increasing `transfers-per-ns` may speed up the convergence of slave zones, but it also may increase the load on the remote name server. `transfers-per-ns` may be overridden on a per-server basis by using the transfers statement in the server section.

Resource Limits

The server's usage of many system resources can be limited. Some operating systems do not support some of the limits, and a warning will be generated if an unsupported limit is set in the configuration file.

Scaled values are allowed when specifying resource limits. For example, `1G` can be used instead of `1073741824` to specify a limit of one gigabyte, unlimited requests unlimited use, or the maximum available amount. Default uses the limit that was in force when the server was started. See `ulimit`(1) for a discussion of `ulimit -a` (ksh only) for defaults.

coresize — The maximum size of a core dump. The default is system dependent.

datasize — The maximum amount of data memory the server may use. The default is system dependent.

files — The maximum number of files that the server may have open concurrently. The default is system dependent.

stacksize — The maximum amount of stack memory the server may use. The default is system dependent.

Topology

All other things being equal, when the server chooses a name server to query from a list of name servers, it prefers the one that is topologically closest to itself. The topology statement takes an `address_match_list` and interprets it in a special way. Each top-level list element is assigned a distance. Non-negated elements get a distance based on their position in the list, where the closer the match is to the start of the list, the shorter the distance is between it and the server. A negated match will be assigned the maximum distance from the server. If there is no match, the address will get a distance which is further than any non-negated list element, and closer than any negated element. For example,

```
topology {
     10/8;
     !1.2.3/24;
     { 1.2/16; 3/8; };
};
```

will prefer servers on network 10 the most, followed by hosts on network 1.2.0.0 (netmask 255.255.0.0) and network 3, with the exception of hosts on network 1.2.3 (netmask 255.255.255.0), which is preferred least of all. The default topology is

```
topology { localhost; localnets; };
```

The Server Section

The syntax of the server section is as follows:

```
server ip_addr {
       [ bogus yes_or_no; ]
       [ transfers number; ]
       [ transfer-format ( one-answer | many-answers ); ]
       [ keys { key_id [key_id ... ] }; ]
};
```

The server statement defines the characteristics to be associated with a remote name server.

If you discover that a server is giving out bad data, marking it as bogus will prevent further queries to it. The default value is `no` .

The server supports two zone transfer methods. The first, `one-answer` , uses one DNS message per resource record transferred. `many-answers` packs as many resource records as possible into a message. `many-answers` is more efficient, but is only known to be understood by BIND 8.1 and patched

versions of BIND 4.9.5. You can specify which method to use for a server with the `transfer-format` option. If `transfer-format` is not specified, the `transfer-format` specified by the options statement will be used.

The transfers will be used in a future release of the server to limit the number of concurrent inbound zone transfers from the specified server. It is checked for syntax but is otherwise ignored.

The `keys` statement is intended for future use by the server. It is checked for syntax but is otherwise ignored.

The Zone Section

The syntax of the zone section is as follows:

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
       type master;
       file path_name;
       [ check-names ( warn | fail | ignore ); ]
       [ allow-update { address_match_list }; ]
       [ allow-query { address_match_list }; ]
       [ allow-transfer { address_match_list }; ]
       [ notify yes_or_no; ]
       [ also-notify { ip_addr; [ ip_addr; ... ] };
};

zone domain_name [ ( in | hs | hesiod | chaos ) ] {
       type ( slave | stub );
       [ file path_name; ]
       masters { ip_addr; [ ip_addr; ... ] };
       [ check-names ( warn | fail | ignore ); ]
       [ allow-update { address_match_list }; ]
       [ allow-query { address_match_list }; ]
       [ allow-transfer { address_match_list }; ]
       [ max-transfer-time-in number; ]
       [ notify yes_or_no; ]
       [ also-notify { ip_addr; [ ip_addr; ... ] };
};

zone . [ ( in | hs | hesiod | chaos ) ] {
       type hint;
       file path_name;
       [ check-names ( warn | fail | ignore ); ]
};
```

Zone types are defined as follows:

master The master copy of the data in a `zone` .

slave A `slave zone` is a replica of a `master zone` . The masters list specifies one or more IP addresses that the `slave` contacts to update its copy of the `zone` . If file is specified, then the replica will be written to the file. Use of file is recommended, since it often speeds server startup and eliminates a needless waste of bandwidth.

stub A `stub zone` is like a `slave zone`, except that it replicates only the NS records of a `master zone` instead of the entire zone.

hint The initial set of root name servers is specified using a `hint zone`. When the server starts up, it uses the root hints to find a root name server and get the most recent list of root name servers.

Note: previous releases of BIND used the term `primary` for a `master zone`, `secondary` for a `slave zone`, and `cache` for a `hint zone`.

The zone's name may optionally be followed by a `class`. If a `class` is not specified, `class in` is used.

Zone options are described as follows:

check-names	See `Name Checking`.
allow-query	See the description of `allow-query` in the `Access Control` section.
allow-update	Specifies which hosts are allowed to submit dynamic DNS updates to the server. The default is to deny updates from all hosts.
allow-transfer	See the description of `allow-transfer` in the `Access Control` section.
max-transfer-time-in	See the description of `max-transfer-time-in` in the `Zone Transfers` section.
notify	See the description of `notify` in the `Boolean Options` section.
also-notify	`also-notify` is only meaningful if `notify` is active for this zone.

The set of machines that will receive a `DNS NOTIFY` message for this zone is made up of all the listed name servers for the zone (other than the primary master) plus any IP addresses specified with `also-notify`. `also-notify` is not meaningful for `stub` zones. The default is the empty list.

The Logging Section

The syntax of the `logging` section is as follows:

```
logging {
       [ channel channel_name {
         ( file path_name
            [ versions ( number | unlimited ) ]
            [ size size_spec ]
          | syslog ( kern | user | mail | daemon | auth | syslog | lpr |
                     news | uucp | cron | authpriv | ftp |
                     local0 | local1 | local2 | local3 |
                     local4 | local5 | local6 | local7 )
```

```
            | null );

          [ severity ( critical | error | warning | notice |
                       info  | debug [ level ] | dynamic ); ]
          [ print-category yes_or_no; ]
          [ print-severity yes_or_no; ]
          [ print-time yes_or_no; ]
        }; ]

        [ category category_name {
          channel_name; [ channel_name; ... ]
        }; ]
        ...
};
```

The `logging` statement configures a wide variety of logging options for the name server. Its channel phrase associates output methods, format options and severity levels with a name that can then be used with the category phrase to select how various classes of messages are logged.

Only one `logging` statement is used to define as many channels and categories as are wanted. If there are multiple `logging` statements in a configuration, the first defined determines the logging, and warnings are issued for the others. If there is no `logging` statement, the default logging configuration will be:

```
logging {
    category default { default_syslog; default_debug; };
    category panic { default_syslog; default_stderr; };
    category packet { default_debug; };
    category eventlib { default_debug; };
};
```

The Channel Phrase

All log output goes to one or more "channels"; you can make as many of them as you want.

Every `channel` definition must include a clause that says whether messages selected for the channel go to a file, to a particular `syslog` facility, or are discarded. It can optionally also limit the message severity level that will be accepted by the channel (default is " `info` "), and whether to include a named-generated time stamp, the category name and / or severity level (default is not to include any).

The word `null` as the destination option for the channel will cause all messages sent to it to be discarded; other options for the channel are meaningless.

The file clause can include limitations both on how large the file is allowed to become, and how many versions of the file will be saved each time the file is opened.

The size option for files is simply a hard ceiling on log growth. If the file ever exceeds the size, then named will just not write anything more to it until the file is reopened; exceeding the size does not automatically trigger a reopen. The default behavior is to not limit the size of the file.

If you use the version logfile option, then named will retain that many backup versions of the file by renaming them when opening. For example, if you choose to keep 3 old versions of the file "lamers.log" then just before it is opened lamers.log.1 is renamed to lames.log.2, lamers.log.0 is renamed to lamers.log.1, and lamers.log is renamed to lamers.log.0. No rolled versions are kept by default. The unlimited keyword is synonymous with 99 in current BIND releases.

The argument for the `syslog( )` clause is a `syslog( )` facility as described in the `syslog`(3) manual page. How `syslogd`(1M) will handle messages sent to this facility is described in the `syslog.conf`(4) manual page. If you have a system which uses a very old version of `syslog( )` that only uses two arguments to the `openlog( )` function, then this clause is silently ignored.

The severity clause works like the "priorities" to `syslog( )`, except that they can also be used if you are writing straight to a file rather than using `syslog( )`. Messages which are not at least of the severity level given will not be selected for the channel; messages of higher severity levels will be accepted.

If you are using `syslog( )`, then the `syslog.conf` priorities will also determine what eventually passes through. For example, defining a channel facility and severity as daemon and debug but only logging `daemon.warning` by way of `syslog.conf` will cause messages of `severity info` and `notice` to be dropped. If the situation were reversed, with named writing messages of only warning or higher, then `syslogd` would print all messages it received from the channel.

The server can supply extensive debugging information when it is in debugging mode. If the server's global debug level is greater than zero, then debugging mode will be active. The global debug level is set either by starting the server with the –`d` option followed by a positive integer, or by sending the server the `SIGUSR1` signal (for example, by using " `ndc trace` "). The global debug level can be set to zero, and debugging mode turned off, by sending the server the `SIGUSR2` signal ("ndc notrace". All debugging messages in the server have a debug level, and higher debug levels give more more detailed output. Channels that specify a specific debug severity, for example:

```
channel specific_debug_level {
    file "foo";
```

```
    severity debug 3;
};
```

will get debugging output of level 3 or less any time the server is in debugging mode, regardless of the global debugging level. Channels with dynamic severity use the server's global level to determine what messages to print.

If `print-time` has been turned on, then the date and time will be logged. `print-time` may be specified for a `syslog()` channel, but is usually pointless since `syslog()` also prints the date and time. If `print-category` is requested, then the category of the message will be logged as well. Finally, if `print-severity` is on, then the severity level of the message will be logged. The `print-options` may be used in any combination, and will always be printed in the following order: `time, category, severity`. Here is an example where all three `print-options` are on:

```
28-Apr-1997 15:05:32.863 default: notice: Ready to answer queries.
```

There are four predefined channels that are used for default logging for `in.named` as follows. How they are used is described in the next section.

```
channel default_syslog {
    syslog daemon;        # send to syslog's daemon facility
    severity info;        # only send priority info and higher
};

channel default_debug {
    file "named.run";     # write to named.run in the working directory
    severity dynamic;     # log at the server's current debug level
};

channel default_stderr {  # writes to stderr
    file "<stderr>";      # this is illustrative only;
  # there's currently
                          # no way of specifying an internal file
                          # descriptor in the configuration language.
    severity info;        # only send priority info and higher
};

channel null {
    null;                 # toss anything sent to this channel
};
```

Once a channel is defined, it cannot be redefined. Thus you cannot alter the built-in channels directly, but you can modify the default logging by pointing categories at channels you have defined.

The Category Phase

There are many categories, so you can send the logs you want to see wherever you want, without seeing logs you do not want. If you do not specify a list of channels for a category, then log messages in that category will be sent to the default category instead. If do not specify a default category, the following "default default" is used:

```
category default { default_syslog; default_debug; };
```

For example, if you want to log security events to a file, but you also want keep the default logging behavior, specify the following:

```
channel my_security_channel {
    file "my_security_file";
    severity info;
};
category security { my_security_channel; default_syslog; default_debug; };
```

To discard all messages in a category, specify the `null` channel:

```
category lame-servers { null; };
category cname { null; };
```

The following categories are available:

default — The catch-all. Many things still are not classified into categories, and they all end up here. Also, if you do not specify any channels for a category, the default category is used instead. If you do not define the default category, the following definition is used:

```
category default { default_syslog; default_debug; };
```

config — High-level configuration file processing.

parser — Low-level configuration file processing.

queries — A short log message is generated for every query the server receives.

lame-servers — Messages like "Lame server on ..."

statistics — Statistics.

panic	If the server has to shut itself down due to an internal problem, it will log the problem in this category as well as in the problem's native category. If you do not define the panic category, the following definition is used: `category panic { default_syslog; default_stderr; };`
update	Dynamic updates.
ncache	Negative caching.
xfer-in	Zone transfers the server is receiving.
xfer-out	Zone transfers the server is sending.
db	All database operations.
eventlib	Debugging info from the event system. Only one channel may be specified for this category, and it must be a file channel. If you do not define the eventlib category, the following definition is used: `category eventlib { default_debug; };`
packet	Dumps of packets received and sent. Only one channel may be specified for this category, and it must be a file channel. If you do not define the packet category, the following definition is used: `category packet { default_debug; };`
notify	The NOTIFY protocol.
cname	Messages like "... points to a CNAME".
security	Approved/unapproved requests.
os	Operating system problems.
insist	Internal consistency check failures.
maintenance	Periodic maintenance events.
load	Zone loading messages.
response-checks	Messages arising from response checking, such as "Malformed response ...", "wrong ans. name ...", "unrelated additional info ...", "invalid RR type ...", and "bad referral ...".

The Key Section

The syntax of the key section is as follows:

```
key key_id {
    algorithm algorithm_id;
    secret secret_string;
};
```

The key section defines a key ID which can be used in a server section to associate an authentication method with a particular name server.

A key ID must be created with the key statement before it can be used in a server definition.

The algorithm_id is a string that specifies a security / authentication algorithm. secret_string is the secret to be used by the algorithm.

The key statement is intended for future use by the server. It is checked for syntax but is otherwise ignored.

The Include Section

The syntax of the include section is as follows:

```
include path_name;
```

The include statement inserts the specified file at the point that the include statement is encountered. It cannot be used within another statement, though, so a line such as acl internal_hosts { "include internal_hosts.acl" } is not allowed. Use include to break the configuration up into easily-managed chunks. For example:

```
include "/etc/security/keys.bind";
include "/etc/acls.bind";
```

could be used at the top of a BIND configuration file in order to include any ACL or key information.

Be careful not to type " #include ", like you would in a C program, because " # " is used to start a comment.

The ACL Format

The syntax of the ACL section is as follows:

```
acl name {
    address_match_list
};
```

The `acl` statement creates a named address match list. It gets its name from a primary use of address match lists: Access Control Lists (ACL s).

Note that an address match list's name must be defined with `acl` before it can be used elsewhere; no forward references are allowed.

The following ACL s are built-in:

any	Allows all hosts.
none	Denies all hosts.
localhost	Allows the IP addresses of all interfaces on the system.
localnets	Allows any host on a network for which the system has an interface.

Zone File Format

The zone files are also known as the authoritative master files (data files) for a zone. In the boot file, references were made to these files as part of the specification of any primary directives.

Two classes of entries populate the zone files, directives and resource records. The start of the zone file is likely to contain one or two directives that establish a context that modifies the way subsequent records are interpreted.

Resource records for a zone determine how a zone is managed by establishing zone characteristics. For example, one type of zone record establishes the zone's mailbox information.

The very first record of each zone file should be a Start-of-Authority record (SOA) for a zone. A multiple-line SOA record is presented below. The meaning of the values in this sample will become clearer with the help of a list that describes the purpose of each field in the zone record (see the SOA list subitem under the `rr-type` list item in, Format of Resource Records in Zone Files).

```
@ IN SOA ucbvax.Berkeley.EDU. rwh.ucbvax.Berkeley.EDU. (
1989020501   ;serial
10800        ;refresh
3600         ;retry
3600000      ;expire
86400 )      ;minimum
```

Resource records normally end at the end of a line, but may be continued across lines between opening and closing parentheses (as demonstrated by the preceding sample).

Comments are introduced by semicolons. They continue to the end of the line.

Directives in Zone Files

There are two control directives that help determine how the zone file is processed, `$INCLUDE` and `$ORIGIN` .

The `$INCLUDE` directive refers to still another file within which zone characteristics are described. Such files typically contain groups of resource records, but they may also contain further directives.

The `$ORIGIN` directive establishes a current origin that is appended to any domain values that do not end with a ‘ . ’ (dot). The placeholder domain represents the first resource record field as shown in Format of Resource Records in Zone Files.The format for these directives is:

```
$INCLUDE filename opt-current-domain
$ORIGIN current-domain
```

where:

current-domain — Specifies the value of the current origin that remains in effect for this configuration file unless a subsequent `$ORIGIN` directive overrides it for the remaining portion of the file.

filename — Specifies a file, the contents of which are, in effect, incorporated into the configuration file at the location of the corresponding `$INCLUDE` directive.

opt-current-domain — Optionally defines a current origin that is applicable only to the records residing in the specified file in the corresponding `$INCLUDE` directive. This directive overrides the origin given in a preceding `$ORIGIN` directive, but only for the scope of the included text. See also `current-domain` .

Neither the opt-current-domain argument of `$INCLUDE` nor the `$ORIGIN` directive in the included file can affect the current origin in effect for the remaining records in the main configuration file (as defined by those `$ORIGIN` directives that reside there).

Format of Resource Records in Zone Files

The format of the resource records is:

```
domain opt-ttl opt-class rr-type rr-data...
```

where:

domain — Specifies the domain being described by the current line and any following lines that lack a value for this field. Beware of any domain values that you enter without full qualification, because the value of the current origin will be appended to them. The value of the current origin is appended when domain does not end with a dot.

A domain value specified as the symbol `@` is replaced with the value of the current origin. The `current-domain` or any locally-overriding `opt-current-domain` value is used as its replacement. (For a discussion of these placeholders, see the earlier discussion of the `$ORIGIN` and `$INCLUDE` directives.)

A domain value specified as a ‘ `.` ’ (dot) represents the root.

opt-ttl — Specifies the number of seconds corresponding to the `time-to-live` value applicable to the zone characteristic that is defined in the remaining fields. This field is optional. It defaults to zero. Zero is interpreted as the minimum value specified in the SOA record for the zone.

opt-class — Specifies the object address type; currently only one type is supported, `IN` , for objects connected to the Internet.

rr-type rr-data ... — Specifies values that describe a zone charateristic. Permissible `rr-type` and other field values are listed below. The field values are listed in the order that they must appear.

A address

Specifies the host address (in `dotted-quad` format). DCE or AFS server.

CNAME canonical-name

Specifies in a `domain-name` format the canonical name for the alias (domain).

HINFO cpu-type OS-type

Host information supplied in terms of a CPU type and an OS type.

MX preference mail-exchanger

Specifies in `domain-name` format a mail exchanger preceded by a preference value (between 0 and 32767), with lower numeric values representing higher logical preferences.

NS authoritative-server

Specifies in `domain-name` format an authoritative name server.

NULL

Specifies a null zone record.

PTR domain-pointer

Specifies in `domain-name` format a domain name pointer.

RP mailbox txt-referral

Offers details about how to reach a responsible person for the domain name.

`retry` `expire` `ttl`

SOA host-domain maintainer-addr serial- no refresh

Establishes the start of a zone of authority in terms of the domain of the originating host (host-domain), the domain address of the maintainer (maintainer-addr), a serial number (serial-no), the refresh period in seconds (refresh), the retry period in seconds (retry), the expiration period in seconds (expire), and the minimum time-to-live period in seconds (ttl). See RFC 1035.

The serial number should be changed each time the master file is changed. Secondary servers check the serial number at intervals specified by the refresh time in seconds; if the serial number changes, a zone transfer will be done to load the new data.

If a master server cannot be contacted when a refresh is due, the retry time specifies the interval at which refreshes should be attempted. If a master server cannot be contacted within the interval given by the expire time, all data from the zone is discarded by secondary servers. The minimum value is the time-to-live used by records in the file with no explicit time-to-live value.

The serial number can be given as a dotted number. However, this is a very unwise thing to do, since the translation to normal integers is via concatenation rather than multiplication and addition. You could spell out the year, month, day of month, and 0..99 version number and still fit it inside the unsigned 32-bit size of this field.

This strategy should work for the forseeable future (but is questionable after the year 4293).

For more detailed information, see *RFC 883* .

rr-data ... See the description of `rr-type` .

Consult *Name Server Operations Guide for BIND* for further information about the supported types of resource records.

EXIT STATUS

The `in.named` process returns the following exit values:

`0` Successful completion.

`1` An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.named` accepts requests at any sensitivity label and replies at the sensitivity label of the client's request. `in.named` can serve DNS clients and can communicate with other DNS servers that are on Trusted Solaris hosts or non-trusted hosts.

Files used by `in.named` should be protected from unauthorized access by having the sensitivity label `ADMIN_LOW` .

Invoking `in.named` requires the trusted path attribute, an effective UID of 0, a process sensitivity label of `ADMIN_LOW` , and the following privileges: `net_mac_read` , `net_privaddr` , `net_upgrade_sl` , `proc_setclr` , `sys_trans_label` , `sys_net_config` , and `sys_config` .

FILES

`/etc/named.conf`	Name server configuration boot file.
`/etc/named.pid`	The process ID (on older systems).
`/var/tmp/named.run`	Debug output.
`/var/tmp/named.stats`	Nameserver statistics data.
`/var/tmp/nameddump.db`	Dump of the name servers database.
`/var/tmp/named.pid`	The process ID (on newer systems).

These files have a sensitivity label of `ADMIN_LOW` .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`resolver`(3N)

SunOS 5.7 Reference Manual

`kill`(1) , `signal`(3B) , `resolv.conf`(4) , `attributes`(5)

Braden, R. (Editor), *Requirements for Internet Hosts - Applications and Support, RFC 1123, Internet Engineering* Task Force - Network Working Group, October 1989.

Mockapetris, Paul, *Domain Names - Concepts and Facilities, RFC 1034,* , Network Information Center, SRI International, Menlo Park, Calif., November 1987.

Mockapetris, Paul, *Domain Names - Implementation and Specification, RFC 1035* , Network Information Center, SRI International, Menlo Park, Calif., November 1987.

Mockapetris, Paul, *Domain System Changes and Observations, RFC 973* , Network Information Center, SRI International, Menlo Park, Calif., January 1986.

Partridge, Craig, *Mail Routing and the Domain System, RFC 974* , Network Information Center, SRI International, Menlo Park, Calif., January 1986.

Vixie, Paul, Dunlap, Keven J., Karels, Michael J., *Name Server Operations Guide for BIND* (public domain), Internet Software Consortium, 1995.

NOTES

The following signals have the specified effect when sent to the server process using the `kill`(1) command:

SIGHUP	Causes the server to read `/etc/named.conf` and reload the database.
SIGHUP	Also causes the server to check the serial number on all secondary zones. Normally the serial numbers are only checked at the intervals specified by the SOA record at the start of each zones-definition file.
SIGINT	Dumps the current database and cache to `/var/tmp/nameddump.db` .
SIGIOT	Dumps statistical data into `/var/tmp/named.stats` . Statistical data are appended to the file.
SIGUSR1	Turns on debugging at the lowest level when received the first time; receipt of each additional `SIGUSR1` signal causes the server to increment the debug level.
SIGUSR2	Turns off debugging completely.

SIGWINCH	Toggles logging of all incoming queries through the syslog system daemon. See `syslogd`(1M) .

NAME

in.rarpd, rarpd – DARPA Reverse Address Resolution Protocol server

SYNOPSIS

/usr/sbin/in.rarpd [–d] –a

/usr/sbin/in.rarpd [–d] *device unit*

DESCRIPTION

`in.rarpd` starts a daemon that responds to Reverse Address Resolution Protocol (RARP) requests. The daemon forks a copy of itself that runs in background. It must be started from the trusted path, with a UID of 0 and the label `ADMIN_LOW`. To succeed, it must inherit the `sys_net_conf` and `net_broadcast` privileges.

RARP is used by machines at boot time to discover their Internet Protocol (IP) address. The booting machine provides its Ethernet address in a RARP request message. Using the `ethers` and `hosts` databases, `in.rarpd` maps this Ethernet address into the corresponding IP address which it returns to the booting machine in an RARP reply message. The booting machine must be listed in both databases for `in.rarpd` to locate its IP address. `in.rarpd` issues no reply when it fails to locate an IP address.

`in.rarpd` uses the STREAMS-based Data Link Provider Interface (DLPI) message set to communicate directly with the datalink device driver.

OPTIONS

The following options are supported:

–a Get the list of available network interfaces from IP using the SIOCGIFADDR ioctl and start a RARP daemon process on each interface returned.

–d Print assorted debugging messages while executing.

EXAMPLES

EXAMPLE 1 Starting an `in.rarpd` Daemon for Each Network Interface Name Returned From `/dev/ip`:

The following command starts an `in.rarpd` for each network interface name returned from `/dev/ip`:

```
example# /usr/sbin/in.rarpd –a
```

EXAMPLE 2 Starting an `in.rarpd` Daemon on the Device `/dev/le` with the Device Instance Number `0`

The following command starts one `in.rarpd` on the device `/dev/le` with the device instance number `0`.

```
example# /usr/sbin/in.rarpd le 0
```

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.rarpd` should be started from the trusted path with a UID `0` and sensitivity label of `ADMIN_LOW`. It must inherit the `sys_net_config` and `net_broadcast` privileges.

FILES

`/etc/ethers`	File or NIS+ map of host names and ethernet addresses.
`/etc/hosts`	File or NIS+ map of Internet host names and addresses.
`/tftpboot`	Directory for remote boot scripts.
`/dev/ip`	List of available network interfaces.
`/dev/arp`	Address resolution protocol list.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`ifconfig`(1M)

SunOS 5.7 Reference Manual

`boot`(1M) , `ethers`(4) , `hosts`(4) , `netconfig`(4) , `attributes`(5) , `dlpi`(7P)

RFC-903, *A Reverse Address Resolution Protocol* , Network Information Center, SRI International.

Unix International, *Data Link Provider Interface* , Version 2, May 7, 1991, Sun Microsystems, 800-6915-01.

NAME

in.rdisc, rdisc – Network router discovery daemon

SYNOPSIS

/usr/sbin/in.rdisc [–a] [–f] [–s] [*send-address*] [*receive-address*]

/usr/sbin/in.rdisc –r [–p *preference*] [–T *interval*] [*send-address*] [*receive-address*]

DESCRIPTION

`in.rdisc` implements the ICMP router discovery protocol. The first form of the command is used on hosts and the second form is used on routers. On a host, `in.rdisc` is invoked at boot time to populate the network routing tables with default routes. On a router, it is also invoked at boot time in order to start advertising the router to all the hosts.

Host (First Form)

On a host, `in.rdisc` listens on the `ALL_HOSTS` (224.0.0.1) multicast address for `ROUTER_ADVERTISE` messages from routers. The received messages are handled by first ignoring those listed router addresses with which the host does not share a network. Among the remaining addresses, the ones with the highest preference are selected as default routers and a default route is entered in the kernel routing table for each one of them.

Optionally, `in.rdisc` can avoid waiting for routers to announce themselves by sending out a few `ROUTER_SOLICITATION` messages to the `ALL_ROUTERS` (224.0.0.2) multicast address when it is started.

A timer is associated with each router address. The address will no longer be considered for inclusion in the routing tables if the timer expires before a new *advertise* message is received from the router. The address will also be excluded from consideration if the host receives an *advertise* message with the preference being maximally negative.

Router (Second Form)

When `in.rdisc` is started on a router, it uses the `SIOCGIFCONF` `ioctl`(2) to find the interfaces configured into the system and it starts listening on the `ALL_ROUTERS` multicast address on all the interfaces that support multicast. It sends out `advertise` messages to the `ALL_HOSTS` multicast address advertising all its IP addresses. A few initial *advertise* messages are sent out during the first 30 seconds and after that it will transmit *advertise* messages approximately every 600 seconds.

When `in.rdisc` receives a *solicitation* message, it sends an *advertise* message to the host that sent the *solicitation* message.

When `in.rdisc` is terminated by a signal, it sends out an *advertise* message with the preference being maximally negative.

OPTIONS

–a Accept all routers independent of the preference they have in their *advertise* messages. Normally, `in.rdisc` only accepts (and enters in the kernel routing tables) the router or routers with the highest preference.

–f	Run in.rdisc forever even if no routers are found. Normally, in.rdisc gives up if it has not received any *advertise* message after soliciting three times, in which case it exits with a non-zero exit code. If –f is not specified in the first form then –s must be specified.
–r	Act as a router, rather than a host.
–s	Send three *solicitation* messages initially to quickly discover the routers when the system is booted. When –s is specified, in.rdisc exits with a non-zero exit code if it can not find any routers. This can be overridden with the –f option.
–p *preference*	Set the preference transmitted in the *solicitation* messages. The default is zero .
–T *interval*	Set the interval between transmitting the *advertise* messages. The default time is 600 seconds.

SUMMARY OF TRUSTED SOLARIS CHANGES

in.rdisc must be started from the trusted path. To access /dev/rawip , in.rdisc must be started with an effective UID of 0 , or it must have the file_dac_read and file_dac_write privileges. To modify kernel routing tables, it must inherit the sys_net_config privilege; to open a raw socket, it needs the net_rawaccess privilege; and to send multicast or broadcast packets, it needs the net_broadcast privilege.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

in.routed(1M)

SunOS 5.7 Reference Manual

ioctl(2) , attributes(5) , icmp(7P) , inet(7P)

Deering, S.E., editor, *ICMP Router Discovery Messages* , RFC 1256, Network Information Center, SRI International, Menlo Park, California, September 1991.

NAME

in.rexecd, rexecd – Remote execution server

SYNOPSIS

in.rexecd

DESCRIPTION

`in.rexecd` is the server for the `rexec`(3N) routine. The server provides remote execution facilities with authentication based on user names and passwords. It is invoked automatically as needed by `inetd`(1M) , and then executes the following protocol:

1) The server reads characters from the socket up to a null (`\\0`) byte. The resultant string is interpreted as an ASCII number, base 10.

2) If the number received in step 1 is non-zero, it is interpreted as the port number of a secondary stream to be used for the `stderr` . A second connection is then created to the specified port on the client's machine.

3) A null terminated user name of at most 16 characters is retrieved on the initial socket.

4) A null terminated password of at most 16 characters is retrieved on the initial socket.

5) A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.

6) `rexecd` then validates the user as is done at login time and, if the authentication was successful, changes to the user's home directory, and establishes the user and group protections of the user. Access is denied unless the user has the remote login authorization. If the `/etc/nologin` file exists, access is denied. If any of these steps fail the connection is aborted and a diagnostic message is returned.

7) A null byte is returned on the connection associated with the `stderr` and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rexecd` .

SUMMARY OF TRUSTED SOLARIS CHANGES

Login is not allowed unless the user has the `remote login` authorization. If the `/etc/nologin` file exists, the user is not allowed to log in.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`rexec`(3N) , `attributes`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr` , after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

`username too long`	The name is longer than 16 characters.
`password too long`	The password is longer than 16 characters.
`command too long`	The command line passed exceeds the size of the argument list (as configured into the system).
`Login incorrect`	No password file entry for the user name existed.
`Password incorrect`	The wrong password was supplied.
`No remote directory`	The `chdir` command to the home directory failed.
`/usr/bin/sh: ...`	The user's login shell could not be started.

NAME

in.rlogind, rlogind – Remote login server

SYNOPSIS

/usr/sbin/in.rlogind –U –T

DESCRIPTION

`in.rlogind` is the server for the `rlogin`(1) program. The server provides a remote login facility with authentication based on privileged port numbers.

`in.rlogind` is invoked by `inetd`(1M) when a remote login connection is established, and executes the following protocol:

- The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection.
- The server checks the client's source address. If an entry for the client exists in both `/etc/hosts` and `/etc/hosts.equiv` , a user logging in from the client is not prompted for a password. If the address is associated with a host for which no corresponding entry exists in `/etc/hosts` , the user is prompted for a password, regardless of whether an entry for the client is present in `/etc/hosts.equiv` . See `hosts`(4) and `hosts.equiv`(4) .

Once the source port and address have been checked, `in.rlogind` allocates a pseudo-terminal and manipulates file descriptors so that the slave half of the pseudo-terminal becomes the `stdin` , `stdout` , and `stderr` for a login process. The login process is an instance of the `login`(1) program, invoked with the –`r` .

The login process then proceeds with the `in.rshd`(1M) authentication process.

The –`U` option is used to pass the UID of the client to `login`(1) . The –`T` option is used if the client has the trusted path attribute.

The parent of the login process manipulates the master side of the pseudo-terminal, operating as an intermediary between the login process and the client instance of the `rlogin` program. In normal operation, a packet protocol is invoked to provide Ctrl-S and Ctrl-Q type facilities and propagate interrupt signals to the remote programs. The login process propagates the client terminal's baud rate and terminal type, as found in the environment variable, `TERM` ; see `environ`(4) .

SUMMARY OF TRUSTED SOLARIS CHANGES

Two new options (–`U` and –`T`) are used in the call to `login`(1) .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`login`(1) , `in.rshd`(1M) , `inetd`(1M) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`rlogin`(1) , `environ`(4) , `hosts`(4) , `hosts.equiv`(4) , `attributes`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr` , after which any network connections are closed. An error is indicated by a leading byte with a value of 1.

`Hostname for your address unknown.`	No entry in the host name database existed for the client's machine.
`Try again.`	A *fork* by the server failed.
`/usr/bin/sh: ...`	The user's login shell could not be started.

NOTES

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but it is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

NAME

in.routed, routed – Network routing daemon

SYNOPSIS

/usr/sbin/in.routed [–s] [–q] [–t] [–g] [–S] [–v] [*logfile*]

DESCRIPTION

`in.routed` is invoked at boot time to manage the network routing tables. The routing daemon uses a variant of the Xerox NS Routing Information Protocol in maintaining up-to-date kernel routing table entries.

In normal operation, `in.routed` listens on `udp`(7P) socket 520 (decimal) for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

When `in.routed` is started, it uses the `SIOCGIFCONF` `ioctl`(2) to find those directly connected interfaces configured into the system and marked "up" (the software loopback interface is ignored). If multiple interfaces are present, it is assumed the host will forward packets between networks. `in.routed` then transmits a *request* packet on each interface (using a broadcast packet if the interface supports it) and enters a loop, listening for *request* and *response* packets from other hosts.

For trusted routing, extended security attributes must be associated with a route along with the simple metric that indicates the number of hops to the destination. The additional security routing information (SRI) includes a sensitivity label range, and can include a CIPSO domain of interpretation, a RIPSO label, and a RIPSO error, and some additional keywords: `ripso_only` , `cipso_only` , and `msix_only` . The SRI combined with the simple metric is called the extended metric, or emetric.

For Trusted Solaris 7 systems, two additional types of packets are exchanged. The first one is *sec_response* , which is like the *response* packet but also carries the SRI for the routes. Similar to the *response* packet, the *sec_response* packet propagates a route while adjusting its metric and SRI one hop at a time. The SRI that is carried in *sec_response* packets cannot be propagated through non-Trusted Solaris gateways.

The second additional packet type is *sec_t_response* , which has the exact format as *sec_response* but with a different command number. The *sec_t_response* packets are used for tunneling. Every time a *response* is sent, a *sec_response* and a *sec_t_response* packet are also sent.

Tunneling can be set up for trusted routing between Trusted Solaris 7 gateways when non-Trusted Solaris gateways exist between the Trusted Solaris 7 gateways. For tunneling to work, all Trusted Solaris gateways must be running Trusted Solaris 2.5.1 or 7, and they must be using the extended `in.routed`(1M) for dynamic routing. Also, the non-Trusted Solaris gateways must be using the standard `in.routed`(1M) for dynamic routing. All gateways must be in the same Intranet. To forward SRI s through non-Trusted Solaris gateways to a target

(sub)network, a Trusted Solaris system sends an unlabeled *sec_t_response* packet in a (sub)network directed broadcast to the target (sub)network on behalf of the non-Trusted Solaris gateway connected to that (sub)network. Trusted Solaris systems on the (sub)network can use the SRI to configure their routing tables correctly, and Trusted Solaris 7 gateways on that (sub)network can propagate the SRI to other (sub)networks. A machine that does tunneling is called the forwarding machine; any Trusted Solaris gateway can be a forwarding machine.

Tunneling is enabled by the existence of the file `/etc/security/tsol/tunnel` , and the target (sub)network addresses are obtained from this file. A Trusted Solaris gateway can be responsible for tunneling to more than one (sub)network. The file is composed of a series of lines, each in the following format:

broadcast_addr

A Trusted Solaris gateway can be responsible for tunneling to more than one (sub)network.

A Trusted Solaris system ignores a *response* packet if it is sent by another Trusted Solaris gateway, because in this case, *sec_response* packets should be used in place of *response* packets. A Trusted Solaris system processes a *response* packet if it is sent by a non-Trusted Solaris gateway. If tunneling is done on behalf of that non-Trusted Solaris gateway, it will process both the *response* packets sent by the non-Trusted Solaris gateway and the *sec_response* packets sent by a remote Trusted Solaris gateway on behalf of the non-Trusted Solaris gateway.

When a *request* packet is received, `in.routed` formulates a reply based on the information maintained in its internal tables. The *response* packet contains a list of known routes, each marked with a “hop count” metric (a count of 16, or greater, is considered “infinite”). The metric associated with each route returned, provides a metric relative to the sender.

sec_response and *sec_t_response* packets are formulated by `AND` ing the emetric of the route with the emetric derived from the outgoing interface. Before the *response* packet is sent, a *sec_response* and a *sec_t_response* packet are sent to the same destination with the same metric and additional SRI .

response , *sec_response* , and *request* packets received by `in.routed` are used to update the routing tables if one of the following conditions is satisfied:

- No routing table entry exists for the destination network or host, and the metric indicates the destination is “reachable” (that is, the hop count is not infinite). For *sec_response* and *sec-t_response* packets, a destination is also unreachable if its SRI restricts all possible packets.

- The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed. The only exception occurs when `in.routed` is supposed to process both the *response* packet from a non-Trusted Solaris gateway and the *sec_response* packet tunneled on behalf of that non-Trusted Solaris gateway. In this situation, if both packets carry routing information for the same route, the SRI from the tunneled *sec_response* packet and the metric from the *response* packet are used.
- The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost effective as the current route.
- The new route describes a shorter route to the destination than the one currently stored in the routing tables; the metric of the new route is compared against the one stored in the table to decide this.

For *sec_response* and *sec_t_response* packets, the last rule above is changed to compare the SRI s as well as the metrics. One route is better than another if (a) its metric is smaller; and (b) its SRI is more relaxed than or equal to that of the other. Note that when comparing the SRI s of two routes, one route cannot always serve as a substitute for the other. For example, if the SRI s of two routes have different sensitivity labels, one SRI cannot be said to be more restrictive, because they restrict different sensitivity label ranges.

If two routes cannot be compared, both routes are kept in the routing table, because they represent two routes to the same destination although with different security characteristics; and both routes are needed.

When an update is applied, `in.routed` records the change in its internal tables and generates a *sec_response* packet and a *response* packet to all directly connected hosts and networks. `in.routed` waits a short period of time (no more than 30 seconds) before modifying the kernel's routing tables to allow possible unstable situations to settle.

In addition to processing incoming packets, `in.routed` also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks.

In addition to the facilities described above, `in.routed` supports the notion of "distant" `passive` and `active` gateways. When `in.routed` is started up, it reads the file `gateways` to find gateways which may not be identified using the

SIOCGIFCONF ioctl . Gateways specified in this manner should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange routing information (that is, they should have a in.routed process running on the machine). Passive gateways are maintained in the routing tables forever. Information regarding their existence is not included in any routing information transmitted. Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway and if no routing information is received for a period of time, the associated route is deleted.

The gateways is comprised of a series of lines, each in the following format:

< net | host> *filename1* gateway *filename2* metric *value* < passive | active >

The net or host keyword indicates if the route is to a network or specific host.

filename1 is the name of the destination network or host. This may be a symbolic name located in networks or hosts , or an Internet address specified in "dot" notation; see inet(3N) .

filename2 is the name or address of the gateway to which messages should be forwarded.

value is a metric indicating the hop count to the destination host or network.

The keyword passive or active indicates if the gateway should be treated as passive or active (as described above).

For both the passive and active gateway, the SRI s of their routes are obtained initially from their remote host template. For an active gateway, further routing information will be exchanged with this machine. If later a *sec_response* packet is received from the active gateway or a *sec_t_response* tunneled on its behalf is received, the initial SRI will be updated. If no *sec_response* packet is ever received for this active gateway, use of the initial SRI is continued. For a passive gateway, no further routing information will be exchanged; therefore, the initial SRI is continuously used.

in.routed must be started from the Trusted path at ADMIN_HIGH . It must inherit the net_mac_read , net_privaddr , net_broadcast , and sys_net_config privileges. If a log file is specified, in.routed must also inherit the file_mac_write privilege.

OPTIONS

–g Is used on internetwork routers to offer a route to the "default" destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers.

–q Is the opposite of the –s option.

–s Forces `in.routed` to supply routing information whether it is acting as an internetwork router or not.

–S If `in.routed` is not acting as an internetwork router it will, instead of entering the whole routing table in the kernel, only enter a default route for each internetwork router. This reduces the the memory requirements without losing any routing reliability.

–t All packets sent or received are printed on standard output. In addition, `in.routed` will not divorce itself from the controlling terminal so that interrupts from the keyboard will kill the process. Any other argument supplied is interpreted as the name of the file in which `in.routed` 's actions should be logged. This log contains information about any changes to the routing tables and a history of recent messages sent and received which are related to the changed route.

–v Allows a logfile (whose name must be supplied) to be created showing the changes made to the routing tables with a timestamp.

FILES

`/etc/gateways`	For distant gateways
`/etc/networks`	Associations of Internet Protocol network numbers with network names
`/etc/hosts`	Internet host table
`/etc/security/tsolgateways`	For trusted routing through listed gateways
`/etc/security/tsol/tunnel`	Tunneling information table for Trusted Solaris hosts

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.routed` should be started at `ADMIN_HIGH` . It must inherit the `net_mac_read` , `net_privaddr` , `net_broadcast` , and `sys_net_config` privileges. If a log file is specified, `in.routed` must also inherit the `file_mac_write` privilege. Because trusted routing considers the security of the route along with the route's metric when making routing decisions, `in.routed` sends two additional types of response packets containing security information for routes: *sec_response* packets for communications with connected Trusted Solaris gateways, and *sec_t_response* packets for tunneling to Trusted Solaris gateways on the other side of non-Trusted Solaris gateways.

SEE ALSO

Trusted Solaris 7 Reference Manual

`route`(1M)

SunOS 5.7 Reference Manual

`ioctl`(2) , `inet`(3N) , `attributes`(5) , `inet`(7P) , `udp`(7P)

NOTES

The kernel's routing tables may not correspond to those of `in.routed` for short periods of time while processes that utilize existing routes exit; the only remedy for this is to place the routing process in the kernel.

`in.routed` should listen to intelligent interfaces, such as an IMP , and to error protocols, such as ICMP , to gather more information.

`in.routed` initially obtains a routing table by examining the interfaces configured on a machine and the `gateways` file. It then sends a request on all directly connected networks for more routing information. `in.routed` does not recognize or use any routing information already established on the machine prior to startup. With the exception of interface changes, `in.routed` does not see any routing table changes that have been done by other programs on the machine, for example, routes added, deleted or flushed by way of the `route`(1M) command. Therefore, these types of changes should not be done while `in.routed` is running. Rather, shut down `in.routed` , make the changes required, and then restart `in.routed` .

NAME

in.rshd, rshd – Remote shell server

SYNOPSIS

in.rshd *host.port*

DESCRIPTION

`in.rshd` is the server for the `rsh`(1) program. The server provides remote execution facilities with authentication based on privileged port numbers.

`in.rshd` is invoked by `inetd`(1M) each time a shell service is requested, and executes the following protocol:

1. The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection. The client's host address (in hex) and port number (in decimal) are the arguments passed to `in.rshd` .
2. The server reads characters from the socket up to a null (`0`) byte. The resultant string is interpreted as an ASCII number, base 10.
3. If the number received in step 1 is non-zero, it is interpreted as the port number of a secondary stream to be used for the `stderr` . A second connection is then created to the specified port on the client's machine. The source port of this second connection is also in the range 0-1023.
4. The server checks the client's source address. If the address is associated with a host for which no corresponding entry exists in the host name data base (see `hosts`(4)), the server aborts the connection. Please refer to the `SECURITY` section below for more details.
5. A null terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as a user identity to use on the *server* 's machine.
6. A null terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as the user identity on the *client* 's machine.
7. A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
8. `in.rshd` checks whether logins are currently allowed by looking for an `/etc/nologin` file. If the file exists, the connection is terminated. If logins are allowed, the user is validated according to the following steps. The remote user name is looked up in the password file and a `chdir` is performed to the user's home directory. If the lookup fails, the connection is terminated. If the `chdir` fails, it does a `chdir` to `/` (root). If the user is not the superuser, (user ID 0), and if the pam_rhosts_auth PAM module is configured for authentication, the file `/etc/hosts.equiv` is consulted for a list of hosts considered "equivalent". If the client's host name is present in this file, the authentication is considered successful. See SECURITY below for a discussion of PAM authentication.

If the lookup fails, or the user is the superuser, then the file `.rhosts` in the home directory of the remote user is checked for the machine name and identity of the user on the client's machine. If this lookup fails, the connection is terminated

9. A null byte is returned on the connection associated with the `stderr` and the command line is passed to the normal login shell of the user. (The PATH variable is set to `/usr/bin` .) The shell inherits the network connections established by `in.rshd` .

SECURITY

`in.rshd` uses `pam`(3) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf` , specifies the modules to be used for `in.rshd` . Here is a partial `pam.conf` file with entries for the `rsh` command using rhosts authentication, UNIX account management, and session management module.

rsh	auth	required	/usr/lib/security/pam_rhosts_auth.so.1
rsh	account	required	/usr/lib/security/pam_unix.so.1
rsh	session	required	/usr/lib/security/pam_unix.so.1

If there are no entries for the `rsh` service, then the entries for the "other" service will be used. To maintain the authentication requirement for `in.rshd` , the rsh entry must always be configured with the pam_rhosts_auth.so.1 module. Multiple authentication modules can not be listed for the `rsh` service.

SUMMARY OF TRUSTED SOLARIS CHANGES

If the `/etc/nologin` file exists, the server will not allow connections. The values of the trusted path, label view, and label-translation process attributes from the client process are propagated to the remote shell.

FILES

`/etc/hosts.equiv`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`rsh`(1) , `hosts`(4) , `attributes`(5)

DIAGNOSTICS

The following diagnostic messages are returned on the connection associated with `stderr` , after which any network connections are closed. An error is indicated by a leading byte with a value of 1 in step 9 above (0 is returned above upon successful completion of all the steps prior to the command execution).

`locuser too long`	The name of the user on the client's machine is longer than 16 characters.
`remuser too long`	The name of the user on the remote machine is longer than 16 characters.
`command too long`	The command line passed exceeds the size of the argument list (as configured into the system).
`Hostname for your address unknown.`	No entry in the host name database existed for the client's machine.
`Login incorrect.`	No password file entry for the user name existed.
`Permission denied.`	The authentication procedure described above failed.
`Can't make pipe.`	The pipe needed for the `stderr` was not created.
`Try again.`	A `fork` by the server failed.

NOTES

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but it is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

NAME

install – Install commands

SYNOPSIS

/usr/sbin/install –c *dira* [–m *mode*] [–u *user*] [–g *group*] [–o] [–s] *file*

/usr/sbin/install –f *dirb* [–m *mode*] [–u *user*] [–g *group*] [–o] [–s] *file*

/usr/sbin/install –n *dirc* [–m *mode*] [–u *user*] [–g *group*] [–o] [–s] *file*

/usr/sbin/install –d | –i [–m *mode*] [–u *user*] [–g *group*] [–o] [–s] *dirx*…

/usr/sbin/install [–m *mode*] [–u *user*] [–g *group*] [–o] [–s] *file* [*dirx*…]

DESCRIPTION

`install` is most commonly used in "makefiles" (see `make`(1S)) to install a *file* in specific locations, or to create directories within a file system. Each *file* is installed by copying it into the appropriate directory.

`install` uses no special privileges to copy files from one place to another. The implications of this are:

- You must have permission to read the files to be installed.
- You must have permission to copy into the destination directory.
- You must have permission to change the modes on the final copy of the file if you want to use the –m option.
- You must assume an administrative role if you want to specify the ownership of the installed file with the –u or –g options. If you are not in an administrative role, the installed file will be owned by you, regardless of who owns the original.

`install` prints messages telling the user exactly what files it is replacing or creating and where they are going.

If no options or directories (*dirx* . . .) are given, `install` searches a set of default directories (`/bin`, `/usr/bin`, `/etc`, `/lib`, and `/usr/lib`, in that order) for a file with the same name as `file`. When the first occurrence is found, `install` issues a message saying that it is overwriting that file with *file*, and proceeds to do so. If the file is not found, the program states this and exits.

If one or more directories (*dirx*…) are specified after *file*, those directories are searched before the default directories.

OPTIONS

–c *dira* — Install *file* in the directory specified by *dira*, if *file* does not yet exist. If it is found, `install` issues a message saying that the file already exists, and exits without overwriting it.

–f *dirb* — Force `file` to be installed in given directory, even if the file already exists. If the file being installed does not already exist, the mode and owner of the new file will be set to `755` and `bin`, respectively. If the file already exists, the mode and owner will be that of the already existing file.

–n *dirc*	If `file` is not found in any of the searched directories, it is put in the directory specified in *dirc*. The mode and owner of the new file will be set to `755` and `bin`, respectively.
–d	Create a directory. Missing parent directories are created as required as in `mkdir –p`. If the directory already exists, the owner, group and mode will be set to the values given on the command line.
–i	Ignore default directory list, searching only through the given directories (*dirx*…).
–m *mode*	The mode of the new file is set to *mode*. Set to `0755` by default.
–u *user*	The owner of the new file is set to *user*. Only available to administrative roles. Set to `bin` by default.
–g *group*	The group id of the new file is set to *group*. Only available to administrative roles. Set to `bin` by default.
–o	If *file* is found, save the "found" file by copying it to `OLD`*file* in the directory in which it was found. This option is useful when installing a frequently used file such as `/bin/sh` or `/lib/saf/ttymon`, where the existing file cannot be removed.
–s	Suppress printing of messages other than error messages.

USAGE

See `largefile`(5) for the description of the behavior of `install` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To specify the ownership of an installed file with the –u or –g options, you must assume an administrative role.

SEE ALSO

Trusted Solaris 7 Reference Manual

`chgrp`(1), `chmod`(1), `chown`(1), `mkdir`(1)

SunOS 5.7 Reference Manual

`cp`(1), `make`(1S), `chown`(1M), `attributes`(5), `largefile`(5)

NAME

in.tftpd, tftpd – Internet Trivial File Transfer Protocol server

SYNOPSIS

in.tftpd [–s] [*homedir*]

DESCRIPTION

`tftpd` is a server that supports the Internet Trivial File Transfer Protocol (TFTP). This server is normally started by `inetd`(1M) and operates at the port indicated in the `tftp` Internet service description in the `/etc/inetd.conf` file. By default, the entry for `in.tftpd` in `etc/inetd.conf` is commented out. To make `in.tftpd` operational, the comment character(s) must be deleted from the file. See `inetd.conf`(4) .

Before responding to a request, the server attempts to change its current directory to *homedir* ; the default directory is `/tftpboot` .

The use of `tftp` does not require an account or password on the remote system. Due to the lack of authentication information, `in.tftpd` will allow only publicly readable files to be accessed. Files may be written only if they already exist and are publicly writable. Note that this extends the concept of “public” to include all users on all hosts that can be reached through the network; this may not be appropriate on all systems, and its implications should be considered before enabling this service.

`in.tftpd` runs with the user ID and group ID set to `[GU]ID_NOBODY` under the assumption that no files exist with that owner or group. However, nothing checks this assumption or enforces this restriction.

OPTIONS

–s Secure. When specified, the directory change to *homedir* must succeed. The daemon also changes its root directory to *homedir* .

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.tftpd` should be started from the trusted path with a UID of `0` ; it must inherit the `proc_chroot` , `proc_owner` , and `proc_setid` privileges.

FILES

`/etc/inetd.conf` Configuration file for inetd.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M)

SunOS 5.7 Reference Manual

`tftp`(1) , `netconfig`(4) , `attributes`(5)

Sollins, K.R., *The TFTP Protocol (Revision 2)* , RFC 783, Network Information Center, SRI International, Menlo Park, California, June 1981.

NAME

list_devices – List allocatable devices

SYNOPSIS

list_devices [–s] [–U uid] –l [*device*]

list_devices [–s] [–U uid] –n [*device*]

list_devices [–s] [–U uid] –u [*device*]

DESCRIPTION

`list_devices` lists the allocatable devices in the system according to specified qualifications.

The *device* and all device special files associated with the device are listed. The device argument is optional and if it is not present, all relevant devices are listed.

OPTIONS

–l [*device*]	List the pathname(s) of the device special files associated with the device that are allocatable to the current process. If *device* is given, list only the files associated with the specified device.
–n [*device*]	List the pathname(s) of device special files associated with the device that are allocatable to the current process but are not currently allocated. If *device* is given, list only the files associated with that device.
–s	Silent. Suppresses any diagnostic output.
–u [*device*]	List the pathname(s) of device special files, associated with the device that are allocated to the owner of the current process. If *device* is given, list only the files associated with that device.
–U *uid*	Use the user ID *uid* instead of the real user ID of the current process when performing the `list_devices` operation. This option requires `proc_setid` privilege to be asserted.

DIAGNOSTICS

`list_devices` returns an nonzero exit status in the event of an error.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –U option requires the `proc_setid` privilege.

FILES

`/etc/security/device_allocate`	Mandatory access control file for devices.
`/etc/security/device_maps`	List of physical devices associated with a device name and type.
`/etc/security/dev/*`	Device storage area.
`/usr/security/lib/*`	Directory of device cleaning scripts.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M), `deallocate`(1M), `device_allocate`(4), `device_maps`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

lockd – Network lock daemon

SYNOPSIS

/usr/lib/nfs/lockd [–g *graceperiod*] [–t *timeout*] [*nthreads*]

DESCRIPTION

The `lockd` utility is part of the NFS lock manager, which supports record locking operations on NFS files. See `fcntl`(2) and `lockf`(3C). The lock manager provides two functions:

- it forwards `fcntl`(2) locking requests for NFS mounted file systems to the lock manager on the NFS server
- it generates local file locking operations in response to requests forwarded from lock managers running on NFS client machines.

State information kept by the lock manager about these locking requests can be lost if the `lockd` is killed or the operating system is rebooted. Some of this information can be recovered as follows. When the server lock manager restarts, it waits for a grace period for all client-site lock managers to submit reclaim requests. Client-site lock managers, on the other hand, are notified by the status monitor daemon, `statd`(1M) , of the restart and promptly resubmit previously granted lock requests. If the lock daemon fails to secure a previously granted lock at the server site, then it sends `SIGLOST` to a process.

OPTIONS

–g *graceperiod*	Specify the number of seconds that clients have to reclaim locks after the server reboots. The default is 45 seconds.
–t *timeout*	Specify the number of seconds to wait before retransmitting a lock request to the remote server. The default value is 15 seconds.
nthreads	Specify the maximum number of concurrent threads that the server can handle. This concurrency is achieved by up to *nthreads* threads created as needed in the kernel. *nthreads* should be based on the load expected on this server. If *nthreads* is not specified, the maximum number of concurrent threads will default to 20.

SUMMARY OF TRUSTED SOLARIS CHANGES

`lockd` must be started with a UID of `0`, a sensitivity label of `ADMIN_LOW`, and a clearance of `ADMIN_HIGH`. It must be started from the trusted path and must have these privileges: `net_mac_read`, `net_privaddr`, `net_upgrade_sl`, and `sys_nfs`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`statd`(1M), `fcntl`(2)

SunOS 5.7 Reference Manual

`lockf`(3C), `attributes`(5)

NAME

lpadmin – Configure the LP print service

SYNOPSIS

lpadmin –p *printer options*

lpadmin –x *dest*

lpadmin –d [*dest*]

lpadmin –S *print-wheel* –A *alert-type* [–W *minutes*] [–Q *requests*]

lpadmin –M –f *form-name* [–a [–o filebreak] [–t *tray-number*]]

DESCRIPTION

`lpadmin` configures the LP print service by defining printers and devices. It is used to add and change printers, to remove printers from service, to set or change the system default destination, to define alerts for printer faults, and to mount print wheels.

OPTIONS

Adding or Changing a Printer

The first form of the `lpadmin` command (`lpadmin` –p *printer options*) is used to configure a new printer or to change the configuration of an existing printer. When creating a new printer, one of three options (–v, –U, or –s) must be supplied. In addition, only one of the following may be supplied: –e, –i, or –m; if none of these three options is supplied, the model standard is used. The –h and –l options are mutually exclusive. Printer and class names may be no longer than 14 characters and must consist entirely of the characters A-Z, a-z, 0-9, dash (-) and underscore (_). If –s is specified, the following options are invalid: –A, –e, –F, –h, –i, –l, –M, –m, –o, –U, –v, and –W.

The following printer options may appear in any order.

–A *alert-type* [–W *minutes*]

The –A option is used to define an alert that informs the administrator when a printer fault is detected, and periodically thereafter, until the printer fault is cleared by the administrator. The *alert-types* are:

`mail`

Send the alert message using mail (see `mail`(1)) to the administrator.

`write`

Write the message to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is chosen arbitrarily.

`quiet`

Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the fault has been cleared and printing resumes, messages will again be sent when another fault occurs with the printer.

`showfault`

Attempt to execute a fault handler on each system that has a print job in the queue. The fault handler is `/etc/lp/alerts/printer`. It is invoked with three parameters: *printer_name*, *date*, and *file_name*. The *file_name* is the name of a file containing the fault message.

`none`

Do not send messages; any existing alert definition for the printer will be removed. No alert will be sent when the printer faults until a different alert-type (except `quiet`) is used.

`shell-command`

Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blank spaces embedded in the command, enclose the command in quotes. Note that the `mail` and `write` values for this option are equivalent to the values `mail` *user-name* and `write` *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another user ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list`

Display the type of the alert for the printer fault. No change is made to the alert.

The message sent appears as follows:

```
The printer printer has stopped printing for the reason given below.
Fix the problem and bring the printer back on line.
Printing has stopped, but will be restarted in a few minutes;
issue an enable command if you want to restart sooner.
```

Unless someone issues the change request:

```
lp –i request-id –P ...
```

to change the page list to print, the current request will be reprinted from the beginning. The reason(s) it stopped (multiple reasons indicate reprinted attempts):*reason*

The LP print service can detect printer faults only through an adequate fast filter and only when the standard interface program or a suitable customized interface program is used. Furthermore, the level of recovery after a fault depends on the capabilities of the filter.

If the *printer* is `all`, the alerting defined in this command applies to all existing printers.

If the –`W` option is not used to arrange fault alerting for *printer*, the default procedure is to mail one message to the administrator of *printer* per fault. This is equivalent to specifying –`W once` or –`W 0`. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

–`c` *class*
Insert *printer* into the specified *class*. *class* will be created if it does not already exist.

–`D` *comment*
Save this *comment* for display whenever a user asks for a full description of *printer* `lpstat`(1)). The LP print service does not interpret this comment.

–`e` *printer*
Copy the interface program of an existing *printer* to be the interface program for *printer*. (Options –`i` and –`m` may not be specified with this option.)

–`F` *fault-recovery*
This option specifies the recovery to be used for any print request that is stopped because of a printer fault, according to the value of *fault-recovery*:

`continue`	Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing.
`beginning`	Start printing the request again from the beginning.
`wait`	Disable printing on *printer* and wait for the administrator or a user to enable printing again. During the wait, the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. (See the –`i` option of the `lp` command.) If no change request is made before printing is enabled, printing resumes at the top of the page where stopped, if the filter allows; otherwise, the request is printed from the beginning.

–`f allow:`*form-list*
–`f deny:`*form-list*
Allow or deny the forms in *form-list* to be printed on *printer*. By default no forms are allowed on a new printer.

For each printer, the LP print service keeps two lists of forms: an “allow-list” of forms that may be used with the printer, and a “deny-list” of forms that

may not be used with the printer. With the –f allow option, the forms listed are added to the allow-list and removed from the deny-list. With the –f deny option, the forms listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the forms in the list may be used on the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the forms in the deny-list may not be used with the printer. All forms can be excluded from a printer by specifying –f deny:all. All forms can be used on a printer (provided the printer can handle all the characteristics of each form) by specifying –f allow:all.

The LP print service uses this information as a set of guidelines for determining where a form can be mounted. Administrators, however, are not restricted from mounting a form on any printer. If mounting a form on a particular printer is in disagreement with the information in the allow-list or deny-list, the administrator is warned but the mount is accepted. Nonetheless, if a user attempts to issue a print or change request for a form and printer combination that is in disagreement with the information, the request is accepted only if the form is currently mounted on the printer. If the form is later unmounted before the request can print, the request is canceled and the user is notified by mail.

If the administrator tries to specify a form as acceptable for use on a printer that doesn't have the capabilities needed by the form, the command is rejected.

Note the other use of –f, with the –M option, below.

The –T option must be invoked first with lpadmin to identify the printer type before the –f option can be used.

–h

Indicate that the device associated with the printer is hardwired. If neither of the mutually exclusive options, –h and –l, is specified, –h is assumed.

–I *content-type-list*

Allow *printer* to handle print requests with the content types listed in a *content-type-list*. If the list includes names of more than one type, the names must be separated by commas or blank spaces. (If they are separated by blank spaces, the entire list must be enclosed in double quotes.)

The type simple is recognized as the default content type for files in the UNIX system. A simple type of file is a data stream containing only printable ASCII characters and the following control characters.

```
Control Character           Octal Value Meaning
BACKSPACE       10          move back one character,
                            except at beginning of line
```

```
TAB             11          move to next tab stop
LINEFEED (newline) 12       move to beginning of next line
FORMFEED        14          move to beginning of next page
RETURN          15          move to beginning of current line
```

To prevent the print service from considering `simple` a valid type for the printer, specify either an explicit value (such as the printer type) in the *content-type-list*, or an empty list. If you do want `simple` included along with other types, you must include `simple` in the *content-type-list*.

Except for `simple`, each *content-type* name is freely determined by the administrator. If the printer type is specified by the –`T` option, then the printer type is implicitly considered to be also a valid content type.

–`i` *interface*
Establish a new interface program for *printer*. *interface* is the pathname of the new program. (The –`e` and –`m` options may not be specified with this option.)

–`l`
Indicate that the device associated with *printer* is a login terminal. The LP scheduler (`lpsched`) disables all login terminals automatically each time it is started. (The –`h` option may not be specified with this option.)

–`M` –`f` *form-name* [–`a` [–`o filebreak`]] [–`t` *tray-number*]
Mount the form *form-name* on *printer*. Print requests that need the pre-printed form *form-name* will be printed on *printer*. If more than one printer has the form mounted and the user has specified `any` (with the –`d` option of the `lp` command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

The page length and width, and character and line pitches needed by the form are compared with those allowed for the printer, by checking the capabilities in the `terminfo` database for the type of printer. If the form requires attributes that are not available with the printer, the administrator is warned but the mount is accepted. If the form lists a print wheel as mandatory, but the print wheel mounted on the printer is different, the administrator is also warned but the mount is accepted.

If the –`a` option is given, an alignment pattern is printed, preceded by the same initialization of the physical printer that precedes a normal print request, with one exception: no banner page is printed. Printing is assumed to start at the top of the first page of the form. After the pattern is printed, the administrator can adjust the mounted form in the printer and press return for another alignment pattern (no initialization this time), and can continue printing as many alignment patterns as desired. The administrator can quit the printing of alignment patterns by typing `q`.

If the –o filebreak option is given, a formfeed is inserted between each copy of the alignment pattern. By default, the alignment pattern is assumed to correctly fill a form, so no formfeed is added.

If the –t *tray-number* option is specified, printer tray *tray-number* will used.

A form is "unmounted" either by mounting a new form in its place or by using the –f none option. By default, a new printer has no form mounted.

Note the other use of –f without the –M option above.

–M –S *print-wheel*
Mount the *print-wheel* on *printer*. Print requests that need the *print-wheel* will be printed on *printer*. If more than one printer has *print-wheel* mounted and the user has specified any (with the –d option of the lp command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

If the *print-wheel* is not listed as acceptable for the printer, the administrator is warned but the mount is accepted. If the printer does not take print wheels, the command is rejected.

A print wheel is "unmounted" either by mounting a new print wheel in its place or by using the option –S none. By default, a new printer has no print wheel mounted.

Note the other uses of the –S option without the –M option described below.

–m *model*
Select *model* interface program, provided with the LP print service, for the printer. (Options –e and –i may not be specified with this option.)

–o *option*
The –o option defines default printer configuration values given to an interface program. The default may be explicitly overwritten for individual requests by the user (see lp(1)), or taken from a preprinted form description (see lpforms(1M) and lp(1)).

There are several options which are pre-defined by the system. In addition, any number of key-value pairs may be defined. Each of the predefined and undefined options are described.

The Predefined Options

The following options are predefined: adjusting printer capabilities, adjusting printer port characteristics, configuring network printers, and controlling the use of banner.

Adjusting Printer Capabilities

length=*scaled-decimal-number*
width=*scaled-decimal-number*
cpi=*scaled-decimal-number*
lpi=*scaled-decimal-number*

The term *scaled-decimal-number* refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a "trailing" letter attached to the number. Three types of *scaled-decimal-numbers* can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing c); numbers that show sizes in inches (marked with a trailing i); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch, or characters per inch.

The option values must agree with the capabilities of the type of physical printer, as defined in the terminfo database for the printer type. If they do not, the command is rejected.

The defaults are defined in the terminfo entry for the specified printer type. The defaults may be reset by:

```
lpadmin –p printername –o length=
lpadmin –p printername o width=
lpadmin –p printername o cpi=
lpadmin –p printername o lpi=
```

Adjusting Printer Port Characteristics

stty="'*stty-option-list*'"

The *stty-option-list* is not checked for allowed values, but is passed directly to the stty program by the standard interface program. Any error messages produced by stty when a request is processed (by the standard interface program) are mailed to the user submitting the request.

The default for stty is:

```
stty="'9600 cs8 –cstopb –parenb ixon
     –ixany opost –olcuc onlcr
     –ocrnl –onocr
     –onlret –ofill nl0 cr0 tab0 bs0 vt0 ff0'"
```

The default may be reset by:

```
lpadmin –p printername –o stty=
```

Configuring Network Printers

dest=*string*
protocol=*string*
bsdctrl=*string*
timeout=*non-negative-integer-seconds*

These four options are provided to support network printing. Each option is passed directly to the interface program; any checking for allowed values is done there.

The value of `dest` is the name of the destination for the network printer; the semantics for value `dest` are dependent on the printer and the configuration. There is no default.

The value of option `protocol` sets the over-the-wire protocol to the printer. The default for option `protocol` is `bsd`. The value of option `bsdctrl` sets the print order of control and data files (BSD protocol only); the default for this option is `control file first`. The value of option `timeout` sets the seed value for backoff time when the printer is busy. The default value for the `timeout` option is `10` seconds. The defaults may be reset by:

lpadmin –**p** *printername* –o protocol=
lpadmin –**p** *printername* –o bsdctrl=
lpadmin –**p** *printername* –o timeout=

Controlling the Use of the Banner Page

nobanner

Allow a user to submit a print request specifying that no banner page be printed.

banner

Force a banner page to be printed with every print request, even when a user asks for no banner page. This is the default. Specify –o `nobanner` to allow users to specify –o `nobanner` with the `lp` command.

Undefined Options

key=value

Each *key=value* is passed directly to the interface program. Any checking for allowed values is done in the interface program.

Any default values for a given *key=value* option are defined in the interface program. If a default is provided, it may be reset by typing the key without any value:

lpadmin –**p** *printername* –o *key=*

–P *paper-name*
Specify a paper type list that the printer supports.

–r *class*
Remove *printer* from the specified *class*. If *printer* is the last member of *class*, then *class* will be removed.

–S *list*
Allow either the print wheels or aliases for character sets named in *list* to be used on the printer.

If the printer is a type that takes print wheels, then *list* is a comma or space separated list of print wheel names. (Enclose the list with quotes if it contains blank spaces.) These will be the only print wheels considered mountable on the printer. (You can always force a different print wheel to be mounted.) Until the option is used to specify a list, no print wheels will be considered mountable on the printer, and print requests that ask for a particular print wheel with this printer will be rejected.

If the printer is a type that has selectable character sets, then *list* is a comma or blank separated list of character set name "mappings" or aliases. (Enclose the list with quotes if it contains blank spaces.) Each "mapping" is of the form *known-name=alias* The *known-name* is a character set number preceded by `cs` (such as `cs3` for character set three) or a character set name from the `terminfo` database entry `csnm`. See `terminfo`(4). If this option is not used to specify a list, only the names already known from the `terminfo` database or numbers with a prefix of `cs` will be acceptable for the printer. If *list* is the word `none`, any existing print wheel lists or character set aliases will be removed.

Note the other uses of the –S with the –M option described above.

The –T option must be invoked first with `lpadmin` to identify the printer type before the –S option can be used.

–s *system-name*[!*printer-name*]
Make a remote printer (one that must be accessed through another system) accessible to users on your system. *system-name* is the name of the remote system on which the remote printer is located it. *printer-name* is the name used on the remote system for that printer. For example, if you want to access *printer1* on *system1* and you want it called *printer2* on your system:

–**p** *printer2* –**s** *system1*!*printer1*

–T *printer-type-list*
Identify the printer as being of one or more *printer-type*s. Each *printer-type* is used to extract data from the `terminfo` database; this information is used to initialize the printer before printing each user's request. Some filters may also use a *printer-type* to convert content for the printer. If this option is not used, the default *printer-type* will be `unknown`; no information will be extracted from `terminfo` so each user request will be printed without first initializing the printer. Also, this option must be used if the following are to work: –o `cpi`, –o `lpi`, –o `width`, and –o `length` options of the `lpadmin` and `lp` commands, and the –S and –f options of the `lpadmin` command.

If the *printer-type-list* contains more than one type, then the *content-type-list* of the –I option must either be specified as `simple`, as empty (`–I ""`), or not specified at all.

–t *number-of-trays*
Specify the number of trays when creating the printer.

–u allow:*login-ID-list*
–u deny:*login-ID-list*
Allow or deny the users in *login-ID-list* access to the printer. By default all users are allowed on a new printer. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	a user on any system
system-name!*login-ID*	a user on system *system-name*
system-name!`all`	all users on system *system-name*
`all`!*login-ID*	a user on all systems
`all`	all users on all systems

For each printer, the LP print service keeps two lists of users: an "allow-list" of people allowed to use the printer, and a "deny-list" of people denied access to the printer. With the –u `allow` option, the users listed are added to the allow-list and removed from the deny-list. With the –u `deny` option, the users listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the users in the list may use the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the users in the deny-list may not use the printer. All users can be denied access to the printer by specifying –u `deny:all`. All users may use the printer by specifying –u `allow:all`.

–U *dial-info*
The –U option allows your print service to access a remote printer. (It does not enable your print service to access a remote printer service.) Specifically, –U assigns the "dialing" information *dial-info* to the printer. *dial-info* is used with the `dial` routine to call the printer. Any network connection supported by the Basic Networking Utilities will work. *dial-info* can be either a phone number for a modem connection, or a system name for other kinds of connections. Or, if –U `direct` is given, no dialing will take place, because the name `direct` is reserved for a printer that is directly connected. If a system name is given, it is used to search for connection details from the file `/etc/uucp/Systems` or related files. The Basic Networking Utilities are required to support this option. By default, –U `direct` is assumed.

–v *device*
Associate a *device* with *printer*. *device* is the path name of a file that is writable by `lp`. Note that the same *device* can be associated with more than one printer.

Removing a Printer Destination

The –x *dest* option removes the destination *dest* (a printer or a class), from the LP print service. If *dest* is a printer and is the only member of a class, then the class will be deleted, too. If *dest* is `all`, all printers and classes are removed. No other *options* are allowed with –x.

Setting/Changing the System Default Destination

The –d [*dest*] option makes *dest* (an existing printer or class) the new system default destination. If *dest* is not supplied, then there is no system default destination. No other *options* are allowed with –d.

Setting an Alert for a Print Wheel

–S *print-wheel* –A *alert-type* [–W *minutes*] [–Q *requests*]

The –S *print-wheel* option is used with the –A *alert-type* option to define an alert to mount the print wheel when there are jobs queued for it. If this command is not used to arrange alerting for a print wheel, no alert will be sent for the print wheel. Note the other use of –A, with the –p option, above.

The *alert-types* are:

`mail`	Send the alert message using the `mail` command to the administrator.
`write`	Write the message, using the `write` command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.
`quiet`	Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the *print-wheel* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the –Q option.
`none`	Do not send messages until the –A option is given again with a different *alert-type* (other than `quiet`).
shell-command	Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the

command, enclose the command in quotes. Note that the `mail` and `write` values for this option are equivalent to the values `mail` *user-name* and `write` *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another user ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list` Display the type of the alert for the print wheel on standard output. No change is made to the alert.

The message sent appears as follows:

The print wheel *print-wheel* needs to be mounted on the printer(s): *printer*(*integer1*requests) *integer2* print requests await this print wheel.

The printers listed are those that the administrator had earlier specified were candidates for this print wheel. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the printer list is the total number of requests awaiting the print wheel. It will be less than the sum of the other numbers if some requests can be handled by more than one printer.

If the *print-wheel* is `all`, the alerting defined in this command applies to all print wheels already defined to have an alert.

If the –`W` option is not given, the default procedure is that only one message will be sent per need to mount the print wheel. Not specifying the –`W` option is equivalent to specifying –`W once` or –`W 0`. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

If the –`Q` option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the print wheel are waiting. If the –`Q` option is not given, or *requests* is `1` or `any` (which are both the default), a message is sent as soon as anyone submits a print request for the print wheel when it is not mounted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/var/spool/lp/*` LP print queue.

`/etc/lp` Printing system control files.

`/etc/lp/alerts/printer` Fault handler for `lpadmin`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the `lpadmin` command requires the `administer printing` authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`enable`(1), `lp`(1), `lpstat`(1), `accept`(1M), `lpforms`(1M), `lpsched`(1M), `lpsystem`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`mail`(1), `stty`(1), `dial`(3N), `terminfo`(4), `attributes`(5)

System Administration Guide, Volume I

NAME

lpfilter – Administer filters used with the LP print service

SYNOPSIS

/usr/sbin/lpfilter –f *filter-name* {– | –i | –l | –x | –F *pathname* }

DESCRIPTION

The `lpfilter` command is used to add, change, delete, or list a filter used with the LP print service. These filters convert the content of a file to have a content type acceptable to a printer.

OPTIONS

Arguments consist of the –f *filter-name* option and exactly one of the arguments appearing within braces ({ }) in the SYNOPSIS.

–f *filter-name* — Specifies the *filter-name* of the filter to be added, changed, reset, deleted, or listed. The filter name `all` is a special filter name defined below. The –f option is required.

– — Adds or changes a filter as specified from standard input. The format of the input is specified below. If –f `all` is specified with the – option, the specified change is made to all existing filters. This is not useful.

–F *pathname* — Adds or changes a filter as specified by the contents of the file *pathname*. The format of the file's contents is specified below. If –f `all` is specified with the –F option, the specified change is made to all existing filters. This is not useful.

–i — Resets a filter to its default settings. Using –f `all` with the –i option restores all filters for which predefined settings are available to their original settings.

–x — Deletes a filter. Using –f `all` with the –x option results in all filters being deleted.

–l — Lists a filter description. Using –f `all` with the –l option produces a list of all filters.

USAGE

Adding or Changing a Filter

The filter named in the –f option is added to the filter table. If the filter already exists, its description is changed to reflect the new information in the input.

When – is specified, standard input supplies the filter description. When –F is specified, the file *pathname* supplies the filter description. One of these two options must be specified to add or change a filter.

When an existing filter is changed with the –F or – option, lines in the filter description that are not specified in the new information are not changed. When a new filter is added with this command, unspecified lines receive default values. See below.

Filters are used to convert the content of a request from its initial type into a type acceptable to a printer. For a given print request, the LP print service knows the following:

- The content type of the request (specified by `lp –T` or determined implicitly)
- The name of the printer (specified by `lp –d`)
- The printer type (specified by `lpadmin –T`)

  The printer type is intended to be a printer model, but some people specify it with a content type even though `lpadmin –I` is intended for this purpose.
- The content types acceptable to the printer (specified by `lpadmin –I`)

  The values specified by the `lpadmin –T` are treated as if they were specified by the `–I` option as well.
- The modes of printing asked for by the originator of the request (specified by various options to `lp`)

The system uses the above information to construct a list of one or more filters that converts the document's content type into a content type acceptable to the printer and consumes all `lp` arguments that invoke filters (`–y` and `–P`).

The contents of the file (specified by the `–F` option) and the input stream from standard input (specified by `–`) must consist of a series of lines, such that each line conforms to the syntax specified by one of the seven lines below. All lists are comma or space separated. Each item contains a description.

```
Input types: content-type-list
Output types: content-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: filter-type
Command: shell-command
Options: template-list
```

`Input types` This gives the content types that can be accepted by the filter. The default is `any`. The document content type must be a member of this list for the initial filter in the sequence.

`Output types` This gives the content types that the filter can produce from any of the input (content) types. The default is `any`. The intersection of the output types of this list and the content types acceptable to the printer (from `lpadmin –I` and `lpadmin –T`) must be non-null for the last filter in the sequence. For adjacent filters in the sequence, the intersection of output types of one and the input types of the next must be non-null.

`Printer types` This gives the printer types for which this printer can be used. The LP print service will restrict the use of the filter to these printer types (from `lpadmin –T`). The default is `any`.

`Printers` This gives the names of the printers for which the filter can be used. The LP print service will restrict the use of the filter to just the printers named. The default is `any`.

`Filter type` This marks the filter as a `slow` filter or a `fast` filter. Slow filters are generally those that take a long time to convert their input (that is, minutes or hours). They are run before the job is scheduled for a printer, to keep the printers from being tied up while the filter is running. If a listed printer is on a remote system, the filter type for it must have the value `slow`. That is, if a client defines a filter, it must be a slow filter. Fast filters are generally those that convert their input quickly (that is, faster than the printer can process the data), or those that must be connected to the printer when run. Fast filters will be given to the interface program to run while connected to the physical printer.

`Command` This specifies which program to run to invoke the filter. The full program pathname as well as fixed options must be included in the *shell-command*; additional options are constructed, based on the characteristics of each print request and on the *Options* field. A command must be given for each filter. The command must accept a data stream as standard input and produce the converted data stream on its standard output. This allows filter pipelines to be constructed to convert data not handled by a single filter.

`Options` This is a comma-separated list of templates used by the LP print service to construct options to the filter from the characteristics of each print request listed in the table later. The `–y` and `–P` arguments to the `lp` command cause a filter sequence to be built even if there is no need for a conversion of content types.

In general, each template is of the following form:

keyword pattern = *replacement*

The *keyword* names the characteristic that the template attempts to map into a filter-specific option; each valid *keyword* is listed in the table below.

A *pattern* is one of the following: a literal pattern of one of the forms listed in the table, a single asterisk (*), or a regular expression. If *pattern* matches the value of the characteristic, the template fits and is used to generate a filter-specific option. The *replacement* is what will be used as the option.

Regular expressions are the same as those found on the `regexp`(5) manual page. This includes the `\(...\)` and `\`*n* constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the &, which can be used to copy the entire *pattern* into the *replacement*.

The *replacement* can also contain a *; it too, is replaced with the entire *pattern*, just like the & of `regexp`(5).

The keywords are:

`lp` Option	Characteristic	*keyword*	Possible *patterns*
`–T`	Content type (input)	`INPUT`	content-type
not applicable	Content type (output)	`OUTPUT`	content-type
not applicable	Printer type	`TERM`	printer-type
`–d`	Printer name	`PRINTER`	*printer-name*
`–f`, `–o` cpi=	Character pitch	`CPI`	integer
`–f`, `–o` lpi=	Line pitch	`LPI`	integer
`–f`, `–o` length=	Page length	`LENGTH`	integer
`–f`, `–o` width=	Page width	`WIDTH`	integer
`–P`	Pages to print	`PAGES`	page-list
`–S`	Character set Print wheel	`CHARSET CHARSET`	character-set-name print-wheel-name
`–f`	Form name	`FORM`	form-name
`–y`	Modes	`MODES`	mode
`–n`	Number of copies	`COPIES`	*integer*

Large File Behavior

See `largefile`(5) for the description of the behavior of `lpfilter` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Printing examples.

For example, the template

```
MODES landscape = –l
```

shows that if a print request is submitted with the –y landscape option, the filter will be given the option –l.

As another example, the template

```
TERM * = –T *
```

shows that the filter will be given the option –T *printer-type* for whichever *printer-type* is associated with a print request using the filter.

As a last example, consider the template

```
MODES prwidth\=\(.*\) = –w\1
```

Suppose a user gives the command

```
lp –y prwidth=10
```

From the table above, the LP print service determines that the –y option is handled by a MODES template. The MODES template here works because the pattern prwidth=) matches the prwidth=10 given by the user. The replacement –w1 causes the LP print service to generate the filter option –w10. If necessary, the LP print service will construct a filter pipeline by concatenating several filters to handle the user's file and all the print options. See sh(1) for a description of a pipeline. If the print service constructs a filter pipeline, the INPUT and OUTPUT values used for each filter in the pipeline are the types of input and output for that filter, not for the entire pipeline.

Resetting a Filter to Defaults

If the filter named is one originally delivered with the LP print service, the –i option restores the original filter description.

Deleting a Filter

The –x option is used to delete the filter specified in filter-name from the LP filter table.

Listing a Filter Description

The –l option is used to list the description of the filter named in filter-name. If the command is successful, the following message is sent to standard output:

Input types: *content-type-list* Output types: *content-type-list* Printer types: *printer-type-list*

If the command fails, an error message is sent to standard error.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the `lpfilter` command requires the `administer printing` authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`lp`(1), `lpadmin`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`sh`(1), `attributes`(5), `largefile`(5), `regexp`(5)

System Administration Guide, Volume I

NOTES

If the `lp` command specifies more than one document, the filtering chain is determined by the first document. Other documents may have a different format, but they will print correctly only if the filter chain is able to handle their format.

NAME

lpforms – Administer forms used with the LP print service

SYNOPSIS

lpforms –f *form-name option*

lpforms –f *form-name* –A *alert-type* [–P *paper-name* [–d]] [–Q *requests*] [–W *minutes*]

DESCRIPTION

The `lpforms` command administers the use of preprinted forms, such as company letterhead paper, with the LP print service. A form is specified by its *form-name*. Users may specify a form when submitting a print request (see `lp`(1)). The argument `all` can be used instead of *form-name* with either of the command lines shown above. The first command line allows the administrator to add, change, and delete forms, to list the attributes of an existing form, and to allow and deny users access to particular forms. The second command line is used to establish the method by which the administrator is alerted that the form *form-name* must be mounted on a printer.

OPTIONS

–f *formname* Specify a form.

The first form of `lpforms` requires that one of the following options (–, –l, –F, –x) must be used:

–F *pathname* To add or change form *form-name*, as specified by the information in *pathname*.

– To add or change form *form-name*, as specified by the information from standard input.

–x To delete form *form-name* (this option must be used separately; it may not be used with any other option).

–l To list the attributes of form *form-name*.

The second form of the `lpforms` command requires the –A *alert-type* option. The other options are optional.

–A *alert-type* Defines an alert to mount the form when there are queued jobs which need it.

–P *paper-name* [–d] Specify the paper name when creating the form. If –d is specified, this paper is the default.

–Q *requests* An alert will be sent when a certain number of print requests that need the form are waiting.

–W *minutes* An alert will be sent at intervals specified by *minutes*.

USAGE

Adding or Changing a Form

The –F *pathname* option is used to add a new form, *form-name*, to the LP print service, or to change the attributes of an existing form. The form description is

taken from *pathname* if the –F option is given, or from the standard input if the – option is used. One of these two options must be used to define or change a form.

pathname is the path name of a file that contains all or any subset of the following information about the form.

```
Page length: scaled-decimal-number1
Page width: scaled-decimal-number2
Number of pages: integer
Line pitch: scaled-decimal-number3
Character pitch: scaled-decimal-number4
Character set choice: character-set/print-wheel [mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

The term "scaled-decimal-number" refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a "trailing" letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing `c`); numbers that show sizes in inches (marked with a trailing `i`); and numbers that show sizes in units appropriate to use (without a trailing letter); lines, characters, lines per inch, or characters per inch.

Except for the last two lines, the above lines may appear in any order. The `Comment:` and *comment* items must appear in consecutive order but may appear before the other items, and the `Alignment pattern:` and the *content* items must appear in consecutive order at the end of the file. Also, the *comment* item may not contain a line that begins with any of the key phrases above, unless the key phrase is preceded with a > sign. Any leading > sign found in the *comment* will be removed when the comment is displayed. There is no case distinction among the key phrases.

When this command is issued, the form specified by *form-name* is added to the list of forms. If the form already exists, its description is changed to reflect the new information. Once added, a form is available for use in a print request, except where access to the form has been restricted, as described under the –u option. A form may also be allowed to be used on certain printers only.

A description of each form attribute is below:

`Page length` and `Page Width`
: Before printing the content of a print request needing this form, the generic interface program provided with the LP print service will initialize the physical printer to handle pages *scaled-decimal-number1* long, and *scaled-decimal-number2* wide using the printer type as a key into the

terminfo(4) database. The page length and page width will also be passed, if possible, to each filter used in a request needing this form.

Number of pages
Each time the alignment pattern is printed, the LP print service will attempt to truncate the *content* to a single form by, if possible, passing to each filter the page subset of 1-*integer*.

Line pitch and Character pitch
Before printing the content of a print request needing this form, the interface program provided with the LP print service will initialize the physical printer to handle these pitches, using the printer type as a key into the terminfo(4) database. Also, the pitches will be passed, if possible, to each filter used in a request needing this form. *scaled-decimal-number3* is in lines-per-centimeter if a c is appended, and lines-per-inch otherwise; similarly, *scaled-decimal-number4* is in characters-per-centimeter if a c is appended, and characters-per-inch otherwise. The character pitch can also be given as elite (12 characters-per-inch), pica (10 characters-per-inch), or compressed (as many characters-per-inch as possible).

Character set choice
When the LP print service alerts an administrator to mount this form, it will also mention that the print wheel *print-wheel* should be used on those printers that take print wheels. If printing with this form is to be done on a printer that has selectable or loadable character sets instead of print wheels, the interface programs provided with the LP print service will automatically select or load the correct character set. If mandatory is appended, a user is not allowed to select a different character set for use with the form; otherwise, the character set or print wheel named is a suggestion and a default only.

Ribbon color
When the LP print service alerts an administrator to mount this form, it will also mention that the color of the ribbon should be *ribbon-color*.

Comment
The LP print service will display the *comment* unaltered when a user asks about this form (see lpstat(1)).

Alignment pattern
When mounting this form, an administrator can ask for the *content* to be printed repeatedly, as an aid in correctly positioning the preprinted form. The optional *content-type* defines the type of printer for which *content* had been generated. If *content-type* is not given, simple is assumed. Note that the *content* is stored as given, and will be readable only by the user lp.

When an existing form is changed with this command, items missing in the new information are left as they were. When a new form is added with this command, missing items will get the following defaults:

```
Page Length: 66
Page Width: 80
Number of Pages: 1
Line Pitch: 6
Character Pitch: 10
Character Set Choice: any
Ribbon Color: any
```

Deleting a Form

The –x option is used to delete the form *form-name* from the LP print service.

Listing Form Attributes

The –l option is used to list the attributes of the existing form *form-name*. The attributes listed are those described under Adding and Changing a Form, above. Because of the potentially sensitive nature of the alignment pattern, only the administrator can examine the form with this command. Other people may use the lpstat(1) command to examine the non-sensitive part of the form description.

Allowing and Denying Access to a Form

The –u option, followed by the argument allow:*login-ID-list* or –u deny:*login-ID-list* lets you determine which users will be allowed to specify a particular form with a print request. This option can be used with the –F or – option, each of which is described above under Adding or Changing a Form.

The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name!*login-ID*	A user on system *system_name*
system_name!all	All users on system *system_name*
all!*login-ID*	A user on all systems
all	All users on all systems

The LP print service keeps two lists of users for each form: an “allow-list” of people allowed to use the form, and a “deny-list” of people that may not use the form. With the –u allow option, the users listed are added to the allow-list and removed from the deny-list. With the –u deny option, the users listed are added to the deny-list and removed from the allow-list. (Both forms of the –u option can be run together with the –F or the – option.)

If the allow-list is not empty, only the users in the list are allowed access to the form, regardless of the content of the deny-list. If the allow-list is empty but the deny-list is not, the users in the deny-list may not use the form, (but all others may use it). All users can be denied access to a form by specifying

–f deny:all. All users can be allowed access to a form by specifying –f allow:all. (This is the default.)

Setting an Alert to Mount a Form

The –f *form-name* option is used with the –A *alert-type* option to define an alert to mount the form when there are queued jobs which need it. If this option is not used to arrange alerting for a form, no alert will be sent for that form.

The method by which the alert is sent depends on the value of the *alert-type* argument specified with the –A option. The *alert-types* are:

mail	Send the alert message using the mail command to the administrator.
write	Write the message, using the write command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.
quiet	Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the form *form-name* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the –Q option.
showfault	Attempt to execute a form alert handler on each system that has a print job for that form in the queue. The fault handler is /etc/lp/alerts/form. It is invoked with three parameters: *form_name*, date, *file_name*. *file_name* is the name of a file containing the form alert message.
none	Do not send messages until the –A option is given again with a different *alert-type* (other than quiet).
shell-command	Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blank spaces embedded in the command, enclose the command in quotes. Note that the mail and write values for this option are equivalent to the values mail *login-ID* and write *login-ID* respectively, where *login-ID* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the su command to change to another login-ID. If the su command has been used to change the user ID, then the *user-name* for the new ID is used.

`list` Display the type of the alert for the form on standard output. No change is made to the alert.

The message sent appears as follows:

The form *form-name* needs to be mounted on the printer(s):*printer* (*integer1* requests). *integer2* print requests await this form. Use the *ribbon-color* ribbon. Use the *print-wheel* print wheel, if appropriate.

The printers listed are those that the administrator has specified as candidates for this form. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the list of printers is the total number of requests awaiting the form. It will be less than the sum of the other numbers if some requests can be handled by more than one printer. The *ribbon-color* and *print-wheel* are those specified in the form description. The last line in the message is always sent, even if none of the printers listed use print wheels, because the administrator may choose to mount the form on a printer that does use a print wheel.

Where any color ribbon or any print wheel can be used, the statements above will read:

```
Use any ribbon.  Use any print-wheel.
```

If *form-name* is `any`, the *alert-type* defined in this command applies to any form for which an alert has not yet been defined. If *form-name* is `all`, the *alert-type* defined in this command applies to all forms.

If the `–W` *minutes* option is not given, the default procedure is that only one message will be sent per need to mount the form. Not specifying the `–W` option is equivalent to specifying `–W once` or `–W 0`. If *minutes* is a number greater than `0`, an alert will be sent at intervals specified by *minutes*.

If the `–Q` *requests* option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the form are waiting. If the `–Q` option is not given, or the value of *requests* is `1` or `any` (which are both the default), a message is sent as soon as anyone submits a print request for the form when it is not mounted.

Listing the Current Alert

The `–f` option, followed by the `–A` option and the argument `list` is used to list the *alert-type* that has been defined for the specified form *form-name*. No change is made to the alert. If *form-name* is recognized by the LP print service, one of the following lines is sent to the standard output, depending on the type of alert for the form.

– When *requests* requests are queued: alert with *shell-command* every *minutes* minutes

– When *requests* requests are queued: write to *user-name* every *minutes* minutes

– When *requests* requests are queued: mail to *user-name* every *minutes* minutes

– No alert

The phrase `every` *minutes* `minutes` is replaced with `once` if *minutes* (`–W` *minutes*) is `0`.

Terminating an Active Alert

The `–A quiet` option is used to stop messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the form has been mounted and then unmounted, messages will again be sent when the number of print requests reaches the threshold *requests*.

Removing an Alert Definition

No messages will be sent after the `–A none` option is used until the `–A` option is given again with a different *alert-type*. This can be used to permanently stop further messages from being sent as any existing alert definition for the form will be removed.

Large File Behavior

See `largefile`(5) for the description of the behavior of `lpforms` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/etc/lp/alerts/form` Fault handler for `lpform`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the `lpforms` command requires the `administer printing` authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`lp`(1), `lpstat`(1), `lpadmin`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`terminfo`(4), `attributes`(5), `largefile`(5)

System Administration Guide, Volume I

NAME

lpmove – Move print requests

SYNOPSIS

lpmove *request- ID destination*

lpmove *destination1 destination2*

DESCRIPTION

The `lpmove` command moves print requests queued by `lp`(1) or `lpr`(1) between destinations. Only use `lpmove` to move jobs on the local system.

`lpmove` requires the `administer printing` authorization.

The first form of `lpmove` moves specific print requests (*request-ID*) to a specific (*destination*).

The second form of the `lpmove` command moves all print requests from one destination (*destination1*) to another (*destination2*). This form of `lpmove` also rejects new print requests for *destination1*.

When moving requests, `lpmove` does not check the acceptance status of the destination to which the print requests are being moved (see `accept`(1M)). `lpmove` does not move requests that have options (for example, content type or requiring a special form) that cannot be handled by the new destination.

OPERANDS

The following operands are supported.

destination The name of the printer or class of printers (see `lpadmin`(1M)) to which `lpmove` moves a *specified* print request. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*), or Federated Naming Service (FNS) () names. See `printers.conf`(4) for information regarding the naming conventions for atomic and FNS names.

destination1 The name of the destination from which `lpmove` moves *all* print requests. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*), or Federated Naming Service (FNS) (…`/service/printer/`…) names. See `printers.conf`(4) for information regarding the naming conventions for atomic and FNS names, and `standards`(5) for information regarding POSIX.

destination2 The name of the destination to which `lpmove` moves all print requests. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*), or Federated Naming Service (FNS) (…`/service/printer/`…) names. See `printers.conf`(4) for information regarding the naming conventions for atomic and FNS names.

request-ID The specific print request to be moved. Specify *request-ID* as the identifier associated with a print request as reported by `lpstat`. See `lpstat`(1).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/var/spool/print/*` LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SUMMARY OF TRUSTED SOLARIS CHANGES

`lpmove` requires the `administer printing` authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`lp`(1), `lpstat`(1), `lpr`(1), `accept`(1M), `lpadmin`(1M), `lpsched`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`printers.conf`(4), `attributes`(5), `standards`(5)

System Administration Guide, Volume I

NAME

lpsched – Start the LP print service

SYNOPSIS

lpsched [–f *num_filters*] [–n *num_notifiers*] [–p *fd_limit*] [–r *reserved_fds*]

DESCRIPTION

The `lpsched` command starts or restarts the LP print service. `lpsched` must inherit these privileges: `file_chown`, `file_dac_read`, `file_dac_write`, `file_dac_search`, `file_downgrade_sl`, `file_mac_read`, `file_mac_search`, `file_mac_write`, `file_owner`, `file_setdac`, `file_setid`, `file_upgrade_sl`, `net_downgrade_sl`, `net_mac_read`, `net_setpriv`, `net_setid`, `proc_setclr`, `proc_setsl`, `proc_setid`, `proc_audit_tcb`, `proc_owner`, `proc_mac_write`, and `sys_trans_label`.

The `lpshut`(1M) command stops the LP print service. Printers that are restarted using `lpsched` reprint (in their entirety) print requests that were stopped by `lpshut`.

OPTIONS

The following options are supported:

–f *num_filters* — Specifies the number of concurrent slow filters that may be run on a print server. A default value of `1` is used if none is specified. Depending on server configuration, a value of `1` may cause printers to remain idle while there are jobs queued to them.

–n *num_notifiers* — Specifies the number of concurrent notification processes that can run on a print server. A default value of `1` is used when none is specified.

–p *fd_limit* — Specifies the file descriptor resource limit for the `lpsched` process. A default value of `4096` is used if none is specified. On extremely large and active print servers, it may be necessary to increase this value.

–r *reserved_fds* — Specifies the number of file descriptors that the scheduler reserves for internal communications under heavy load. A default value of `2` is used when none is specified. It should not be necessary to modify this value unless instructed to do so when troubleshooting problems under high load.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`non-zero` — An error occurred.

FILES

`/var/spool/lp/*` LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`lpsched` must be started from the Trusted Path. It must be started as lp or root at the label `admin_high` and must inherit appropriate privileges.

SEE ALSO

Trusted Solaris 7 Reference Manual

`enable`(1), `lp`(1), `lpstat`(1), `lpmove`(1M), `lpshut`(1M), `lpadmin`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`attributes`(5)

System Administration Guide, Volume I

NAME

lpshut – Stop the LP print service

SYNOPSIS

lpshut

DESCRIPTION

The `lpshut` command stops the LP print service.

`lpshut` requires the `administer printing` authorization.

Printers that are printing when `lpshut` is invoked stop printing. Start or restart printers using `lpsched`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/var/spool/lp/*` LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`lpshut` requires the `administer printing` authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`lp`(1), `lpstat`(1), `lpadmin`(1M), `lpmove`(1M), `lpsched`(1M)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`attributes`(5)

System Administration Guide, Volume I

NAME

lpsystem – Register remote systems with the print service

DESCRIPTION

The `lpsystem` command is obsolete. The print system no longer uses the information generated by `lpsystem`. See `lpadmin`(1M), `lpusers`(1M) or `printers.conf`(4) for equivalent functionality.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the `lpsystem` command requires the `administer printing` authorization

SEE ALSO

Trusted Solaris 7 Reference Manual

`lpadmin`(1M), `lpusers`(1M)

SunOS 5.7 Reference Manual

`printers.conf`(4), `attributes`(5)

NAME

lpusers – Set printing queue priorities

SYNOPSIS

lpusers –d *priority-level*

lpusers –q *priority-level* –u *login-ID-list*

lpusers –u *login-ID-list*

lpusers –q *priority-level*

lpusers –l

DESCRIPTION

The `lpusers` command sets limits to the queue priority level that can be assigned to jobs submitted by users of the LP print service.

The first form of the command (with –d) sets the system-wide priority default to *priority-level*, where *priority-level* is a value of `0` to `39`, with `0` being the highest priority. If a user does not specify a priority level with a print request (see `lp`(1)), the default priority level is used. Initially, the default priority level is `20`.

The second form of the command (with –q and –u) sets the default highest *priority-level* (0-39) that the users in *login-ID-list* can request when submitting a print request. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name`!`*login-ID*	A user on the system *system_name*
system_name`!all`	All users on system *system_name*
`all!`*login-ID*	A user on all systems
`all`	All users on all systems

Users that have been given a limit cannot submit a print request with a higher priority level than the one assigned, nor can they change a request that has already been submitted to have a higher priority. Any print requests submitted with priority levels higher than allowed will be given the highest priority allowed.

The third form of the command (with –u) removes any explicit priority level for the specified users.

The fourth form of the command (with –q) sets the default highest priority level for all users not explicitly covered by the use of the second form of this command.

The last form of the command (with –l) lists the default priority level and the priority limits assigned to users.

OPTIONS

–d *priority-level* Set the system-wide priority default to *priority-level*.

–q *priority-level* –u *login-ID-list* Set the default highest *priority-level* that the users in *login-ID-list* can

request when submitting a print request.

–u *login-ID-list* Remove any explicit priority level for the specified users.

–q *priority-level* Set the default highest priority level for all users not explicitly covered.

–l List the default priority level and the priority limits assigned to users.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

non-zero An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the lpusers command requires the administer printing authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

lp(1)

SunOS 5.7 Reference Manual

attributes(5)

NAME

modload – Load a kernel module

SYNOPSIS

modload [–p] [–e *exec_file*] *filename*

DESCRIPTION

`modload` loads the loadable module *filename* into the running system. *filename* is an object file produced by `ld –r`. If *filename* is an absolute pathname then the file specified by that absolute path is loaded. If *filename* does not begin with a '/' then the path to load *filename* is relative to the current directory unless the –p option is specified. The kernel's modpath variable can be set using the `/etc/system` file. The default value of the kernel's modpath variable is set to the path where the operating system was loaded. Typically this is `/kernel /usr/kernel`. Hence if you type:

```
example# modload drv/foo
```

The kernel will look for `./drv/foo`.

If you type:

```
example# modload -p drv/foo
```

The kernel will look for `/kernel/drv/foo` and then `/usr/kernel/drv/foo`.

OPTIONS

–p — Use the kernel's internal modpath variable as the search path for the module.

–e *exec_file* — Specify the name of a shell script or executable image file that is executed after the module is successfully loaded. The first argument passed is the module ID (in decimal). The other argument is module specific. The module specific information is: the block and character major numbers for drivers, the system call number for system calls, or, for other module types, the index into the appropriate kernel table. See `modinfo`(1M)

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the `sys_devices` privilege.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`ld`(1), `add_drv`(1M), `modunload`(1M)

SunOS 5.7 Reference Manual

`kernel`(1M), `modinfo`(1M), `system`(4), `attributes`(5), `modldrv`(9S), `modlinkage`(9S), `modlstrmod`(9S), `module_info`(9S)

Writing Device Drivers Solaris 1.x to 2.x Transition Guide

NOTES

Use `add_drv`(1M) to add device drivers, not modload. See *Writing Device Drivers* for procedures on adding device drivers.

NAME

modunload – Unload a module

SYNOPSIS

modunload –i *module_id* [–e *exec_file*]

DESCRIPTION

modunload unloads a loadable module from the running system. The *module_id* is the ID of the module as shown by modinfo(1M). If ID is 0, all modules that were autoloaded which are unloadable, are unloaded. Modules loaded by modload(1M) are not affected.

OPTIONS

–i *module_id* Specify the module to be unloaded.

–e *exec_file* Specify the name of a shell script or executable image file to be executed before the module is unloaded. The first argument passed is the module ID (in decimal). There are two additional arguments that are module specific. For loadable drivers, the second and third arguments are the block major and character major numbers respectively. For loadable system calls, the second argument is the system call number. For loadable exec classes, the second argument is the index into the execsw table. For loadable filesystems, the second argument is the index into the vfssw table. For loadable streams modules, the second argument is the index into the fmodsw table. For loadable scheduling classes, the second argument is the index into the class array. Minus one is passed for an argument that does not apply.

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the sys_devices privilege.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

modload(1M)

SunOS 5.7 Reference Manual

modinfo(1M), attributes(5)

NAME

mount, umount – Mount or unmount file systems and remote resources

SYNOPSIS

mount [–p | –v]

mount [–F *FSType*] [*generic_options*] [–o *specific_options*] [–0] *special* | *mount_point*

mount [–F *FSType*] [*generic_options*] [–o *specific_options*] [–0] [–S *attribute_list*] *special mount_point*

mount –a [–F *FSType*] [–V] [*current_options*] [–o *specific_options*] [–S *attribute_list*] [*mount_point*…]

umount [–V] [–o *specific_options*] *special* | *mount_point*

umount –a [–V] [–o *specific_options*] [*mount_point*…]

DESCRIPTION

`mount` attaches a file system to the file system hierarchy at the *mount_point* , which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

`umount` unmounts a currently mounted file system, which may be specified either as a *mount_point* or as *special* , the device on which the file system resides.

`mount` and `umount` maintain a table of mounted file systems in `/etc/mnttab` , which is described in `mnttab`(4) . `mount` adds an entry to the mount table; `umount` removes an entry from the table.

When invoked with both the *special* and *mount_point* arguments and the –F option, `mount` validates all arguments except for *special* and invokes the appropriate *FSType* -specific `mount` module. If invoked with no arguments, `mount` lists all the mounted file systems recorded in the mount table, `/etc/mnttab` . If invoked with a partial argument list (with only one of *special* or *mount_point* , or with both *special* or *mount_point* specified but not *FSType*), `mount` will search `/etc/vfstab` for an entry that will supply the missing arguments. If no entry is found, and the special argument starts with "/", the default local file system type specified in `/etc/default/fs` will be used. Otherwise the default remote file system type will be used. The default remote file system type is determined by the first entry in the `/etc/dfs/fstypes` file. After filling in missing arguments, `mount` will invoke the *FSType* -specific `mount` module.

The –S option can be used to assign any or all of the following mount-time security attributes to the named file system when appropriate: an ACL , a mode, a user ID , a group ID , a sensitivity label, forced privilege(s), allowed privilege(s), a file attribute flag, a filesystem label range, or an MLD prefix. If the –S option is not used, `mount` also searches `/etc/security/tsol/vfstab_adjunct` for any security attributes that may be specified there for the file system being

mounted. Specifying mount-time attributes is useful only when mounting file systems that do not support the attributes.

Mount-time security attributes should be specified for file systems whose objects do not have any attributes, such as user and group ID s, and for file systems whose objects do not support the Trusted Solaris extended security attributes, such as sensitivity labels. When a required attribute is not specified at mount-time, a default value is applied. The defaults are described in the `OPTIONS` section, where the keywords are defined for the `–S` option.

File system types `UFS` , `TMPFS` , and `NFS` (from a Trusted Solaris server) have a full set of Trusted Solaris extended security attributes already defined. (See the `getfsattr`(1M) man page for how to get attributes on mounted file systems). Because the attributes can be changed on these file systems *after* they are mounted, they are called *variable* file systems. For example, the sensitivity label on a file in a variable file system can be changed by an authorized user. The security attributes on a variable file system can be overridden at mount time, but individual objects in the file system retain any attributes that were originally set on the objects.

File systems that do not support the Trusted Solaris extended security attributes are called *fixed* because any attributes assigned to them (either at mount time or by default) cannot be changed. For example, the sensitivity label specified at mount time for a fixed-attribute file system cannot be changed on any of the objects in that file system. An object that is moved or copied from the fixed file system to a variable file system can be changed after the move.

Mount-time security attributes override existing security attributes on a file system. However, mount-time attributes never override security attributes on the files and directories within the file system.

Without privilege, `mount` can be used to list mounted file systems and resources. To be able to mount and unmount, the `mount` command must have the `sys_mount` privilege and must run with an effective UID of `0` . The `umount` command must have the `sys_mount` privilege. Mandatory and discretionary read access is required both to the mount point and to the device being mounted; otherwise, MAC or DAC override privileges are required as described in `Intro`(2) . To succeed in all cases, `mount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` . To succeed in all cases, `umount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , and `file_dac_search` .

OPTIONS

`–F` *FSType*

Used to specify the *FSType* on which to operate. The *FSType* must be specified or must be determinable from `/etc/vfstab` , or by consulting `/etc/default/fs` or `/etc/dfs/fstypes` .

`–a` [*mount_points* . . .]
Perform `mount` or `umount` operations in parallel, when possible.

If mount points are not specified, `mount` will mount all file systems whose `/etc/vfstab` "mount at boot" field is "yes". If mount points are specified, then `/etc/vfstab` "mount at boot" field will be ignored.

If mount points are specified, `umount` will only unmount those mount points. If none is specified, then `umount` will attempt to unmount all filesystems in `/etc/mnttab` , with the exception of certain system required file systems: `/` , `/usr` , `/var` , `/proc` , `/dev/fd` , and `/tmp` .

`–p`
Print the list of mounted file systems in the `/etc/vfstab` format. Must be the only option specified.

`–v`
Print the list of mounted file systems in verbose format. Must be the only option specified.

`–V`
Echo the complete command line, but do not execute the command. `umount` generates a command line by using the options and arguments provided by the user and adding to them information derived from `/etc/mnttab` . This option should be used to verify and validate the command line.

generic_options
Options that are commonly supported by most *FSType* -specific command modules. The following options are available:

`–m`
Mount the file system without making an entry in `/etc/mnttab` .

`–g`
Globally mount the file system. On a clustered system, this globally mounts the file system on all nodes of the cluster. On a non-clustered system this has no effect.

`–o`
Specify *FSType* -specific options in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the *FSType* -specific module of the command. (See `mount_ufs`(1M))

–O

Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error “device busy”.

–r

Mount the file system read-only.

–S *attribute_list*

Specify in *attribute_list* a quoted semicolon-separated list of security attributes to associate with the file-system mount. Each attribute is specified with a value assigned to a keyword in semicolon-separated fields. All keywords are optional and follow the format:

keyword=value

where *keyword* is one of the following:

`acc_acl`	Sets the same ACL on all files or directories in the file system. See `aclfromtext`(3) for the format.
`mode`	Sets a DAC permission mode for each object in the file system. The only supported mode is the absolute mode, which is specified using octal numbers. See the description for the absolute-mode parameter on the `chmod`(1) man page. (Because the mode is an object-level attribute that has precedence over any mount-time attributes, setting a mode is only useful in the rare case when the type of file system being mounted does not support permission bits. In such cases, it is recommended that an explicit value be specified for the mode.)
`attr_flg`	Sets an attribute flag on all files in the file system. The only supported `attr_flag` value is `public` , whose effect is that when certain read operations are performed on any object in the file system on which this flag is set, audit records are not generated even when the operations are part of a preselected audit class, with the following exception. If the audit pseudo event for use of privilege (`AUE_UPRIV`) is included in a preselected audit class and if the operation involves the use of privilege), then an audit record is always generated. With the previous exception, the read operations for which audit records are

not generated when the public flag is set are: access(2) , fgetcmwlabel(2) , fgetsldname(2) , fstatvfs(2) , getcmwfsrange(2) , getcmwlabel(2) , getfpriv(2) , getmldadorn(2) , getsldname(2) , lgetcmwlabel(2) , lstat(2) , open(2) —read only, pathconf(2) , preadl(2) , readl(2) , readlink(2) , stat(2) , statvfs(2) , mldlstat(3) , and mldstat(3) . See *Trusted Solaris Audit Administration* and *Trusted Solaris Administrator's Procedures* for more details.

`gid` Sets the group ID for all objects in the file system. (Because the GID is an object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have GID s on its files or directories. In such cases, it is recommended that an explicit value be specified for the GID .)

`uid` Sets the user ID for all objects in the file system. (Because the UID is an object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have UID s on its files or directories. In such cases, it is recommended that an explicit value be specified for the UID .)

`slabel` Sets the sensitivity label for all objects in the file system. Specify the sensitivity label in hexadecimal or text format.

`forced` Specify one or more forced privileges for all executable files in the file system. Specify symbolic privilege name(s) in a comma-separated list (such as: `forced=file_audit, file_chown;`) or use `all` to indicate all privileges. Using `none` or omitting the keyword results in no forced privileges being applied. See `priv_desc`(4) . Any forced privileges must be a subset of the allowed privileges.

`allowed` Specify one or more allowed privilege(s) for all executable files in the file system. Specify symbolic privilege names in a comma-separated list (such as: `allowed=file_audit, file_chown;`) or use `all` to indicate all privileges. Using `none` or omitting the keyword results in no allowed privileges being applied.

See `priv_desc`(4) for names of privileges. Any allowed privilege(s) must be a superset of the forced privileges.

`low_range` Specify the lower bound of the file system label range as a sensitivity label in text format.

`hi_range` Specify the upper bound of the file system label range as a sensitivity label in text format.

`mld_prefix` Set a prefix to be used in the adorned names of multilevel directories. (See `multilevel directories` in the `DEFINITIONS` in `Intro`(2) for more about the MLD prefix.) Specify the value in text format (such as: `.MLD.` or `.hidden.`). On unlabeled (fixed attribute) file systems, the prefix generally has no useful effect—with the exception that an `mld_prefix` should be supplied if a variable filesystem is being mounted on the unlabeled filesystem and the root of the variable filesystem is an MLD .

Any of the above keywords may be omitted.

Note - Note: The semicolon separators between keyword / value pairs and any brackets used to specify sensitivity labels must be commented out so that the separators and brackets can be interpreted properly by the shell.

When a keyword appears without an attribute value or when a keyword is missing, a default value is assigned to that attribute. The default values for fixed attribute file systems are:

`acc_acl` None

`mode` The mode should always be explictly set for file systems that do not support file access modes, such as MS-DOS (`pcfs` type) file systems.

`attr_flag` None

`gid` The GID should always be explictly set for file systems that do not support group ID s, such as MS-DOS (`pcfs` type) file systems.

`uid` The UID should always be explictly set for file systems that do not support user ID s, such as MS-DOS (`pcfs` type) file systems.

`slabel`	The default sensitivity label of a fixed file system being mounted from a local device (such as a hard disk, floppy, or CD-ROM) is the sensitivity label of the device. For an allocated device, the file system is assigned the sensitivity label at which the device was allocated.
`forced`	None
`allowed`	None
`low_range`	`ADMIN_LOW`
`hi_range`	`ADMIN_HIGH`
`mld_prefix`	None

For example, the assignment of `forced=;` results in the default of "none" being applied.

USAGE

See `largefile`(5) for the description of the behavior of `mount` and `umount` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

SUMMARY OF TRUSTED SOLARIS CHANGES

Trusted Solaris security policy applies when mounting and unmounting file systems.

Mount-time security attributes may be specified either by using `mount` with the –S option on the command line or by specifying the attributes in the `vfstab_adjunct` file. Mount-time security attributes override existing security attributes on a file system. However, they never override security attributes on the files and directories within the file system. When access-control decisions are made, security attributes on a file or directory take precedence over security attributes specified either at the filesystem level or at mount time.

Except when merely listing mounted file systems and resources, `mount` must run with an effective UID of `0` and with the `sys_mount` privilege. `umount` also must run with an effective UID of `0` and with the `sys_mount` privilege. To succeed in all cases, `mount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` .

Note - Information labels (IL s) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any IL s on communications and files from systems running earlier releases as ADMIN_LOW .

Objects still have CMW labels, and CMW labels still include the IL component: IL[SL] ; however, the IL component is fixed at ADMIN_LOW .

As a result, Trusted Solaris 7 has the following characteristics:

- IL s do not display in window labels; SL s (Sensitivity Labels) display alone within brackets.
- IL s do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return ADMIN_LOW .
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting IL s are always ADMIN_LOW , and cannot be set on any objects.
- Options related to information labels in the label_encodings(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

FILES

/etc/mnttab	Mount table
/etc/default/fs	Default local file system type. Default values can be set for the following flags in /etc/default/fs . For example: LOCAL=ufs Specifies that LOCAL: is the default partition for a command if no *FSType* is specified.
/etc/vfstab	List of default parameters for each file system.
/etc/security/tsol/vfstab_adjunct	Mount-time attributes for file systems.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

getfsattr(1M) , getmldadorn(1) , mount_hsfs(1M) , mount_nfs(1M) , mount_pcfs(1M) , mount_tmpfs(1M) , mount_ufs(1M) , mountall(1M) , setfsattr(1M) , setmnt(1M) , mnttab(4) , priv_desc(4) , vfstab(4) , vfstab_adjunct(4)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`setfacl`(1), `mount_cachefs`(1M), `default_fs`(4), `attributes`(5)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mountall, umountall – Mount, unmount multiple file systems

SYNOPSIS

mountall [–F *FSType*] [–l | –r][*file_system_table*]

umountall [–k] [–s] [–F *FSType*] [–l | –r]

umountall [–k] [–s] [–h *host*]

DESCRIPTION

`mountall` is used to mount file systems specified in a file system table. The file system table must be in `vfstab`(4) format. If no *file_system_table* is specified, `/etc/vfstab` will be used. If '–' is specified as *file_system_table* , `mountall` will read the file system table from the standard input. `mountall` only mounts those file systems with the `mount at boot` field set to `yes` in the *file_system_table* .

Each file system which has an `fsckdev` entry specified in the file system table will be checked using `fsck`(1M) in order to determine if it may be safely mounted. If the file system does not appear mountable, it is fixed using `fsck` before the mount is attempted. File systems with a '–' entry in the `fsckdev` field will be mounted without first being checked.

`umountall` causes all mounted file systems except `root` , `/proc` , `/var` , and `/usr` to be unmounted. If the *FSType* is specified, `mountall` and `umountall` limit their actions to the *FSType* specified. There is no guarantee that `umountall` will unmount *busy* filesystems, even if the –k option is specified.

`mountall` and `umountall` must run with an effective UID of `0` and with the `sys_mount` privilege.

Mandatory and discretionary read access are required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2) . To succeed in all cases, the `mountall` and `umountall` commands need the privileges: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` .

OPTIONS

–F Specify the *FSType* of the file system to be mounted or unmounted.

–h *host* Unmount all file systems listed in `/etc/mnttab` that are remote-mounted from host.

–k Use the `fuser` –k *mount-point* command. See the `fuser`(1M) for details. The –k option sends the `SIGKILL` signal to each process using the file. As this option spawns kills for each process, the kill messages may not show up immediately. There is no guarantee that `umountall` will unmount *busy* filesystems, even if the –k option is specified.

–l Limit the action to local file systems.

–r Limit the action to remote file system types.

–s Do not perform the umount operation in parallel.

SUMMARY OF TRUSTED SOLARIS CHANGES

Trusted Solaris security policy applies when mounting and unmounting file systems.

mountall and umountall must run with an effective UID of 0 and with the sys_mount privilege.

Mandatory and discretionary read access are required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in Intro(2) . To succeed in all cases, the mountall and umountall commands need the privileges: file_mac_read , file_dac_read , file_mac_write , file_dac_write , file_mac_search , file_dac_search , net_privaddr , proc_setsl , proc_setil , sys_mount , and sys_trans_label .

Mount-time security attributes may be specified in the vfstab_adjunct file.

FILES

/etc/mnttab mounted file system table

/etc/vfstab table of file system defaults

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

mount(1M) , mnttab(4) , vfstab(4) , vfstab_adjunct(4)

SunOS 5.7 Reference Manual

fsck(1M) , attributes(5)

DIAGNOSTICS

No messages are printed if the file systems are mountable and clean.

Error and warning messages come from fsck(1M) and mount(1M) .

NAME

mountd – Server for NFS mount requests and NFS access checks

SYNOPSIS

/usr/lib/nfs/mountd [–v] [–r]

DESCRIPTION

`mountd` is an RPC server that answers requests for NFS access information and file system mount requests. It reads the file `/etc/dfs/sharetab` to determine which file systems are available for mounting by which remote machines. See `sharetab`(4). `nfsd` running on the local server will contact `mountd` the first time an NFS client tries to access the file system to determine whether the client should get read-write, read-only, or no access. This access can be dependent on the security mode used in the remoted procedure call from the client. See `share_nfs`(1M).

The command also provides information as to what file systems are mounted by which clients. This information can be printed using the `showmount`(1M) command.

The `mountd` daemon is automatically invoked in run level 3.

The `sys_nfs`, `sys_devices`, `sys_net_config`, `sys_audit`, `net_mac_read`, `net_privaddr`, `file_mac_read`, `file_mac_write`, `file_mac_search`, `file_dac_search`, `proc_setsl`, and `proc_setclr` privileges are required to run this daemon. This daemon must be run with an effective UID of `0` and be started from the Trusted Path; otherwise, the daemon sets its sensitivity label to `ADMIN_LOW` and clearance to `ADMIN_HIGH`.

OPTIONS

–v Run the command in verbose mode. Each time `mountd` determines what access a client should get, it will log the result to the console, as well as how it got that result.

–r Reject mount requests from clients. Clients that have file systems mounted will not be affected.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `sys_nfs`, `sys_devices`, `sys_net_config`, `sys_audit`, `net_mac_read`, `net_privaddr`, `file_mac_read`, `file_mac_search`, `file_dac_search`, `proc_setil`, `proc_setsl`, and `proc_setclr` privileges are required to run this daemon. This daemon must be run with an effective UID of `0` and be started from the Trusted Path. If not started at a sensitivity label of `ADMIN_LOW` and clearance of `ADMIN_HIGH`, `mountd` sets its sensitivity label and clearance to these values.

For the mount request to succeed, this daemon requires the client to have the `sys_mount` privilege. Unless the –n option is specified, the client request must have a UID equal to `0` and must bind to a privileged port.

FILES

`/etc/dfs/sharetab` shared file system table

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`nfsd`(1M), `share_nfs`(1M), `showmount`(1M), `sharetab`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

If `nfsd` is running, `mountd` must also be running in order to be assured that the NFS server can respond to requests, otherwise, the NFS service can hang.

Some routines that compare hostnames use case-sensitive string comparisons; some do not. If an incoming request fails, verify that the case of the hostname in the file to be parsed matches the case of the hostname called for, and attempt the request again.

NAME

mount_hsfs – Mount hsfs file systems

SYNOPSIS

mount –F hsfs [*generic_options*] [–o *FSType-specific_options*] [–O] [–S *attribute_list*] *special* | *mount_point*

mount –F hsfs [*generic_options*] [–o *FSType-specific_options*] [–O] [–S *attribute_list*] *special mount_point*

DESCRIPTION

`mount` attaches a High Sierra file system (hsfs) to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

If `mount` is invoked with *special* or *mount_point* as the only arguments, `mount` will search `/etc/vfstab` to fill in the missing arguments, including the *FSType-specific_options*; see `mount`(1M) for more details.

If the file system being mounted contains `Rock Ridge` extensions, by default they will be used, enabling support of features not normally available under High Sierra file systems, such as symbolic links and special files.

Security attributes can be specified at mount time, either with the –S option on the `mount` command line or in the `vfstab_adjunct`(4) file. See the `DESCRIPTION` in the `mount`(1M) man page for more about specifying security attributes.

To succeed, the `mount` command must have the `sys_mount` privilege and must run with an effective UID of `0`. Mandatory and discretionary read access are required to both the mount point and the device being mount; to override MAC and DAC restrictions requires privilege as described in `Intro`(2). To succeed in all cases, `mount –F hsfs` needs the `file_mac_read` and `file_mac_write` privileges.

OPTIONS

generic_options

See `mount`(1M) for the list of supported options.

–o

Specify `hsfs` file system specific options. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

`global` | `noglobal`

If `global` is specified and supported on the file system, and the system in question is part of a cluster, the file system will be globally visible on all nodes of the cluster. If `noglobal` is specified, the mount will not be globally visible. The default behavior is `noglobal`.

`ro`

Mount the file system read-only. This option is required.

`nrr`

`no Rock Ridge`: if `Rock Ridge` extensions are present in the file system, ignore them; interpret it as a regular High Sierra file system.

`notraildot`

File names on High Sierra file systems consist of a proper name and an extension separated by a '`.`' (dot) character. By default, the separating dot is always considered part of the file's name for all file access operations, even if there is no extension present. Specifying `notraildot` makes it optional to specify the trailing dot to access a file whose name lacks an extension.

Exceptions: This option is effective only on file systems for which `Rock Ridge` extensions are not active, either because they are not present on the CD-ROM, or they are explicitly ignored via the `nrr` option. If `Rock Ridge` extensions are active, `hsfs` quietly ignores this option.

`nomaplcase`

File names on High Sierra cdroms with no `Rock Ridge` extensions present should be uppercase characters only. By default, `hsfs` maps file names read from a `non-Rock Ridge` disk to all lowercase characters. `nomaplcase` turns off this mapping. The exceptions for `notraildot` discused above apply to `nomaplcase`.

`nosuid`

By default the file system is mounted with setuid execution allowed. Specifying `nosuid` causes the file system to be mounted with setuid execution disallowed.

`devices` | `nodevices`

Allow (disallow) access to character and block devices. The default is `devices`.

Note: In the Trusted Solaris environment, device special files are typically located only in the `/dev` and `/devices` directories in the root file system. All other file systems should be mounted with the `nodevices` option to prevent recognition of devices that may reside in any other directories. The recognition of devices is also affected by the use of the `devices` or `nodevices` options to the `share`(1M) command, either on the command line or in the `dfstab`(4) file.

`priv` | `nopriv`

Forced privileges on executables are allowed or disallowed. The default is `priv`. The recognition of forced privileges is also affected by the use of the `priv` or `nopriv` option to the `share`(1M) command, either on the command line or in the `dfstab`(4) file.

–O
Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error `device busy`.

–S *attribute_list*
See the `DESCRIPTION` and the attribute list on the `mount`(1M) man page.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `nodevices` and `nopriv` options have been added. Trusted Solaris security policy applies when mounting and unmounting file systems.

Except when merely listing mounted file systems and resources, `mount` must run with an effective UID of `0` and with the `sys_mount` privilege.

Mandatory and discretionary read access is required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2). To succeed in all cases, `mount –F hsfs` needs the `file_mac_read` and `file_mac_write` privileges.

Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Specifying attributes fails for file systems and file system objects (files or directories) that already have a specified attribute. Trusted Solaris security policy applies when mounting. See the `mount`(1M) and `vfstab_adjunct`(4) man pages for more details.

FILES

`/etc/mnttab`
Table of mounted file systems.

`/etc/vfstab`
List of default parameters for each file system.

`/etc/security/tsol/vfstab_adjunct`
Mount-time attributes for file systems.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mountall`(1M), `mount`(2), `mnttab`(4), `vfstab`(4), `vfstab_adjunct`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_nfs – Mount remote NFS resources

SYNOPSIS

mount [–F nfs] [*generic_options*] [–o *specific_options*] [–O] [–S *attribute_list*] *resource*

mount [–F nfs] [*generic_options*] [–o *specific_options*] [–O] [–S *attribute_list*] *mount_point*

mount [–F nfs] [*generic_options*] [–o *specific_options*] [–O] [–S *attribute_list*] *resource mount_point*

DESCRIPTION

The `mount` utility attaches a named *resource* to the file system hierarchy at the pathname location *mount_point*, which must already exist. If *mount_point* has any contents prior to the `mount` operation, the contents remain hidden until the *resource* is once again unmounted.

If the resource is listed in the `/etc/vfstab` file, the command line can specify either *resource* or *mount_point*, and `mount` will consult `/etc/vfstab` for more information. If the –F option is omitted, `mount` takes the file system type from `/etc/vfstab`.

If the resource is not listed in the `/etc/vfstab` file, then the command line must specify both the *resource* and the *mount_point*.

A named *resource* can have one of the following formats:

host:*pathname*
: Where *host* is the name of the NFS server host, and *pathname* is the path name of the directory on the server being mounted. The path name is interpreted according to the server's path name parsing rules and is not necessarily slash-separated, though on most servers, this will be the case.

nfs:/ /*host*[:*port*]/*pathname*
: This is an NFS URL and follows the standard convention for NFS URLs as described in *Internet RFC 2225 — NFS URL Scheme*. See the discussion of URL's and the public option under `NFS FILE SYSTEMS` below for a more detailed discussion.

A comma–separated list of *host*:*pathname* and/or *nfs*:/ /*host*[:*port*]/*pathname* resources
: See the discussion of Replicated file systems and failover under `NFS FILE SYSTEMS` below for a more detailed discussion.

`mount` maintains a table of mounted file systems in `/etc/mnttab`, described in `mnttab`(4). See `mount`(1M) for more details.

Security attributes can be specified at mount time, either with the –S option on the mount command line or in the vfstab_adjunct(4) file. See the DESCRIPTION in the mount man page for more about specifying security attributes.

Trusted Solaris security policy applies when mounting and unmounting file systems.

mount must run with an effective UID of 0 and with the sys_mount and net_privaddr privileges. To succeed in all cases, mount also needs: file_mac_read, file_dac_read, file_mac_write, file_dac_write, file_mac_search, file_dac_search, proc_setsl, and sys_trans_label.

OPTIONS

See mount(1M) for the list of supported *generic_options*.

–o *specific_options*

Set file system specific options according to a comma-separated list with no intervening spaces.

acdirmax=*n*

Hold cached attributes for no more than *n* seconds after directory update. The default value is 60.

acdirmin=*n*

Hold cached attributes for at least *n* seconds after directory update. The default value is 30.

acregmax=*n*

Hold cached attributes for no more than *n* seconds after file modification. The default value is 60.

acregmin=*n*

Hold cached attributes for at least *n* seconds after file modification. The default value is 3.

actimeo=*n*

Set *min* and *max* times for regular files and directories to *n* seconds.

bg | fg

If the first attempt fails, retry in the background, or, in the foreground. The default is fg.

devices | nodevices

Allow (disallow) opens on character and block devices. The default is devices.

Note: In the Trusted Solaris environment, device special files are typically located only in the `/dev` and `/devices` directories in the root file system. All other file systems should be mounted with the `nodevices` option to prevent recognition of devices that may reside in any other directories.

`grpid`

By default, the GID associated with a newly created file will obey the System V semantics; that is, the GID is set to the effective GID of the calling process. This behavior may be overridden on a per-directory basis by setting the set-GID bit of the parent directory; in this case, the GID of a newly created file is set to the GID of the parent directory (see `open`(2) and `mkdir`(2)). Files created on file systems that are mounted with the `grpid` option will obey BSD semantics independent of whether the set-GID bit of the parent directory is set; that is, the GID is unconditionally inherited from that of the parent directory.

`hard` | `soft`

Return an error if the server does not respond, or continue the retry request until the server responds. The default value is `hard`.

`intr` | `nointr`

Allow (do not allow) keyboard interrupts to kill a process that is hung while waiting for a response on a hard-mounted file system. The default is `intr`, which makes it possible for clients to interrupt applications that may be waiting for a remote mount.

`kerberos`

This option has been deprecated in favor of the `sec=`*krb4* option.

`noac`

Suppress data and attribute caching.

`port=`*n*

The server IP port number. The default is `NFS_PORT`. If the `port` option is specified, and if the resource includes one or more NFS URLs, and if any of the URLs include a `port` number, then the `port` number in the option and in the URL must be the same.

`posix`

Request POSIX.1 semantics for the file system. Requires a mount Version 2 `mountd`(1M) on the server. See `standards`(5) for information regarding POSIX.

`priv` | `nopriv`

Forced privileges on executables are allowed or disallowed. The default is `priv`.

`proto=`<*netid*>

<*netid*> is a value of network_id field from entry in the `/etc/netconfig` file. By default, the transport protocol used for the NFS mount will be first available connection oriented transport supported on both the client and the server. If no connection oriented transport is found, then the first available connectionless transport is used. This default behavior can be overridden with the `proto=`<*netid*> option.

`public`

The `public` option forces the use of the public file handle when connecting to the NFS server. The resource specified may or may not have an NFS URL. See the discussion of URL's and the public option under `NFS FILE SYSTEMS` below for a more detailed discussion.

`quota` | `noquota`

Enable or prevent `quota`(1M) to check whether the user is over quota on this file system; if the file system has quotas enabled on the server, quotas will still be checked for operations on this file system. This option is not supported in the Trusted Solaris environment.

`remount`

Remounts a read-only file system as read-write (using the `rw` option). This option cannot be used with other `–o` options, and this option works only on currently mounted read-only file systems.

`retrans=`*n*

Set the number of NFS retransmissions to *n*. The default value is `5`. For connection-oriented transports, this option has no effect because it is assumed that the transport will perform retransmissions on behalf of NFS.

`retry=`*n*

The number of times to retry the `mount` operation. The default is `10000`.

`ro` | `rw`

resource is mounted read-only or read-write. The default is `rw`.

`rsize=`*n*

Set the read buffer size to *n* bytes. The default value is `32768` when using Version 3 of the NFS protocol. The default can be negotiated down if the server prefers a smaller transfer size. When using Version 2, the default value is `8192`.

`sec=`*mode*

Set the security *mode* for NFS transactions. If `sec=` is not specified, then the default action is to use `AUTH_SYS` over NFS Version 2 mounts, or to negotiate a *mode* over NFS Version 3 mounts. NFS Version 3 mounts negotiate a security mode when the server returns an array of security modes. The client will pick the first mode in the array that is supported on the client. Only one mode can be specified with the `sec=` option. See `nfssec`(5) for the available *mode* options.

`secure`

This option has been deprecated in favor of the `sec=`*dh* option.

`suid` | `nosuid`

Allow or disallow `setuid` execution. The default is `suid`.

`timeo=`*n*

Set the NFS timeout to *n* tenths of a second. The default value is `11` tenths of a second for connectionless transports, and `600` tenths of a second for connection-oriented transports.

`vers=`<*NFS version number*>

By default, the version of NFS protocol used between the client and the server is the highest one available on both systems. If the NFS server does not support NFS Version 3 protocol, then the NFS mount will use NFS Version 2 protocol.

Note: File systems being mounted from Trusted Solaris 1.2 servers should be specified with `vers=2`. Because the Trusted Solaris 2.5.1 environment does not recognize security attributes, such as labels, on file systems mounted from NFS Version 2 servers, all such filesystems should be mounted as unlabeled filesystems and should have mount-time security attributes supplied for them either with the –`S` option or in the `vfstab_adjunct` file.

`wsize=`*n*

Set the write buffer size to *n* bytes. The default value is `32768` when using Version 3 of the NFS protocol. The default can be negotiated down if the server prefers a smaller transfer size. When using Version 2, the default value is `8192`.

–`O` Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error “device busy”.

NFS FILE SYSTEMS

Background versus Foreground

File systems mounted with the –bg option indicate that mount is to retry in the background if the server's mount daemon (mountd(1M)) does not respond. mount retries the request up to the count specified in the retry=*n* option. Once the file system is mounted, each NFS request made in the kernel waits timeo=*n* tenths of a second for a response. If no response arrives, the time-out is multiplied by 2 and the request is retransmitted. When the number of retransmissions has reached the number specified in the retrans=*n* option, a file system mounted with the soft option returns an error on the request; one mounted with the hard option prints a warning message and continues to retry the request.

Hard versus Soft

File systems that are mounted read-write or that contain executable files should always be mounted with the hard option. Applications using soft mounted file systems may incur unexpected I/O errors, file corruption, and unexpected program core dumps. The soft option is not recommended.

Authenticated Requests

The server may require authenticated NFS requests from the client. Either sec=*dh* or sec=*krb4* authentication may be required. See nfssec(5).

URLs and the public option

If the public option is specified, or if the *resource* includes and NFS URL, mount will attempt to connect to the server using the public file hand lookup protocol. See *Internet RFC 2054 — WebNFS Client Specification*. If the server supports the public file handle, the attempt is successful; mount will not need to contact the server's rpcbind(1M), and the mountd(1M) daemons to get the port number of the mount server and the initial file handle of *pathname*, respectively. If the NFS client and server are separated by a firewall that allows all outbount connections through specific ports, such as NFS_PORT, then this enables NFS operations through the firewall. The public option and the NFS URL can be specified independently or together. They interact as specified in the following matrix:

	resource style	
	host:*pathname*	NFS URL
public option	+ force public file handle and fail mountif not supported.	+ force public file handle and fail mountif not supported.

	+ use Native paths	+ use Canonical paths
default	+ use MOUNT protocol	+ try public file handle with Canonical paths. Fall back to MOUNT protocol if not supported.

A *Native path* is a path name that is interpreted according to conventions used on the native operating system of the NFS server. A *Canonical path* is a path name that is interpreted according to the URL rules. See *Internet RFC 1738 — Uniform Resource Locators (URL)*. Also, see EXAMPLES for uses of *Native* and *Canonical* paths.

`Replicated file systems and failover`

resource can list multiple read–only file systems to be used to provide data. These file systems should contain equivalent directory structures and identical files. It is also recommended that they be created by a utility such as `rdist`(1). The file systems may be specified either with a comma–separated list of *host:/pathname* entries and / or NFS URL entries, or with a comma –separated list of hosts, if all file system names are the same. If multiple file systems are named and the first server in the list is down, failover will use the next alternate server to access files. If the read–only option is not chosen, replication will be disabled. File access will block on the original if NFS locks are active for that file.

File Attributes

To improve NFS read performance, files and file attributes are cached. File modification times get updated whenever a write occurs. However, file access times may be temporarily out-of-date until the cache gets refreshed.

The attribute cache retains file attributes on the client. Attributes for a file are assigned a time to be flushed. If the file is modified before the flush time, then the flush time is extended by the time since the last modification (under the assumption that files that changed recently are likely to change soon). There is a minimum and maximum flush time extension for regular files and for directories. Setting `actimeo=`*n* sets flush time to *n* seconds for both regular files and directories.

Setting `actimeo=0` disables attribute caching on the client. This means that every reference to attributes will be satisfied directly from the server though file data will still be cached. While this guarantees that the client always has the latest file attributes from the server, it has an adverse effect on performance through additional latency, network load, and server load.

Setting the `noac` option also disables attribute caching, but has the further effect of disabling client write caching. While this guarantees that data written by an application will be written directly to a server, where it can be viewed

immediately by other clients, it has a significant adverse effect on client write performance. Data written into memory-mapped file pages (mmap(2)) will not be written directly to this server.

EXAMPLES

EXAMPLE 1 Mounting An NFS File System

To mount an NFS file system:

```
example# mount serv:/usr/src /usr/src
```

EXAMPLE 2 Mounting An NFS File System Read-Only With No Suid Privileges

To mount an NFS file system read-only with no suid privileges:

```
example# mount -r -o nosuid serv:/usr/src /usr/src
```

EXAMPLE 3 Mounting An NFS File System Over Version 2, With The UDP Transport

To mount an NFS file system over Version 2, with the UDP transport:

```
example# mount -o vers=2,proto=udp serv:/usr/src /usr/src
```

EXAMPLE 4 Mounting An NFS File System Using An NFS URL

To mount an NFS file system using an NFS URL (a canonical path):

```
example# mount nfs://serv/usr/man /usr/man
```

EXAMPLE 5 Mounting An NFS File System Forcing Use Of The Public File Handle

To mount an NFS file system and force the use of the public file handle and an NFS URL (a canonical path) that has a non 7–bit ASCII escape sequence:

```
example# mount -o public nfs://serv/usr/%A0abc /mnt/test
```

EXAMPLE 6 Mounting An NFS File System Using A Native Path

To mount an NFS file system using a native path (where the server uses colons (":") as the component separator) and the public file handle:

```
example# mount -o public serv:C:doc:new /usr/doc
```

EXAMPLE 7 Mounting an NFS file system using AUTH_KERB authentication.

To mount an NFS file system using AUTH_KERB authentication:

```
example# mount –o sec=krb4 serv:/usr/src /usr/src
```

EXAMPLE 8 Mounting a replicated set of NFS file systems with the same pathnames.

To mount a replicated set of NFS file systems with the same pathnames:

```
example# mount serv–a,serv–b,serv–c:/usr/man /usr/man
```

EXAMPLE 9 Mounting a replicated set of NFS file systems with different pathnames.

To mount a replicated set of NFS file systems with different pathnames:

```
example# mount serv–x:/usr/man,serv–y:/var/man,nfs://serv-z/man /usr/man
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The –o quota option has been removed; and the nodevices and nopriv options have been added.

Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Trusted Solaris security policy applies when mounting.

mount must run with an effective UID of 0 and with the sys_mount and net_privaddr privileges. To succeed in all cases, mount also needs: file_mac_read, file_dac_read, file_mac_write, file_dac_write, file_mac_search, file_dac_search, proc_setsl, and sys_trans_label.

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as ADMIN_LOW.

Objects still have CMW labels, and CMW labels still include the IL component: IL[SL]; however, the IL component is fixed at ADMIN_LOW.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return ADMIN_LOW.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always ADMIN_LOW, and cannot be set on any objects.
- Options related to information labels in the label_encodings(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

FILES

/etc/mnttab	table of mounted file systems
/etc/dfs/fstypes	default distributed file system type
/etc/vfstab	table of automatically mounted resources

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mountall`(1M), `mountd`(1M), `mkdir`(2), `mount`(2), `open`(2), `umount`(2), `mnttab`(4), `vfstab_adjunct`(4)

SunOS 5.7 Reference Manual

`mmap`(2), `lofs`(7FS), `attributes`(5)

NOTES

The sensitivity label mount-time attributes are only useful for mounts from NFS servers that are not labels-cognizant. The mount-time sensitivity label must always be equal to the assigned `def_sl`, if one is specified, in the NFS server's combination `tnrhdb`(4)/ `tnrhtp`(4) entry. An unlabeled file system is always mounted at the sensitivity label specified for the unlabeled server in the trusted networking databases; if a different sensitivity label is specified at mount time, the mount fails.

An NFS server should not attempt to mount its own file systems. See `lofs`(7FS).

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on *the directory to which the symbolic link refers*, rather than being mounted on top of the symbolic link itself.

SunOS 4.X used the `biod` maintenance procedure to perform parallel read-ahead and write-behind on NFS clients. SunOS 5.X made `biod` obsolete with multi-threaded processing, which transparently performs parallel read-ahead and write-behind.

Since the root (`/`) file system is mounted read-only by the kernel during the boot process, only the `remount` option (and options that can be used in conjuction with `remount`) affect the root (`/`) entry in the `/etc/vfstab` file.

NAME

mount_pcfs – Mount pcfs file systems

SYNOPSIS

mount –F pcfs [*generic_options*] [–o *FSType-specific_options*] [–S *attribute_list*] *special* | *mount_point*

mount –F pcfs [*generic_options*] [–o *FSType-specific_options*] [–S *attribute_list*] *special mount_point*

DESCRIPTION

`mount` attaches an MS-DOS file system (`pcfs`) to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

If `mount` is invoked with *special* or *mount_point* as the only arguments, `mount` will search `/etc/vfstab` to fill in the missing arguments, including the *FSType-specific_options*; see `mount`(1M) for more details.

The *special* argument can be one of two special device file types:

- A floppy disk, such as `/dev/diskette0` or `/dev/diskette1`.
- A DOS logical drive on a hard disk expressed as *device-name*`:`*logical-drive*, where *device-name* specifies the special block device-file for the whole disk and *logical-drive* is either a drive letter (c through z) or a drive number (1 through 24). Examples are `/dev/dsk/c0t0d0p0:`*c* and `/dev/dsk/c0t0d0p0:`*1*.

The *special* device file type must have a formatted MS-DOS file system with either a 12-bit, 16-bit, or 32-bit File Allocation Table.

Security attributes can be specified at mount time, either with the –S option on the `mount` command line or in the `vfstab_adjunct`(4) file. See the `DESCRIPTION` in the `mount` man page for more about specifying security attributes.

To succeed, the `mount` command must have the `sys_mount` privilege and must run with an effective UID of `0`. Mandatory and discretionary read access is required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2). To succeed in all cases, `mount –F pcfs` needs the `file_mac_read` and `file_mac_write` privileges.

OPTIONS

generic_options
See `mount`(1M) for the list of supported options.

–o
Specify `pcfs` file system specific options. The following options are available:

`rw|ro`

Mount the file system read/write or read-only. The default is `rw`.

`foldcase|nofoldcase`

Force uppercase characters in filenames to lowercase when reading them from the filesystem. This is for compatibility with the previous behavior of `pcfs`. The default is `nofoldcase`.

–S *attribute_list*

See the definition of the –S option in the `OPTIONS` section of the `mount`(1M) man page.

SUMMARY OF TRUSTED SOLARIS CHANGES

Trusted Solaris security policy applies when mounting and unmounting file systems.

Except when merely listing mounted file systems and resources, `mount` must run with an effective UID of `0` and with the `sys_mount` privilege.

Mandatory and discretionary read access is required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2). To succeed in all cases, `mount –F pcfs` need the `file_mac_read` and `file_mac_write` privileges.

Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Trusted Solaris security policy applies when mounting.

FILES

`/etc/mnttab` table of mounted file systems

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mountall`(1M), `mount`(2), `mnttab`(4), `vfstab`(4), `vfstab_adjunct`(4)

SunOS 5.7 Reference Manual

`attributes`(5), `pcfs`(7FS)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_tmpfs – Mount tmpfs file systems

SYNOPSIS

mount [–F tmpfs] [–o size= *sz*] [–O] [–S *attribute_list*] *special mount_point*

DESCRIPTION

`tmpfs` is a memory-based file system which uses kernel resources relating to the VM system and page cache as a file system.

`mount` attaches a `tmpfs` file system to the file system hierarchy at the pathname location *mount_point*, which must already exist. If *mount_point* has any contents prior to the `mount` operation, these remain hidden until the file system is once again unmounted. The attributes (mode, owner, and group) of the root of the `tmpfs` filesystem are inherited from the underlying *mount_point*, along with some security attributes (sensitivity label, attribute flags), provided that those attributes are determinable. If not, the root's attributes are set to their default values.

The *special* argument is usually specified as `swap` but is in fact disregarded and assumed to be the virtual memory resources within the system.

Security attributes can be specified at mount time, either with the –S option on the `mount` command line or in the `vfstab_adjunct`(4) file. See the `DESCRIPTION` in the `mount` man page for more about specifying security attributes.

To succeed, the `mount` command must have the `sys_mount` privilege and must run with an effective UID of `0`. Mandatory and discretionary read access is required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2). To succeed in all cases, `mount –F hsfs` also needs: `file_mac_read`, `file_dac_read`, `file_mac_write`, `file_dac_write`, `file_mac_search`, `file_dac_search`, `net_privaddr`, `proc_setsl`, `proc_setil`, and `sys_trans_label`.

OPTIONS

–o Specify `ufs` file system specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

`size=`*sz* The *sz* argument controls the size of this particular `tmpfs` file system. If the argument is has a 'k' suffix, the number will be interpreted as a number of kilobytes. An 'm' suffix will be interpreted as a number of megabytes. No suffix is

	interpreted as bytes. In all cases, the actual size of the file system is the number of bytes specified, rounded up to the physical pagesize of the system.
`suid` | `nosuid`	Setuid execution allowed or disallowed. The default is `suid`. `nosuid` without an explicit `devices` implies `nodevices`.
`devices` | `nodevices`	Allow (disallow) access to character and block devices. The default is `devices`.
	Note: In the Trusted Solaris environment, device special files are typically located only in the `/dev` and `/devices` directories in the root file system. All other file systems should be mounted with the `nodevices` option to prevent recognition of devices that may reside in any other directories. The recognition of devices is also affected by the use of the `devices` or `nodevices` options to the `share`(1M) command, either on the command line or in the `dfstab`(4) file.
`priv` | `nopriv`	Forced privileges on executables are allowed or disallowed. The default is `priv`. The recognition of forced privileges is also affected by the use of the `priv` or `nopriv` option to the `share`(1M) command, either on the command line or in the `dfstab`(4) file.

–O Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error "device busy".

–S *attribute_list* See the definition of the –S option in the OPTIONS section of the mount(1M) man page.

SUMMARY OF TRUSTED SOLARIS CHANGES

The nodevices and nopriv options have been added. Trusted Solaris security policy applies when mounting and unmounting file systems.

mount must run with an effective UID of 0 and with the sys_mount privilege. To succeed in all cases, mount also needs: file_mac_read, file_dac_read, file_mac_write, file_dac_write, file_mac_search, file_dac_search, net_privaddr, proc_setsl, proc_setil, and sys_trans_label.

Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Trusted Solaris security policy applies when mounting.

FILES

/etc/mnttab table of mounted file systems

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

mount(1M), mkdir(2), mount(2), open(2), umount(2), mnttab(4), vfstab_adjunct(4)

SunOS 5.7 Reference Manual

attributes(5), tmpfs(7FS)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_ufs – Mount ufs file systems

SYNOPSIS

mount –F ufs [*generic_options*] [–o *specific_options*] [–O] [–S *attribute_list*] *special* | *mount_point*

mount –F ufs [*generic_options*] [–o *specific_options*] [–O] [–S *attribute_list*] *special* *mount_point*

DESCRIPTION

The `mount` utility attaches a ufs file system to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

If `mount` is invoked with *special* or *mount_point* as the only arguments, `mount` will search `/etc/vfstab` to fill in the missing arguments, including the *specific_options*. See `mount`(1M) for more details.

If *special* and *mount_point* are specified without any *specific_options*, the default is `rw`.

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

Security attributes can be specified at mount time, either with the –S option on the `mount` command line or in the `vfstab_adjunct`(4) file. See the `DESCRIPTION` in the `mount`(1M) man page for more about specifying security attributes.

Except when merely listing mounted file systems and resources, `mount` must run with an effective UID of `0` and with the `sys_mount` privilege.

To succeed in all cases, `mount` needs: `file_mac_read`, `file_dac_read`, `file_mac_write`, `file_dac_write`, `file_mac_search`, `file_dac_search`, `net_privaddr`, `proc_setsl`, `proc_setil`, `sys_mount`, and `sys_trans_label`.

OPTIONS

See `mount`(1M) for the list of supported *generic_options*.

The following options are supported:

–o *specific_options*

Specify `ufs` file system specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

`noatime`

By default, the file system is mounted with normal access time (`atime`) recording. If `noatime` is specified, the file system will ignore access time updates on files, except when they coincide with updates to the `ctime` or `mtime`. See `stat`(2). This option reduces disk activity on file systems where access times are unimportant (for example, a Usenet news spool).

`noatime` turns off access time recording regardless of `dfratime` or `nodfratime`.

`devices` | `nodevices`

Allow (disallow) opens on character and block devices. The default is `devices`.

Note: In the Trusted Solaris environment, device special files are typically located only in the `/dev` and `/devices` directories in the root file system. All other file systems should be mounted with the `nodevices` option to prevent recognition of devices that may reside in any other directories.

`dfratime` | `nodfratime`

By default, writing access time updates to the disk may be deferred (`dfratime`) for the file system until the disk is accessed for a reason other than updating access times. `nodfratime` disables this behavior.

`f`

Fake an `/etc/mnttab` entry, but do not actually mount any file systems. Parameters are not verified.

`forcedirectio` | `noforcedirectio`

If `forcedirectio` is specified and supported by the file system, then for the duration of the mount forced direct I/O will be used. If the filesystem is mounted using `forcedirectio`, then data is transferred directly between user address space and the disk. If the filesystem is mounted using `noforcedirectio`, then data is buffered in kernel address space when data is transferred between user address space and the disk. `forcedirectio` is a performance option that benefits only from large sequential data transfers. The default behavior is `noforcedirectio`.

`global` | `noglobal`

If `global` is specified and supported on the file system, and the system in question is part of a cluster, the file system will be globally visible on all nodes of the cluster. If `noglobal` is specified, the mount will not be globally visible. The default behavior is `noglobal`.

`intr` | `nointr`

Allow (do not allow) keyboard interrupts to kill a process that is waiting for an operation on a locked file system. The default is `intr`.

`largefiles` | `nolargefiles`

If `nolargefiles` is specified and supported by the file system, then for the duration of the mount it is guaranteed that all regular files in the file system have a size that will fit in the smallest object of type `off_t` supported by the system performing the mount. The mount will fail if there are any files in the file system not meeting this criterion. If `largefiles` is specified, there is no such guarantee. The default behavior is `largefiles`.

If `nolargefiles` is specified, `mount` will fail for `ufs` if the file system to be mounted has contained a large file (a file whose size is greater than or equal to 2 Gbyte) since the last invocation of `fsck` on the file system. The large file need not be present in the file system at the time of the mount for the mount to fail; it could have been created previously and destroyed. Invoking `fsck` [see `fsck_ufs`(1M)] on the file system will reset the file system state if no large files are present. After invoking `fsck`, a successful mount of the file system with `nolargefiles` specified indicates the absence of large files in the file system; an unsuccessful mount attempt indicates the presence of at least one large file.

`logging` | `nologging`

If `logging` is specified, then logging is enabled for the duration of the mounted file system. Logging is the process of storing transactions (changes that make up a complete UFS operation) in a log before the transactions are applied to the file system. Once a transaction is stored, the transaction can be applied to the file system later. This prevents file systems from becoming inconsistent, therefore eliminating the need to run `fsck`. And, because `fsck` can be bypassed, logging reduces the time required to reboot a system if it crashes, or after an unclean halt. The default behavior is `nologging`.

The log is allocated from free blocks on the file system, and is sized approximately 1 Mbyte per 1 Gbyte of file system, up to a maximum of 64 Mbytes. Logging can be enabled on any UFS, including root (`/`). The log created by UFS logging is continually flushed as it fills up. The log is totally flushed when the file system is unmounted or as a result of the `lockfs -f` command.

`m`

Mount the file system without making an entry in `/etc/mnttab`.

`onerror=`*action*

This option specifies the action that UFS should take to recover from an internal inconsistency on a file system. Specify *action* as `panic`, `lock`, or `umount`. These values cause a forced system shutdown, a file system lock to be applied to the file system, or the file system to be forcibly unmounted, respectively. The default is `panic`.

`priv` | `nopriv`

Forced privileges on executables are allowed or disallowed. The default is `priv`.

`quota`

This option is not supported in Trusted Solaris; any attempt to set this option is ignored. Quotas are turned on for the file system.

`remount`

Remounts a read-only file system as read-write (using the `rw` option). This option can be used only in conjunction with the `f`, `logging`|`nologging`, `m`, and `noatime` options. This option works only on currently mounted read-only file systems.

`rq`

Read-write with quotas turned on. Equivalent to `rw, quota`.

`ro` | `rw`

Read-only or read-write. Default is `rw`.

`suid` | `nosuid`

Allow or disallow `setuid` execution. The default is `suid`. This option can also be used when mounting devices.

–O

Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error “device busy”.

–S *attribute_list*

See the definition of the –S option in the `OPTIONS` section of the `mount`(1M) man page.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `o quota` option has been removed; the `nodevices` and `nopriv` options have been added.

Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that

do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Trusted Solaris security policy applies when mounting.

`mount` must run with an effective UID of `0` and with the `sys_mount` privilege.

To succeed in all cases, the `mount` command needs the privileges: `file_mac_read`, `file_dac_read`, `file_mac_write`, `file_dac_write`, `file_mac_search`, `file_dac_search`, `net_privaddr`, `proc_setsl`, `proc_setil`, `sys_mount`, and `sys_trans_label`.

FILES

`/etc/mnttab` table of mounted file systems

`/etc/vfstab` list of default parameters for each file system

`/etc/security/tsol/vfstab_adjunct`
mount-time attributes for file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mountall`(1M), `mountd`(1M), `mnttab`(4), `vfstab`(4), `vfstab_adjunct`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

Since the root (`/`) file system is mounted read-only by the kernel during the boot process, only the `remount` option (and options that can be used in conjunction with `remount`) affect the root (`/`) entry in the `/etc/vfstab` file.

NAME

in.named, named – Internet domain name server

SYNOPSIS

in.named [–d *debuglevel*] [–q] [–r] [–f] [–p *remote/local-port*] [–w *dirname*] [[–b | –c] *configfile*]

DESCRIPTION

`in.named` is the Internet domain name server. `in.named` spawns the `named-xfer` process whenever it needs to perform a zone transfer. See `named-xfer`(1M) .

The `in.named` name service is used by hosts on the Internet to provide access to the Internet distributed naming database. See *RFC 1034* and *RFC 1035* for more information on the Internet domain name system.

With no arguments, `in.named` reads the default configuration file `/etc/named.conf` for any initial data, and listens for queries. Any additional arguments beyond those shown in the `SYNOPSIS` section are interpreted as the names of boot files. If multiple boot files are specified, only the last is used.

The name server reads the boot file to obtain instructions on where to find its initial data.

In a Trusted Solaris system, `in.named` listens for input requests on a multilevel port (MLP) and sends responses to the DNS client at the sensitivity label of the client's request. Thus, though `in.named` runs at the sensitivity label `ADMIN_LOW` , it can accept requests at any sensitivity label. `in.named` can also serve DNS clients and communicate with other DNS name servers on either Trusted Solaris hosts or non-trusted hosts.

The DNS name server running on a Trusted Solaris machine is viewed as a supplier of public information, and the name database that it maintains is considered trusted. `in.named` requires the trusted path attribute, and it requires that the `/etc/named.boot` file, zone files, and other configuration files that it uses be at the sensitivity label `ADMIN_LOW` . As part of the name database, these files and their contents are also considered trusted; thus `in.named` can query any DNS name server specified in the files. The DNS name servers specified in these files may reside on either Trusted Solaris hosts or non-trusted hosts.

OPTIONS

–w *dirname*	Change the current working directory of `in.named` to *dirname* .
–b *bootfile*	Use bootfile rather than `/etc/named.conf` . This options allows filenames to begin with a leading dash.
–c *bootfile*	Use bootfile rather than `/etc/named.conf` . This options allows filenames to begin with a leading dash.

–d *level* — Print debugging information. *level* is a number indicating the level of messages printed.

–p *remote/local-port* — Use different, port numbers. The default is the standard port number as returned by `getservbyname`(3N) for service domain. The –p argument can specify up to two port numbers. The specification of two port numbers requires a ' / ' (slash) separator. In this case, the first port is used when contacting remote servers, and the second one is the service port bound by the local instance of `in.named` . This option is used mostly for debugging purposes.

–q — Trace all incoming queries. Note: this option is ignored in favor of the boot file directive, `options query-log` , when both options are used.

–r — Turns recursion off in the server. Answers can come only from local (primary or secondary) zones. This option can be used on root servers. Note: This option will probably be eventually abandoned in favor of the boot file directive, `options no-recursion` .

USAGE

/etc/named.conf File Directives

The following is a simple configuration file `/etc/named.conf` containing directives to guide the `in.named` process at startup time.

```
options {
             directory  "/usr/local/adm/named";
             pid-file   "/var/named/named.pid";
             named-xfer "/usr/sbin/named-xfer";
             forwarders       {
                      10.0.0.78;
                      10.2.0.78;
             };
             transfers-in     10;
             forward only;
             fake-iquery yes;
             pollfd-chunk-size 20;
      };

      logging {
             category lame-servers { null; };
             category cname  { null; };
      };
```

```
zone "." in {
        type hint;
        file "root.cache";
};

zone "cc.berkeley.edu" in {
        type slave;
        file "128.32.137.3";
        masters { 128.32.137.8; };
};

zone "6.32.128.in-addr.arpa" in {
        type slave;
        file "128.32.137.3";
        masters { 128.32.137.8; };
};
zone "0.0.127.in-addr.arpa" in {
        type master;
        file "master/db.127";
};

zone "berkeley.edu" in {
        type master;
        file "berkeley.edu.zone";
};

zone "32.128.in-addr.arpa" in {
        type master;
        file "ucbhosts.rev";
};
```

The configuration file consists of sections and comments. Sections end with a ' `;` ' and contain statements which are enclosed in ' `{ }` ' and may span multiple lines. The following sections are supported: `options` , `zone` , `server` , `logging` , `acl` , `include` , and `key` .

Comments Syntax

The following are examples of comments syntax in BIND 8.1:

```
/* This is a BIND comment as in C */
// This is a BIND comment as in C++
#  This is a BIND comment as in common Unix shells and perl
```

WARNING: you cannot use the semicolon character (`;`) to start a comment.

Options Section

The sytax of the options section is as follows:

```
options {
       [ directory path_name; ]
       [ named-xfer path_name; ]
       [ pid-file path_name; ]
       [ auth-nxdomain yes_or_no; ]
```

```
        [ fake-iquery yes_or_no; ]
        [ fetch-glue yes_or_no; ]
        [ multiple-cnames yes_or_no; ]
        [ notify yes_or_no; ]
        [ recursion yes_or_no; ]
        [ forward ( only | first ); ]
        [ forwarders { [ in_addr ; [ in_addr ; ... ] ] }; ]
        [ check-names ( master | slave | response ) ( warn | fail | ignore); ]
        [ allow-query { address_match_list }; ]
        [ allow-transfer { address_match_list }; ]
        [ listen-on [ port ip_port ] { address_match_list }; ]
        [ query-source [ address ( ip_addr | * ) ] [ port ( ip_port | * ) ] ; ]
        [ max-transfer-time-in number; ]
        [ transfer-format ( one-answer | many-answers ); ]
        [ transfers-in  number; ]
        [ transfers-out number; ]
        [ transfers-per-ns number; ]
        [ coresize size_spec ; ]
        [ datasize size_spec ; ]
        [ files size_spec ; ]
        [ stacksize size_spec ; ]
        [ clean-interval number; ]
        [ interface-interval number; ]
        [ scan-interval number; ]
        [ topology { address_match_list }; ]
      };
```

Definitions and Use of Options

The `options` section sets up global options to be used by BIND. This section may appear at only once in a configuration file; if more than one occurrence is found, the first occurrence determines the actual options used, and a warning will be generated. If there is no `options` section, an `options` block with each `option` set to its default will be used.

Pathnames

directory — The working directory of the server. Any non-absolute pathnames in the configuration file will be taken as relative to this directory. The default location for most server output files (for example, " `named.run` ") is this directory. If a directory is not specified, the working directory defaults to " `.` ", the directory from which the server was started. The directory specified should be an absolute path.

named-xfer — The pathname to the `named-xfer` program that the server uses for inbound zone transfers. If not specified, the default is operating system dependent, for example, “ `/usr/sbin/named-xfer` ”).

pid-file — The pathname of the file the server writes its process ID in. If not specified, the default is operating system dependent, but is usually " `/var/run/named.pid` " or " `/etc/named.pid`

". The pid-file is used by programs like " ndc " that want to send signals to the running nameserver.

Boolean Options

auth-nxdomain — If yes , then the AA bit is always set on NXDOMAIN responses, even if the server is not actually authoritative. The default is yes . Do not turn off auth-nxdomain unless you are sure you know what you are doing, as some older software will not like it.

fake-iquery — If yes , the server will simulate the obsolete DNS query type IQUERY . The default is no .

fetch-glue — If yes (the default), the server will fetch "glue" resource records it does not have when constructing the additional data section of a response. fetch-glue no can be used in conjunction with recursion no to prevent the server's cache from growing or becoming corrupted (at the cost of requiring more work from the client).

multiple-cnames — If yes, then multiple CNAME resource records will be allowed for a domain name. The default is no . Allowing multiple CNAME records is against standards and is not recommended. Multiple CNAME support is available because previous versions of BIND allowed multiple CNAME records, and these records have been used for load balancing by a number of sites.

notify — If yes (the default), DNS NOTIFY messages are sent when a zone the server is authoritative for changes. The use of NOTIFY speeds convergence between the master and its slaves. Slave servers that receive a NOTIFY message and understand it will contact the master server for the zone and see if they need to do a zone transfer, and if they do, they will initiate it immediately. The notify option may also be specified in the zone section, in which case it overrides the options notify statement.

recursion — If yes , and a DNS query requests recursion , then the server will attempt to do all the work required to answer the query. If recursion is not on, the server will return a referral to the client if it doesn't know the answer. The default is yes . See also fetch-glue above.

Forwarding

The forwarding facility can be used to create a large sitewide cache on a few servers, reducing traffic over links to external name servers. It can also be used to allow queries by servers that do not have direct access to the Internet, but wish to look up exterior names anyway. Forwarding occurs only on those queries for which the server is not authoritative, and it does not have the answer in its cache.

forward — This option is only meaningful if the `forwarders` list is not empty. A value of first, the default, causes the server to query the `forwarders` first, and if that doesn't answer the question, the server will then look for the answer itself. If `only` is specified, the server will only query the `forwarders` .

forwarders — Specifies the IP addresses to be used for forwarding. The default is the empty list (no forwarding).

Future versions of BIND 8 will provide a more powerful forwarding system. The syntax described above will continue to be supported.

Name Checking

The server can check domain names based upon their expected client contexts. For example, a domain name used as a hostname can be checked for compliance with the valid hostnames defined in the RFC s. Three checking methods are available:

ignore — No checking is done.

warn — Names are checked against their expected client contexts. Invalid names are logged, but processing continues normally.

fail — Names are checked against their expected client contexts. Invalid names are logged, and the offending data is rejected.

The server can check names in three areas: master zone files, slave zone files, and in responses to queries the server has initiated. If `check-names response fail` has been specified, and answering the client's question would require sending an invalid name to the client, the server will send a `REFUSED` response code to the client.

The defaults are:

```
check-names master fail;
check-names slave warn;
check-names response ignore;
```

`check-names` may also be specified in the zone section, in which case it overrides the options `check-names` statement. When used in a zone section, the area is not specified (because it can be deduced from the zone type).

Access Control

Access to the server can be restricted based on the IP address of the requesting system. See address_match_list for details on how to specify IP address lists.

`allow-query` — Specifies which hosts are allowed to ask ordinary questions. `allow-query` may also be specified in the zone section, in which case it overrides the options `allow-query` statement. If not specified, the default is to allow queries from all hosts.

`allow-transfer` — Specifies which hosts are allowed to receive zone transfers from the server. `allow-transfer` may also be specified in the zonesection, in which case it overrides the options `allow-transfer` statement. If not specified, the default is to allow transfers from all hosts.

Interfaces

The interfaces and ports that the server will answer queries from may be specified using the `listen-on` option. `listen-on` takes an optional port, and an `address_match_list` . The server will listen on all interfaces allowed by the address match list. If a port is not specified, port 53 will be used.

Multiple listen-on statements are allowed. For example,

```
listen-on { 5.6.7.8; };
listen-on port 1234 { !1.2.3.4; 1.2/16; };
```

If no `listen-on` is specified, the server will listen on port 53 on all interfaces.

Query Address

If the server does not know the answer to a question, it will query other name servers. `query-source` specifies the address and port used for such queries. If address is `*` or is omitted, a wildcard IP address (`INADDR_ANY`) will be used. If port is `*` or is omitted, a random unprivileged port will be used. The default is:

```
query-source address * port *;
```

Note: `query-source` currently applies only to UDP queries; TCP queries always use a wildcard IP address and a random unprivileged port.

Zone Transfers

max-transfer-time-in — Inbound zone transfers (`named-xfer` processes) running longer than this many minutes will be terminated. The default is `120` minutes.

transfer-format — The server supports two zone transfer methods. `one-answer` uses one DNS message per resource record transferred. `many-answers` packs as many resource records as possible into a message.

`many-answers` is more efficient, but is only known to be understood by BIND 8.1 and patched versions of BIND 4.9.5. The default is `one-answer` . `transfer-format` may be overridden on a per-server basis by using the server section.

transfers-in — The maximum number of inbound zone transfers that can be running concurrently. The default value is 10. Increasing `transfers-in` may speed up the convergence of slave zones, but it also may increase the load on the local system.

transfers-out — This option will be used in the future to limit the number of concurrent outbound zone transfers. It is checked for syntax, but is otherwise ignored.

transfers-per-ns — The maximum number of inbound zone transfers (`named-xfer` processes) that can be concurrently transferring from a given remote name server. The default value is 2. Increasing `transfers-per-ns` may speed up the convergence of slave zones, but it also may increase the load on the remote name server. `transfers-per-ns` may be overridden on a per-server basis by using the transfers statement in the server section.

Resource Limits

The server's usage of many system resources can be limited. Some operating systems do not support some of the limits, and a warning will be generated if an unsupported limit is set in the configuration file.

Scaled values are allowed when specifying resource limits. For example, `1G` can be used instead of `1073741824` to specify a limit of one gigabyte, unlimited requests unlimited use, or the maximum available amount. Default uses the limit that was in force when the server was started. See `ulimit`(1) for a discussion of `ulimit -a` (ksh only) for defaults.

coresize — The maximum size of a core dump. The default is system dependent.

datasize — The maximum amount of data memory the server may use. The default is system dependent.

files — The maximum number of files that the server may have open concurrently. The default is system dependent.

stacksize — The maximum amount of stack memory the server may use. The default is system dependent.

Topology

All other things being equal, when the server chooses a name server to query from a list of name servers, it prefers the one that is topologically closest to itself. The topology statement takes an `address_match_list` and interprets it in a special way. Each top-level list element is assigned a distance. Non-negated elements get a distance based on their position in the list, where the closer the match is to the start of the list, the shorter the distance is between it and the server. A negated match will be assigned the maximum distance from the server. If there is no match, the address will get a distance which is further than any non-negated list element, and closer than any negated element. For example,

```
topology {
     10/8;
     !1.2.3/24;
     { 1.2/16; 3/8; };
};
```

will prefer servers on network 10 the most, followed by hosts on network 1.2.0.0 (netmask 255.255.0.0) and network 3, with the exception of hosts on network 1.2.3 (netmask 255.255.255.0), which is preferred least of all. The default topology is

```
topology { localhost; localnets; };
```

The Server Section

The syntax of the server section is as follows:

```
server ip_addr {
       [ bogus yes_or_no; ]
       [ transfers number; ]
       [ transfer-format ( one-answer | many-answers ); ]
       [ keys { key_id [key_id ... ] }; ]
};
```

The server statement defines the characteristics to be associated with a remote name server.

If you discover that a server is giving out bad data, marking it as bogus will prevent further queries to it. The default value is `no` .

The server supports two zone transfer methods. The first, `one-answer` , uses one DNS message per resource record transferred. `many-answers` packs as many resource records as possible into a message. `many-answers` is more efficient, but is only known to be understood by BIND 8.1 and patched

versions of BIND 4.9.5. You can specify which method to use for a server with the `transfer-format` option. If `transfer-format` is not specified, the `transfer-format` specified by the options statement will be used.

The transfers will be used in a future release of the server to limit the number of concurrent inbound zone transfers from the specified server. It is checked for syntax but is otherwise ignored.

The `keys` statement is intended for future use by the server. It is checked for syntax but is otherwise ignored.

The Zone Section

The syntax of the zone section is as follows:

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
       type master;
       file path_name;
       [ check-names ( warn | fail | ignore ); ]
       [ allow-update { address_match_list }; ]
       [ allow-query { address_match_list }; ]
       [ allow-transfer { address_match_list }; ]
       [ notify yes_or_no; ]
       [ also-notify { ip_addr; [ ip_addr; ... ] };
};

zone domain_name [ ( in | hs | hesiod | chaos ) ] {
       type ( slave | stub );
       [ file path_name; ]
       masters { ip_addr; [ ip_addr; ... ] };
       [ check-names ( warn | fail | ignore ); ]
       [ allow-update { address_match_list }; ]
       [ allow-query { address_match_list }; ]
       [ allow-transfer { address_match_list }; ]
       [ max-transfer-time-in number; ]
       [ notify yes_or_no; ]
       [ also-notify { ip_addr; [ ip_addr; ... ] };
};

zone . [ ( in | hs | hesiod | chaos ) ] {
       type hint;
       file path_name;
       [ check-names ( warn | fail | ignore ); ]
};
```

Zone types are defined as follows:

master The master copy of the data in a `zone` .

slave A `slave zone` is a replica of a `master zone` . The masters list specifies one or more IP addresses that the `slave` contacts to update its copy of the `zone` . If file is specified, then the replica will be written to the file. Use of file is recommended, since it often speeds server startup and eliminates a needless waste of bandwidth.

stub A `stub zone` is like a `slave zone`, except that it replicates only the NS records of a `master zone` instead of the entire zone.

hint The initial set of root name servers is specified using a `hint zone`. When the server starts up, it uses the root hints to find a root name server and get the most recent list of root name servers.

Note: previous releases of BIND used the term `primary` for a `master zone`, `secondary` for a `slave zone`, and `cache` for a `hint zone`.

The zone's name may optionally be followed by a `class`. If a `class` is not specified, `class in` is used.

Zone options are described as follows:

check-names	See `Name Checking`.
allow-query	See the description of `allow-query` in the `Access Control` section.
allow-update	Specifies which hosts are allowed to submit dynamic DNS updates to the server. The default is to deny updates from all hosts.
allow-transfer	See the description of `allow-transfer` in the `Access Control` section.
max-transfer-time-in	See the description of `max-transfer-time-in` in the `Zone Transfers` section.
notify	See the description of `notify` in the `Boolean Options` section.
also-notify	`also-notify` is only meaningful if `notify` is active for this zone.

The set of machines that will receive a `DNS NOTIFY` message for this zone is made up of all the listed name servers for the zone (other than the primary master) plus any IP addresses specified with `also-notify`. `also-notify` is not meaningful for `stub` zones. The default is the empty list.

The Logging Section

The syntax of the `logging` section is as follows:

```
logging {
       [ channel channel_name {
         ( file path_name
            [ versions ( number | unlimited ) ]
            [ size size_spec ]
          | syslog ( kern | user | mail | daemon | auth | syslog | lpr |
                     news | uucp | cron | authpriv | ftp |
                     local0 | local1 | local2 | local3 |
                     local4 | local5 | local6 | local7 )
```

```
            | null );

          [ severity ( critical | error | warning | notice |
                       info  | debug [ level ] | dynamic ); ]
          [ print-category yes_or_no; ]
          [ print-severity yes_or_no; ]
          [ print-time yes_or_no; ]
        }; ]

        [ category category_name {
          channel_name; [ channel_name; ... ]
        }; ]
        ...
};
```

The `logging` statement configures a wide variety of logging options for the name server. Its channel phrase associates output methods, format options and severity levels with a name that can then be used with the category phrase to select how various classes of messages are logged.

Only one `logging` statement is used to define as many channels and categories as are wanted. If there are multiple `logging` statements in a configuration, the first defined determines the logging, and warnings are issued for the others. If there is no `logging` statement, the default logging configuration will be:

```
logging {
    category default { default_syslog; default_debug; };
    category panic { default_syslog; default_stderr; };
    category packet { default_debug; };
    category eventlib { default_debug; };
};
```

The Channel Phrase

All log output goes to one or more "channels"; you can make as many of them as you want.

Every `channel` definition must include a clause that says whether messages selected for the channel go to a file, to a particular `syslog` facility, or are discarded. It can optionally also limit the message severity level that will be accepted by the channel (default is " `info` "), and whether to include a named-generated time stamp, the category name and / or severity level (default is not to include any).

The word `null` as the destination option for the channel will cause all messages sent to it to be discarded; other options for the channel are meaningless.

The file clause can include limitations both on how large the file is allowed to become, and how many versions of the file will be saved each time the file is opened.

The size option for files is simply a hard ceiling on log growth. If the file ever exceeds the size, then named will just not write anything more to it until the file is reopened; exceeding the size does not automatically trigger a reopen. The default behavior is to not limit the size of the file.

If you use the version logfile option, then named will retain that many backup versions of the file by renaming them when opening. For example, if you choose to keep 3 old versions of the file "lamers.log" then just before it is opened lamers.log.1 is renamed to lames.log.2, lamers.log.0 is renamed to lamers.log.1, and lamers.log is renamed to lamers.log.0. No rolled versions are kept by default. The unlimited keyword is synonymous with 99 in current BIND releases.

The argument for the `syslog()` clause is a `syslog()` facility as described in the `syslog`(3) manual page. How `syslogd`(1M) will handle messages sent to this facility is described in the `syslog.conf`(4) manual page. If you have a system which uses a very old version of `syslog()` that only uses two arguments to the `openlog()` function, then this clause is silently ignored.

The severity clause works like the "priorities" to `syslog()`, except that they can also be used if you are writing straight to a file rather than using `syslog()`. Messages which are not at least of the severity level given will not be selected for the channel; messages of higher severity levels will be accepted.

If you are using `syslog()`, then the `syslog.conf` priorities will also determine what eventually passes through. For example, defining a channel facility and severity as daemon and debug but only logging `daemon.warning` by way of `syslog.conf` will cause messages of `severity info` and `notice` to be dropped. If the situation were reversed, with named writing messages of only warning or higher, then `syslogd` would print all messages it received from the channel.

The server can supply extensive debugging information when it is in debugging mode. If the server's global debug level is greater than zero, then debugging mode will be active. The global debug level is set either by starting the server with the –`d` option followed by a positive integer, or by sending the server the `SIGUSR1` signal (for example, by using "`ndc trace`"). The global debug level can be set to zero, and debugging mode turned off, by sending the server the `SIGUSR2` signal ("ndc notrace". All debugging messages in the server have a debug level, and higher debug levels give more more detailed output. Channels that specify a specific debug severity, for example:

```
channel specific_debug_level {
    file "foo";
```

```
    severity debug 3;
};
```

will get debugging output of level 3 or less any time the server is in debugging mode, regardless of the global debugging level. Channels with dynamic severity use the server's global level to determine what messages to print.

If `print-time` has been turned on, then the date and time will be logged. `print-time` may be specified for a `syslog( )` channel, but is usually pointless since `syslog( )` also prints the date and time. If `print-category` is requested, then the category of the message will be logged as well. Finally, if `print-severity` is on, then the severity level of the message will be logged. The `print-options` may be used in any combination, and will always be printed in the following order: `time, category, severity`. Here is an example where all three `print-options` are on:

```
28-Apr-1997 15:05:32.863 default: notice: Ready to answer queries.
```

There are four predefined channels that are used for default logging for `in.named` as follows. How they are used is described in the next section.

```
channel default_syslog {
    syslog daemon;        # send to syslog's daemon facility
    severity info;        # only send priority info and higher
};

channel default_debug {
    file "named.run";     # write to named.run in the working directory
    severity dynamic;     # log at the server's current debug level
};

channel default_stderr {  # writes to stderr
    file "<stderr>";      # this is illustrative only;
  # there's currently
                          # no way of specifying an internal file
                          # descriptor in the configuration language.
    severity info;        # only send priority info and higher
};

channel null {
    null;                 # toss anything sent to this channel
};
```

Once a channel is defined, it cannot be redefined. Thus you cannot alter the built-in channels directly, but you can modify the default logging by pointing categories at channels you have defined.

The Category Phase

There are many categories, so you can send the logs you want to see wherever you want, without seeing logs you do not want. If you do not specify a list of channels for a category, then log messages in that category will be sent to the default category instead. If do not specify a default category, the following "default default" is used:

```
category default { default_syslog; default_debug; };
```

For example, if you want to log security events to a file, but you also want keep the default logging behavior, specify the following:

```
channel my_security_channel {
    file "my_security_file";
    severity info;
};
category security { my_security_channel; default_syslog; default_debug; };
```

To discard all messages in a category, specify the `null` channel:

```
category lame-servers { null; };
category cname { null; };
```

The following categories are available:

default — The catch-all. Many things still are not classified into categories, and they all end up here. Also, if you do not specify any channels for a category, the default category is used instead. If you do not define the default category, the following definition is used:

```
category default { default_syslog; default_debug; };
```

config — High-level configuration file processing.

parser — Low-level configuration file processing.

queries — A short log message is generated for every query the server receives.

lame-servers — Messages like "Lame server on ..."

statistics — Statistics.

panic	If the server has to shut itself down due to an internal problem, it will log the problem in this category as well as in the problem's native category. If you do not define the panic category, the following definition is used: `category panic { default_syslog; default_stderr; };`
update	Dynamic updates.
ncache	Negative caching.
xfer-in	Zone transfers the server is receiving.
xfer-out	Zone transfers the server is sending.
db	All database operations.
eventlib	Debugging info from the event system. Only one channel may be specified for this category, and it must be a file channel. If you do not define the eventlib category, the following definition is used: `category eventlib { default_debug; };`
packet	Dumps of packets received and sent. Only one channel may be specified for this category, and it must be a file channel. If you do not define the packet category, the following definition is used: `category packet { default_debug; };`
notify	The NOTIFY protocol.
cname	Messages like "... points to a CNAME".
security	Approved/unapproved requests.
os	Operating system problems.
insist	Internal consistency check failures.
maintenance	Periodic maintenance events.
load	Zone loading messages.
response-checks	Messages arising from response checking, such as "Malformed response ...", "wrong ans. name ...", "unrelated additional info ...", "invalid RR type ...", and "bad referral ...".

The Key Section

The syntax of the key section is as follows:

```
key key_id {
    algorithm algorithm_id;
    secret secret_string;
};
```

The key section defines a key ID which can be used in a server section to associate an authentication method with a particular name server.

A key ID must be created with the key statement before it can be used in a server definition.

The algorithm_id is a string that specifies a security / authentication algorithm. secret_string is the secret to be used by the algorithm.

The key statement is intended for future use by the server. It is checked for syntax but is otherwise ignored.

The Include Section

The syntax of the include section is as follows:

```
include path_name;
```

The include statement inserts the specified file at the point that the include statement is encountered. It cannot be used within another statement, though, so a line such as acl internal_hosts { "include internal_hosts.acl" } is not allowed. Use include to break the configuration up into easily-managed chunks. For example:

```
include "/etc/security/keys.bind";
include "/etc/acls.bind";
```

could be used at the top of a BIND configuration file in order to include any ACL or key information.

Be careful not to type " #include ", like you would in a C program, because " # " is used to start a comment.

The ACL Format

The syntax of the ACL section is as follows:

```
acl name {
    address_match_list
};
```

The `acl` statement creates a named address match list. It gets its name from a primary use of address match lists: Access Control Lists (ACL s).

Note that an address match list's name must be defined with `acl` before it can be used elsewhere; no forward references are allowed.

The following ACL s are built-in:

any	Allows all hosts.
none	Denies all hosts.
localhost	Allows the IP addresses of all interfaces on the system.
localnets	Allows any host on a network for which the system has an interface.

Zone File Format

The zone files are also known as the authoritative master files (data files) for a zone. In the boot file, references were made to these files as part of the specification of any primary directives.

Two classes of entries populate the zone files, directives and resource records. The start of the zone file is likely to contain one or two directives that establish a context that modifies the way subsequent records are interpreted.

Resource records for a zone determine how a zone is managed by establishing zone characteristics. For example, one type of zone record establishes the zone's mailbox information.

The very first record of each zone file should be a Start-of-Authority record (SOA) for a zone. A multiple-line SOA record is presented below. The meaning of the values in this sample will become clearer with the help of a list that describes the purpose of each field in the zone record (see the SOA list subitem under the `rr-type` list item in, Format of Resource Records in Zone Files).

```
@ IN SOA ucbvax.Berkeley.EDU. rwh.ucbvax.Berkeley.EDU. (
1989020501   ;serial
10800        ;refresh
3600         ;retry
3600000      ;expire
86400 )      ;minimum
```

Resource records normally end at the end of a line, but may be continued across lines between opening and closing parentheses (as demonstrated by the preceding sample).

Comments are introduced by semicolons. They continue to the end of the line.

Directives in Zone Files

There are two control directives that help determine how the zone file is processed, `$INCLUDE` and `$ORIGIN` .

The `$INCLUDE` directive refers to still another file within which zone characteristics are described. Such files typically contain groups of resource records, but they may also contain further directives.

The `$ORIGIN` directive establishes a current origin that is appended to any domain values that do not end with a ‘ . ’ (dot). The placeholder domain represents the first resource record field as shown in Format of Resource Records in Zone Files.The format for these directives is:

```
$INCLUDE filename opt-current-domain
$ORIGIN current-domain
```

where:

current-domain	Specifies the value of the current origin that remains in effect for this configuration file unless a subsequent `$ORIGIN` directive overrides it for the remaining portion of the file.
filename	Specifies a file, the contents of which are, in effect, incorporated into the configuration file at the location of the corresponding `$INCLUDE` directive.
opt-current-domain	Optionally defines a current origin that is applicable only to the records residing in the specified file in the corresponding `$INCLUDE` directive. This directive overrides the origin given in a preceding `$ORIGIN` directive, but only for the scope of the included text. See also `current-domain` .

Neither the opt-current-domain argument of `$INCLUDE` nor the `$ORIGIN` directive in the included file can affect the current origin in effect for the remaining records in the main configuration file (as defined by those `$ORIGIN` directives that reside there).

Format of Resource Records in Zone Files

The format of the resource records is:

```
domain opt-ttl opt-class rr-type rr-data...
```

where:

domain — Specifies the domain being described by the current line and any following lines that lack a value for this field. Beware of any domain values that you enter without full qualification, because the value of the current origin will be appended to them. The value of the current origin is appended when domain does not end with a dot.

A domain value specified as the symbol `@` is replaced with the value of the current origin. The `current-domain` or any locally-overriding `opt-current-domain` value is used as its replacement. (For a discussion of these placeholders, see the earlier discussion of the `$ORIGIN` and `$INCLUDE` directives.)

A domain value specified as a ‘ `.` ’ (dot) represents the root.

opt-ttl — Specifies the number of seconds corresponding to the `time-to-live` value applicable to the zone characteristic that is defined in the remaining fields. This field is optional. It defaults to zero. Zero is interpreted as the minimum value specified in the SOA record for the zone.

opt-class — Specifies the object address type; currently only one type is supported, `IN` , for objects connected to the Internet.

rr-type rr-data ... — Specifies values that describe a zone charateristic. Permissible `rr-type` and other field values are listed below. The field values are listed in the order that they must appear.

A address

Specifies the host address (in `dotted-quad` format). DCE or AFS server.

CNAME canonical-name

Specifies in a `domain-name` format the canonical name for the alias (domain).

HINFO cpu-type OS-type

Host information supplied in terms of a CPU type and an OS type.

MX preference mail-exchanger

Specifies in `domain-name` format a mail exchanger preceded by a preference value (between 0 and 32767), with lower numeric values representing higher logical preferences.

NS authoritative-server

Specifies in `domain-name` format an authoritative name server.

NULL

Specifies a null zone record.

PTR domain-pointer

Specifies in `domain-name` format a domain name pointer.

RP mailbox txt-referral

Offers details about how to reach a responsible person for the domain name.

`retry` `expire` `ttl`

SOA host-domain maintainer-addr serial- no refresh

Establishes the start of a zone of authority in terms of the domain of the originating host (host-domain), the domain address of the maintainer (maintainer-addr), a serial number (serial-no), the refresh period in seconds (refresh), the retry period in seconds (retry), the expiration period in seconds (expire), and the minimum time-to-live period in seconds (ttl). See RFC 1035.

The serial number should be changed each time the master file is changed. Secondary servers check the serial number at intervals specified by the refresh time in seconds; if the serial number changes, a zone transfer will be done to load the new data.

If a master server cannot be contacted when a refresh is due, the retry time specifies the interval at which refreshes should be attempted. If a master server cannot be contacted within the interval given by the expire time, all data from the zone is discarded by secondary servers. The minimum value is the time-to-live used by records in the file with no explicit time-to-live value.

The serial number can be given as a dotted number. However, this is a very unwise thing to do, since the translation to normal integers is via concatenation rather than multiplication and addition. You could spell out the year, month, day of month, and 0..99 version number and still fit it inside the unsigned 32-bit size of this field.

This strategy should work for the forseeable future (but is questionable after the year 4293).

For more detailed information, see *RFC 883* .

rr-data ... See the description of `rr-type` .

Consult *Name Server Operations Guide for BIND* for further information about the supported types of resource records.

EXIT STATUS

The `in.named` process returns the following exit values:

`0` Successful completion.

`1` An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.named` accepts requests at any sensitivity label and replies at the sensitivity label of the client's request. `in.named` can serve DNS clients and can communicate with other DNS servers that are on Trusted Solaris hosts or non-trusted hosts.

Files used by `in.named` should be protected from unauthorized access by having the sensitivity label `ADMIN_LOW` .

Invoking `in.named` requires the trusted path attribute, an effective UID of 0, a process sensitivity label of `ADMIN_LOW` , and the following privileges: `net_mac_read` , `net_privaddr` , `net_upgrade_sl` , `proc_setclr` , `sys_trans_label` , `sys_net_config` , and `sys_config` .

FILES

`/etc/named.conf`	Name server configuration boot file.
`/etc/named.pid`	The process ID (on older systems).
`/var/tmp/named.run`	Debug output.
`/var/tmp/named.stats`	Nameserver statistics data.
`/var/tmp/nameddump.db`	Dump of the name servers database.
`/var/tmp/named.pid`	The process ID (on newer systems).

These files have a sensitivity label of `ADMIN_LOW` .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`resolver`(3N)

SunOS 5.7 Reference Manual

`kill`(1) , `signal`(3B) , `resolv.conf`(4) , `attributes`(5)

Braden, R. (Editor), *Requirements for Internet Hosts - Applications and Support, RFC 1123, Internet Engineering* Task Force - Network Working Group, October 1989.

Mockapetris, Paul, *Domain Names - Concepts and Facilities, RFC 1034,* , Network Information Center, SRI International, Menlo Park, Calif., November 1987.

Mockapetris, Paul, *Domain Names - Implementation and Specification, RFC 1035* , Network Information Center, SRI International, Menlo Park, Calif., November 1987.

Mockapetris, Paul, *Domain System Changes and Observations, RFC 973* , Network Information Center, SRI International, Menlo Park, Calif., January 1986.

Partridge, Craig, *Mail Routing and the Domain System, RFC 974* , Network Information Center, SRI International, Menlo Park, Calif., January 1986.

Vixie, Paul, Dunlap, Keven J., Karels, Michael J., *Name Server Operations Guide for BIND* (public domain), Internet Software Consortium, 1995.

NOTES

The following signals have the specified effect when sent to the server process using the `kill`(1) command:

SIGHUP	Causes the server to read `/etc/named.conf` and reload the database.
SIGHUP	Also causes the server to check the serial number on all secondary zones. Normally the serial numbers are only checked at the intervals specified by the SOA record at the start of each zones-definition file.
SIGINT	Dumps the current database and cache to `/var/tmp/nameddump.db` .
SIGIOT	Dumps statistical data into `/var/tmp/named.stats` . Statistical data are appended to the file.
SIGUSR1	Turns on debugging at the lowest level when received the first time; receipt of each additional `SIGUSR1` signal causes the server to increment the debug level.
SIGUSR2	Turns off debugging completely.

SIGWINCH	Toggles logging of all incoming queries through the syslog system daemon. See `syslogd`(1M) .

NAME

ndd – Get and set driver configuration parameters

SYNOPSIS

ndd [–set] *driver parameter* [*value*]

DESCRIPTION

ndd gets and sets selected configuration parameters in some kernel drivers. Currently, ndd only supports the drivers that implement the TCP/IP Internet protocol family. Each driver chooses which parameters to make visible using ndd. Since these parameters are usually tightly coupled to the implementation, they are likely to change from release to release. Some parameters may be read-only.

If the –set option is omitted, ndd queries the named *driver*, retrieves the value associated with the specified *parameter*, and prints it. If the –set option is given, ndd passes *value*, which must be specified, down to the named *driver* which assigns it to the named *parameter*.

By convention, drivers that support ndd also support a special read-only *parameter* named "?" which can be used to list the parameters supported by the driver.

EXAMPLES

EXAMPLE 1 Getting Parameters Supported By The TCP Driver

To see which parameters are supported by the TCP driver, use the following command:

```
example% ndd /dev/tcp \?
```

The parameter name "?" may need to be escaped with a backslash to prevent its being interpreted as a shell meta character.

EXAMPLE 2 Disabling packet forwarding

The following command sets the value of the parameter *ip_forwarding* in the IP driver to zero. This disables IP packet forwarding.

```
example% ndd –set /dev/ip ip_forwarding 0
```

EXAMPLE 3 Viewing current IP forwarding table

To view the current IP forwarding table, use the following command:

```
example% ndd /dev/ip ip_ire_status
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The –set option must inherit the sys_net_config privilege to set driver parameters.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

SunOS 5.7 Reference Manual

`ioctl`(2), `attributes`(5), `arp`(7P), `ip`(7P), `tcp`(7P), `udp`(7P)

NOTES

The parameters supported by each driver may change from release to release. Like programs that read `/dev/kmem`, user programs or shell scripts that execute `ndd` should be prepared for parameter names to change.

The `ioctl()` command that `ndd` uses to communicate with drivers is likely to change in a future release. User programs should avoid making dependencies on it.

The meanings of many `ndd` parameters make sense only if you understand how the driver is implemented.

NAME

netstat – Show network status

SYNOPSIS

netstat [–anv]

netstat [–g | –m | –p | –s | –f *address_family*][–n] [–P *protocol*]

netstat {[–i] [–I *interface*] } [*interval*]

netstat –r [–anv]

netstat –R [–anv] [*system*] [*core*]

netstat –M [–ns]

netstat –D [–I *interface*]

DESCRIPTION

`netstat` displays the contents of various network-related data structures in various formats, depending on the options you select.

The first form of the command displays a list of active sockets for each protocol. The second form selects one from among various other network data structures. The third form shows the state of the interfaces. The fourth form displays the routing table, the fifth form displays the routing table with extended metric information, the sixth form displays the multicast routing table, and the seventh form displays the state of DHCP on one or all interfaces.

OPTIONS

–a — Show the state of all sockets and all routing table entries; normally, sockets used by server processes are not shown and only interface, host, network, and default routes are shown.

–f *address_family* — Limit statistics or address control block reports to those of the specified address_family, which can be one of:

- `inet` For the `AF_INET` address family
- `unix` For the `AF_UNIX` address family

–g — Show the multicast group memberships for all interfaces.

–i — Show the state of the interfaces that are used for TCP/IP traffic. See `ifconfig`(1M).

–m — Show the STREAMS statistics.

–n — Show network addresses as numbers. `netstat` normally displays addresses as symbols. This option may be used with any of the display formats.

–p — Show the address resolution (ARP) tables.

–r — Show the routing tables.

–s	Show per-protocol statistics. When used with the –M option, show multicast routing statistics instead.
–v	Verbose. Show additional information for the sockets and the routing table.
–I *interface*	Show the state of a particular interface. *interface* can be any valid interface such as ie0 or le0.
–M	Show the multicast routing tables. When used with the –s option, show multicast routing statistics instead.
–P *protocol*	Limit display of statistics or state of all sockets to those applicable to *protocol*.
–d	Show the state of all interfaces that are under Dynamic Host Configuration Protocol (DHCP) control.
–D	Show the status of DHCP configured interfaces.

OPERANDS

interval	If *interval* is specified, netstat displays interface information over the last *interval* seconds, repeating forever.

DISPLAYS

Active Sockets (First Form)

netstat [–anv] [*system*] [*core*]

The display for each active socket shows the local and remote address, the send and receive queue sizes (in bytes), the send and receive windows (in bytes), and the internal state of the protocol.

The symbolic format normally used to display socket addresses is either

hostname.*port*

when the name of the host is specified, or

network.port

if a socket address specifies a network but no specific host.

The numeric host address or network number associated with the socket is used to look up the corresponding symbolic hostname or network name in the hosts or networks database.

If the network or hostname for an address is not known (or if the –n option is specified), the numerical network address is shown. Unspecified, or "wildcard", addresses and ports appear as "*". For more information regarding the Internet naming conventions, refer to inet(7P).

TCP Sockets

The possible state values for TCP sockets are as follows:

`BOUND`	Bound, ready to connect or listen.
`CLOSED`	Closed. The socket is not being used.
`CLOSING`	Closed, then remote shutdown; awaiting acknowledgment.
`CLOSE_WAIT`	Remote shutdown; waiting for the socket to close.
`ESTABLISHED`	Connection has been established.
`FIN_WAIT_1`	Socket closed; shutting down connection.
`FIN_WAIT_2`	Socket closed; waiting for shutdown from remote.
`IDLE`	Idle, opened but not bound.
`LAST_ACK`	Remote shutdown, then closed; awaiting acknowledgment.
`LISTEN`	Listening for incoming connections.
`SYN_RECEIVED`	Initial synchronization of the connection under way.
`SYN_SENT`	Actively trying to establish connection.
`TIME_WAIT`	Wait after close for remote shutdown retransmission.

Network Data Structures (Second Form)

`netstat` [`–s` | `–g` | `–m` | `–p` | `–f` *address_family*] [`–P` *protocol*] [`–n`] [*system*] [*core*]

The form of the display depends upon which of the `–g`, `–m`, `–p`, or `–s` options you select.

`–g`	Displays the list of multicast group membership.
`–m`	Displays the memory usage, for example, STREAMS mblks.
`–p`	Displays the address resolution table. This is similar to `arp`(1M).
`–s`	Displays the statistics for the various protocol layers.

The statistics use the MIB specified variables. The defined values for `ipForwarding` are:

forwarding(1)	Acting as a gateway.
not-forwarding(2)	Not acting as a gateway.

If you specify more than one of these options, `netstat` displays the information for each one of them.

Interface Status (Third Form)

`netstat –i` | `–I` *interface* [*interval*] [*system*] [*core*]

The interface status display lists information for all current interfaces, one interface per line. If an interface is specified using the `–I` option, it displays information for only the specified interface.

The list consists of the interface name, `mtu` (maximum transmission unit, or maximum packet size) (see `ifconfig`(1M)), the network to which the interface is attached, addresses for each interface, and counter associated with the interface. The counters show the number of input packets, input errors, output packets, output errors, and collisions, respectively. For Point-to-Point interfaces, the Net / Dest field is the name or address on the other side of the link.

If the –`n` option is specified, the list displays the IP address instead of the interface name.

If an optional *interval* is specified, the output will be continuously displayed in *interval* seconds until interrupted by the user.

The input interface is specified using the –`I` option. In this case, the list only displays traffic information in columns; the specified interface is first, the total count is second. This column list has the format of:

```
     input   le0      output         input  (Total)   output
packets errs packets errs colls packets errs packets errs colls
227681  0    65947   1    502   261331  0    99597   1     502
10      0    0       0    0     10      0    0       0     0
8       0    0       0    0     8       0    0       0     0
10      0    2       0    0     10      0    2       0     0
```

If the input interface is not specified, the first interface of address family `inet` will be displayed.

Routing Table (Fourth Form)

`netstat` –`r` [–`anv`] [*system*] [*core*]

The routing table display lists the available routes and the status of each. Each route consists of a destination host or network, and a gateway to use in forwarding packets. The *flags* column shows the status of the route (`U` if "`up`"), whether the route is to a gateway (`G`), and whether the route was created dynamically by a redirect (`D`). If the –`a` option is specified, there will be routing entries with flags for combined routing and address resolution entries (`A`), broadcast addresses (`B`), and the local addresses for the host (`L`).

Interface routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface.

The `refcnt` column gives the current number of routes that share the same link layer address.

The `use` column displays the number of packets sent using a combined routing and address resolution (`A`) or a broadcast (`B`) route. For a local (`L`) route, this count is the number of packets received, and for all other routes it is the number of times the routing entry has been used to create a new combined route and address resolution entry.

The *interface* entry indicates the network interface utilized for the route.

Routing Table with Extended Metric Information (Fifth Form)

`netstat –R [ –anv ] [ system ] [ core ]`

This form is the same as that of `–r` with the following additional information. If a route displayed has extended metric (emetric) information, it is displayed in the next line indented by a tab. Since a route may have multiple emetrics, each is displayed one line at a time. The format of display is the same as the format of specification in `route`(1M); that is, each field includes a keyword and, if there is value to follow, an equal sign (`=`) and the value. The fields are separated by commas (`,`). Depending on the nature of the route (that is, local or remote) and how it was entered into the routing table, there may be no emetric available for a particular route. In that case, no emetric is displayed.

Multicast Routing Tables (Sixth Form)

`netstat –M [ –ns ] [ system ] [ core ]`

The multicast routing table consists of the virtual interface table and the actual routing table.

DHCP Interface Information (Seventh Form)

`netstat –D [ –I interface ]`

The DHCP interface information consists of the interface name, its current state, lease information (when the lease began, when it will expire, and when renewal begins), and counts of the number of protocol exchanges done on behalf of the interface.

Below is a sample command line and output from a host with five interfaces under DHCP control:

```
# netstat –D
Interface  Status       Sent Received  Rejects
le0        BOUND           1        1        0
         (Began,Expires,Renew) = (12/04/1996 18:08, 12/04/1996 19:08,
12/04/1996 18:38) qe0        BOUND           1        1        0
         (Began,Expires,Renew) = (12/04/1996 18:08, 12/04/1996 19:08,
12/04/1996 18:38) qe1        BOUND           1        1        0
         (Began,Expires,Renew) = (12/04/1996 18:08, 12/04/1996 19:08,
12/04/1996 18:38) qe2        BOUND           1        1        0
         (Began,Expires,Renew) = (12/04/1996 18:08, 12/04/1996 19:08,
12/04/1996 18:38) qe3        SELECTING        4        0        0
#
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The –R option requires the `net_rawaccess` privilege. The –R option must also run with a uid of `0`, or have the `file_dac_read` and `file_dac_write` privileges.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`ifconfig`(1M)

SunOS 5.7 Reference Manual

`iostat`(1M), `vmstat`(1M), `hosts`(4), `networks`(4), `protocols`(4), `services`(4), `attributes`(5)

NOTES

The kernel's tables can change while `netstat` is examining them, creating incorrect or partial displays.

If you need to examine network status information following a kernel crash, use the `crash`(1M) utility on the `savecore`(1M) output.

NAME

setfsattr, newsecfs – Set security attributes on an existing or newly created file system

SYNOPSIS

/usr/sbin/setfsattr {[–a *access-acl*] | [–l *sensitivity-level-range*] | [–m *MLD-prefix*] | [–p *allowed-privilege-set*] | [–P *forced-privilege-set*] | [–s *CMW -Label*] }…{*special* | *filesystem* }

/usr/sbin/newsecfs {[–a *access-acl*] | [–l *sensitivity-level-range*] | [–M] | [–m *MLD-prefix*] | [–o *newfs options*] [–p *allowed-privilege-set*] | [–P *forced-privilege-set*] | [–s *CMW -Label*] }…{*special* | *filesystem* }

DESCRIPTION

`setfsattr` changes the security attributes of a file system. The file system may be specified either as a *filesystem* or as *special* , the device on which the file system resides. *filesystem* must be in `/etc/vfstab` , and it must be unmounted before `setfsattr` is invoked on it. `setfsattr` requires at least one option be specified; if not, an error is returned.

`newsecfs` works similarly to `setfsattr` except that it runs `newfs`(1M) on the file system prior to setting the security attributes, then sets the label on the `lost+found` directory to `[ADMIN_HIGH]` .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–a *access-acl*
: Set the file system access ACL . The specified ACL must be a valid access ACL .

–l *sensitivity-level-range*
: Set the filesystem sensitivity level range, a semicolon-separated pair of sensitivity labels. The labels must be valid sensitivity labels for the system. The first in the pair is the minimum sensitivity label, and it must be dominated by the second label, the maximum sensitivity label. The default is `ADMIN_LOW;ADMIN_HIGH` .

–M
: Create the root directory of the file system as a multilevel directory (MLD). This option is available only with the `newsecfs` command.

–m *MLD-prefix*	Set the file system MLD prefix. The default is " .MLD. ". The MLD prefix is the string that disables multilevel directory translation in pathname lookup.
–o *newfs options*	Set the file system newfs options. The options must be exactly the same as those expected by the newfs(1M) command. This option is available only with newsecfs .
–p *allowed-privileges*	Set the file system allowed-privilege set, specified as a text-string of comma-separated privilege names. The privileges in the allowed set must include all privileges in the forced set, or the operation fails.
–P *forced-privileges*	Set the filesystem forced-privilege set, specified as a text string of comma-separated privilege names. All privileges in the forced set must also be in the allowed set, or the operation fails.
–s *CMW-Label*	Set the filesystem CMW label.

USAGE

To specify arguments that include semicolons or embedded spaces (such as for the –l and –o options), use quotes to enclose the arguments.

EXAMPLES

EXAMPLE 1 To set an access ACL

```
% setfsattr -a
\\"user:joni:rw-,user::rwx,group::r--,mask::rw-,other::---" filesystem
```

EXAMPLE 2 To create a new file system with a limited label range

To create a new file system with an allowable label range of Confidential to Secret, use this command:

```
% newsecfs -l 'confidential;secret' raw_device
```

RETURN VALUES

setfsattr exits with one of these values:

0 Success.

1 Failure.

SEE ALSO

Trusted Solaris 7 Reference Manual

`fork`(2)

SunOS 5.7 Reference Manual

`mkfs`(1M) , `newfs`(1M) , `terminfo`(4) , `attributes`(5)

NAME

nfsd – NFS daemon

SYNOPSIS

/usr/lib/nfs/nfsd [–a] [–c *#_conn*] [–l *listen_backlog*] [–p *protocol*] [–t *device*] [*nservers*]

DESCRIPTION

nfsd is the daemon that handles client file-system requests. Users must have the sys_nfs privilege to run this daemon.

The nfsd daemon is automatically invoked in run level 3 with the –a option.

By default nfsd will start over the TCP and UDP transports.

A previously invoked nfsd daemon started with or without options must be stopped before invoking another nfsd command.

OPTIONS

The following options are supported:

–a — Start a NFS daemon over all available connectionless and connection-oriented transports, including TCP and UDP.

–c *#_conn* — This sets the maximum number of connections allowed to the NFS server over connection-oriented transports. By default, the number of connections is unlimited.

–l — Set connection queue length for the NFS TCP over a connection-oriented transport. The default value is 32 entries.

–p *protocol* — Start a NFS daemon over the specified protocol.

–t *device* — Start a NFS daemon for the transport specified by the given device.

OPERANDS

The following operands are supported:

nservers — Set the maximum number of concurrent NFS requests that the server can handle. This concurrency is achieved by up to *nservers* threads created as needed in the kernel. *nservers* should be based on the load expected on this server. 16 is the usual number of *nservers*. If *nservers* is not specified, the maximum number of concurrent NFS requests will default to 1.

USAGE

If the NFS_PORTMON variable is set, then clients are required to use privileged ports (ports < IPPORT_RESERVED) in order to get NFS services. In the Trusted Solaris environment, this variable is set to 1 by default. This variable has been moved from the "nfs" module to the "nfssrv" module. To set the variable, edit the /etc/system file and add this entry:

```
set nfssrv:nfs_portmon = 1
```

EXIT STATUS

0 Daemon started successfully.

1 Daemon failed to start.

FILES

`.nfs`*XXX*	Client machine pointer to an open-but-unlinked file
`/etc/init.d/nfs.server`	Shell script for starting `nfsd`
`/etc/system`	System configuration information file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The `sys_nfs` and `net_mac_read` privileges are required to run this daemon. `NFS_PORTMON` has been set to `1` by default.

SEE ALSO

Trusted Solaris 7 Reference Manual

`mountd`(1M), `sharetab`(4)

SunOS 5.7 Reference Manual

`ps`(1), `system`(4), `attributes`(5)

NFS Administration Guide

NOTES

1. The NFS service uses kernel threads to process all of the NFS requests. Currently, system utilization associated with these threads is not charged to the `nfsd` process. Therefore, `ps`(1) can report `0` cpu time associated with the NFS daemon, even though NFS processing is taking place on the server.
2. Manually starting and restarting `nfsd` is not recommended. If it is necessary to do so, use the NFS server start/stop script (`/etc/init.d/nfs.server`). See *NFS Administration Guide* for more information.

NAME

nfsstat – NFS statistics

SYNOPSIS

nfsstat [–cmnrsz]

DESCRIPTION

`nfsstat` displays statistical information about the NFS and RPC (Remote Procedure Call), interfaces to the kernel. It can also be used to reinitialize this information. If no options are given the default is

```
nfsstat –cnrs
```

That is, display everything, but reinitialize nothing.

To succeed with no option or with any option other than –z, `nfsstat` requires MAC and DAC read access to `/dev/mem`. To succeed with the –z option, `nfsstat` requires MAC and DAC write access to `/dev/mem` and the `sys_config` privilege.

OPTIONS

–c Display client information. Only the client side NFS and RPC information will be printed. Can be combined with the –n and –r options to print client NFS or client RPC information only.

–m Display statistics for each NFS mounted file system. This includes the server name and address, mount flags, current read and write sizes, the retransmission count, and the timers used for dynamic retransmission. The `srtt` value contains the smoothed round trip time, the `dev` value contains the estimated deviation, and the `cur` value is the current backed-off retransmission value.

–n Display NFS information. NFS information for both the client and server side will be printed. Can be combined with the –c and –s options to print client or server NFS information only.

–r Display RPC information.

–s Display server information.

–z Zero (reinitialize) statistics. This option requires the `sys_config` privilege and can be combined with any of the above options to zero particular sets of statistics after printing them.

DISPLAYS

The server RPC display includes the following fields:

`calls` The total number of RPC calls received.

`badcalls` The total number of calls rejected by the RPC layer (the sum of `badlen` and `xdrcall` as defined below).

`nullrecv` The number of times an RPC call was not available when it was thought to be received.

`badlen` The number of RPC calls with a length shorter than a minimum-sized RPC call.

`xdrcall` The number of RPC calls whose header could not be XDR decoded.

`dupchecks` The number of RPC calls that looked up in the duplicate request cache.

`dupreqs` The number of RPC calls that were found to be duplicates.

The server NFS display shows the number of NFS calls received (`calls`) and rejected (`badcalls`), and the counts and percentages for the various calls that were made.

The client RPC display includes the following fields:

`calls` The total number of RPC calls made.

`badcalls` The total number of calls rejected by the RPC layer.

`badxids` The number of times a reply from a server was received which did not correspond to any outstanding call.

`timeouts` The number of times a call timed out while waiting for a reply from the server.

`newcreds` The number of times authentication information had to be refreshed.

`badverfs` The number of times the call failed due to a bad verifier in the response.

`timers` The number of times the calculated time-out value was greater than or equal to the minimum specified time-out value for a call.

`cantconn` The number of times the call failed due to a failure to make a connection to the server.

`nomem` The number of times the call failed due to a failure to allocate memory.

`interrupts` The number of times the call was interrupted by a signal before completing.

`retrans` The number of times a call had to be retransmitted due to a timeout while waiting for a reply from the server. Applicable only to RPC over connection-less transports.

The client NFS display shows the number of calls sent and rejected, as well as the number of times a `CLIENT` handle was received (`clgets`), the number of

times the `CLIENT` handle cache had no unused entries (`cltoomany`), as well as a count of the various calls and their respective percentages.

The `–m` option includes information about mount flags set by mount options, mount flags internal to the system, and other mount information. See `mount_nfs`(1M).

The following fields provide failover information:

`noresponse` How many times servers have failed to respond.

`failover` How many times a new server has been selected.

`remap` How may times files have been re-evaluated to the new server.

`currserver` Which server is currently providing NFS service. See the *NFS Administration Guide* for additional details.

The following mount flags are set by mount options:

`auth` `auth` has one of the following values:

- `none` No authentication.
- `unix` UNIX style authentication (UID, GID).
- `short` Shorthand UNIX style authentication.
- `des` `des` style authentication (encrypted timestamps).
- `krb4` `kerberos` style authentication.

`hard` Hard mount.

`soft` Soft mount.

`intr` Interrupts allowed on hard mount.

`nointr` No interrupts allowed on hard mount.

`noac` Client is not caching attributes.

`rsize` Read buffer size in bytes.

`wsize` Write buffer size in bytes.

`retrans` NFS retransmissions.

`nocto` No close-to-open consistency.

`llock` Local locking being used (no lock manager).

`grpid` System V group id inheritance.

`rpctimesync` RPC time sync.

The following mount flags are internal to the system:

`printed` "Not responding" message printed.

`down` Server is down.

`dynamic` Dynamic transfer size adjustment.

`link` Server supports links.

`symlink` Server supports symbolic links.

`readdir` Use `readdir` instead of `readdirplus`.

`acl` Server supports NFS_ACL. The following flags relate to additional mount information:

`vers` NFS version.

`proto` Protocol.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed with no option or with any option other than –`z`, `nfsstat` requires MAC and DAC read access to `/dev/mem`. To succeed with the –`z` option, `nfsstat` requires MAC and DAC write access to `/dev/mem` and the `sys_config` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount_nfs`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

Solaris Advanced Installation Guide

NFS Administration Guide

NAME

nis_cachemgr – NIS+ utility to cache location information about NIS+ servers

SYNOPSIS

/usr/sbin/nis_cachemgr [–i] [–v]

DESCRIPTION

The nis_cachemgr daemon maintains a cache of NIS+ directory objects and active servers for domains. It is responsible for locating servers for a domain on behalf of client processes. This improves performance because only one process has to search for servers. The cache contains location information necessary to contact the NIS+ servers. This includes transport addresses, information neeeded to authenticate the server, and a time to live field which gives a hint on how long the directory object can be cached. The cache helps to improve the performance of the clients that are traversing the NIS+ name space. nis_cachemgr should be running on all the machines that are using NIS+. However, it is not required that the nis_cachemgr program be running in order for NIS+ requests to be serviced.

The cache maintained by this program is shared by all the processes that access NIS+ on a machine. The cache is maintained in a file that is memory mapped (see mmap(2)) by all the processes. On start up, nis_cachemgr initializes the cache from the cold start file (see nisinit(1M)) and preserves unexpired entries that already exist in the cache file. Thus, the cache survives machine reboots.

The nis_cachemgr program is normally started from a system startup script. nisshowcache(1M) can be used to look at the cached objects and active servers.It must be started by a user with a UID of 0 and at a sensitivity label of ADMIN_LOW. Upon startup nis_cachemgr must inherit the net_mac_read and net_upgrade_sl privileges.

The nisprefadm(1M) command (see the SunOS 5.7 Reference Manual) can be used to control which NIS+ servers the nis_cachemgr program will try to select.

The nis_cachemgr program makes NIS+ requests under the NIS+ principal name of the host on which it runs. Before running nis_cachemgr, security credentials for the host should be added to the cred.org_dir table in the host's domain using nisaddcred(1M). Credentials of type DES will be needed if the NIS+ service is operating at security level 2 (see rpc.nisd(1M)). See the WARNINGS section, below. Additionally, a "keylogin –r " should be done on the machine.

OPTIONS

–i Force nis_cachemgr to ignore the previous cache file and reinitialize the cache from just the cold start file. By default, the cache manager initializes itself from both the cold start file and the old cache file, thereby maintaining the entries in the cache across machine reboots.

–v This flag sets verbose mode. In this mode, the nis_cachemgr program logs not only errors and warnings, but also additional status

messages. The additional messages are logged using `syslog`(3) with a priority of `LOG_INFO`.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `nis_cachemgr` must be started by a user with a UID of `0` and at a sensitivity level of `ADMIN_LOW`. At startup it must inherit the `net_mac_read` and `net_upgrade_sl` privileges.

FILES

`/var/nis/NIS_SHARED_DIRCACHE`	The shared cache file
`/var/nis/NIS_COLD_START`	The coldstart file
`/etc/init.d/rpc`	Initialization scripts for NIS+

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

SunOS 5.7 Reference Manual

`keylogin`(1), `nisaddcred`(1M), `nisinit`(1M), `nisshowcache`(1M), `mmap`(2), `syslog`(3), `nisfiles`(4), `attributes`(5)

DIAGNOSTICS

The `nis_cachemgr` daemon logs error messages and warnings using `syslog`(3). Error messages are logged to the `DAEMON` facility with a priority of `LOG_ERR`. Warning messages are logged with a priority of `LOG_WARNING`. Additional status messages can be obtained using the `–v` option.

NAME

rpc.nisd, nisd – NIS+ service daemon

SYNOPSIS

/usr/sbin/rpc.nisd [–ACDFhlv] [–Y [–B [–t *netid*]]] [–d *dictionary*] [–L *load*] [–S *level*]

DESCRIPTION

The rpc.nisd daemon is an RPC service that implements the NIS+ service. This daemon must be running on all machines which serve a portion of the NIS+ namespace. A Trusted Solaris 7 system must be the root master in the NIS+ configuration.

rpc.nisd is usually started from a system startup script. It must be started through a role that has a UID of 0 and run with a sensitivity label of ADMIN_LOW . (For example, the role might be assigned the predefined NIS+ security administration and NIS+ administration profiles.) Upon startup, rpc.nisd must inherit the net_mac_read , net_upgrade_sl , and proc_setsl privileges.

OPTIONS

–A Authentication verbose mode. The daemon logs all the authentication related activities to syslogd(1M) with LOG_INFO priority.

–C Open diagnostic channel on /dev/console .

–D Debug mode (don't fork).

–F Force the server to do a checkpoint of the database when it starts up. Forced checkpoints may be required when the server is low on disk space. This option removes updates from the transaction log that have propagated to all of the replicas.

–h Print list of options.

–u Allow updates from non-Trusted Solaris TCB clients.

–v Verbose. With this option, the daemon sends a running narration of what it is doing to the syslog daemon (see syslogd(1M)) at LOG_INFO priority. This option is most useful for debugging problems with the service (see also –A option).

–Y ypserve and other NIS (YP) compatibility is not supported in Trusted Solaris. Using this option may put the daemon in an unknown state.

–B ypserve and other NIS (YP) compatibility is not supported in Trusted Solaris. Using this option may put the daemon in an unknown state.

–t *netid* Use *netid* as the transport for communication between rpc.nisd and rpc.nisd_resolv . The default transport is ticots(7D) (tcp on SunOS 4.x systems).

–d *dictionary* Specify an alternate dictionary for the NIS+ database. The primary use of this option is for testing. Note that the

string is not interpreted, rather it is simply passed to the `db_initialize( )` function. See `nis_db`(3N) .

–L *number*
: Specify the "load" the NIS+ service is allowed to place on the server. The load is specified in terms of the *number* of child processes that the server may spawn. This *number must* be at least 1 for the callback functions to work correctly. The default is 128.

–S *level*
: Set the authorization security level of the service. The argument is a number between 0 and 2. By default, the daemon runs at security level 2.

  `0` Security level 0 is designed to be used for testing and initial setup of the NIS+ namespace. When running at level 0, the daemon does not enforce any access controls. Any client is allowed to perform any operation, including updates and deletions.

  `1` At security level 1, the daemon accepts both `AUTH_SYS` and `AUTH_DES` credentials for authenticating clients and authorizing them to perform NIS+ operations. This is not a secure mode of operation since `AUTH_SYS` credentials are easily forged. It should not be used on networks in which any untrusted users may potentially have access.

  `2` At security level 2, the daemon only accepts authentication using the security mechanisms configured by `nisauthconf`(1M) . The default security mechanism is `AUTH_DES` . Security level 2 is the default if the –S option is not used.

EXAMPLES

EXAMPLE 1 Setting up the NIS+ service.

The following example sets up the NIS+ service.

```
example% rpc.nisd
```

EXAMPLE 2 Setting Up NIS+ Service Emulating YP With DNS Forwarding

The following example sets up the NIS+ service, emulating YP with DNS forwarding.

```
example% rpc.nisd -YB
```

ENVIRONMENT VARIABLES

`NETPATH` The transports that the NIS+ service will use can be limited by setting this environment variable (see `netconfig`(4)).

SUMMARY OF TRUSTED SOLARIS CHANGES

A Trusted Solaris 7 system must be the root master of the NIS+ configuration. The `rpc.nisd` daemon must inherit the `net_mac_read` , `net_upgrade_sl` , and `proc_setsl` privileges upon startup. The daemon must be started by a role with a UID of `0` and run with a sensitivity label of `ADMIN_LOW` . ypserver and other NIS (YP) compatibility is not supported.

FILES

`/var/nis/data/parent.object`
This file describes the namespace that is logically above the NIS+ namespace. The most common type of parent object is a DNS object. This object contains contact information for a server of that domain.

`/var/nis/data/root.object`
This file describes the root object of the NIS+ namespace. It is a standard XDR -encoded NIS+ directory object that can be modified by authorized clients using the `nis_modify`(3N) interface.

`/etc/init.d/rpc`
Initialization script for NIS+ .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`nis_cachemgr`(1M) , `rpc.nisd_resolv`(1M) , `rpc.nispasswdd`(1M) , `resolv.conf`(4)

SunOS 5.7 Reference Manual

`nisinit`(1M) , `nissetup`(1M) , `nslookup`(1M) , `syslogd`(1M) , `nis_db`(3N) , `netconfig`(4) , `nisfiles`(4) , `attributes`(5) , `ticots`(7D)

NAME

rpc.nisd_resolv, nisd_resolv – NIS+ service daemon

SYNOPSIS

rpc.nisd_resolv [–v | –V][–F [–C *fd*]] [–t *xx*] [–p *yy*]

DESCRIPTION

`rpc.nisd_resolv` is an auxiliary process which provides DNS forwarding service for NIS hosts requests to both `ypserv` and `rpc.nisd` that are running in the NIS compatibility mode. It is generally started by invoking `rpc.nisd`(1M) with the –B option or `ypserv`(1M) with the –d option. Although it is not recommended, `rpc.nisd_resolv` can also be started independently with the following options.

This command is not supported in the Trusted Solaris environment because `ypserv` and other NIS(YP) compatibility is unsupported.

OPTIONS

–F Run in foreground.

–C Use *fd* for service `xprt` (from `nisd`).

fd

–v Verbose. Send output to the `syslog` daemon.

–V Verbose. Send output to `stdout` .

–t Use transport *xx* .

xx

–p Use transient program# *yy* .

yy

SUMMARY OF TRUSTED SOLARIS CHANGES

This command is not supported in the Trusted Solaris environment.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc.nisd`(1M)

SunOS 5.7 Reference Manual

`nslookup`(1M) , `resolv.conf`(4) , `attributes`(5)

NOTES This command requires that the `/etc/resolv.conf` file be setup for communication with a DNS nameserver. The `nslookup` utility can be used to verify communication with a DNS nameserver. See `resolv.conf`(4) and `nslookup`(1M) .

NAME

rpc.nispasswdd, nispasswdd – NIS+ password update daemon

SYNOPSIS

/usr/sbin/rpc.nispasswdd [–a *attempts*] [–c *minutes*] [–D] [–g] [–v]

DESCRIPTION

`rpc.nispasswdd` daemon is an ONC+ RPC service that services password update requests from `nispasswd`(1) . It updates password entries in the NIS+ `passwd` table.

`rpc.nispasswdd` is normally started from a system startup script after the NIS+ server (`rpc.nisd`(1M)) has been started. `rpc.nispasswdd` will determine whether it is running on a machine that is a master server for one or more NIS+ directories. If it discovers that the host is not a master server, then it will promptly exit. It will also determine if `rpc.nisd`(1M) is running in NIS(YP) compatibility mode (the –Y option) and will register as `yppasswdd` for NIS(YP) clients as well.

ypserv and other NIS (YP) compatibility is not supported.

`rpc.nispasswdd` will syslog all failed password update attempts, which will allow an administrator to determine whether someone was trying to "crack" the passwords.

`rpc.nispasswdd` has to be run by a superuser.

OPTIONS

–a *attempts*	Set the maximum number of attempts allowed to authenticate the caller within a password update request session. Failed attempts are `syslogd`(1M) and the request is cached by the daemon. After the maximum number of allowed attempts the daemon severs the connection to the client. The default value is set to `3` .
–c *minutes*	Set the number of minutes a failed password update request should be cached by the daemon. This is the time during which if the daemon receives further password update requests for the same user and authentication of the caller fails, then the daemon will simply not respond. The default value is set to `30` minutes.
–D	Debug. Run in debugging mode.
–g	Generate DES credential. By default the DES credential is not generated for a user if who does not have one. By specifying this option, if a user does not have a credential, then one will be generated and stored in the NIS+ cred table.
–v	Verbose. With this option, the daemon sends a running narration of what it is doing to the `syslog` daemon. This option is useful for debugging problems.

EXIT STATUS

`0` success

`1` an error has occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

`rpc.nispasswdd` must be run with a UID of `0` and with a sensitivity label of `ADMIN_LOW`. On startup, `rpc.nispasswdd` must inherit the `net_mac_read` and `net_upgrade_sl` privileges. ypserv and other NIS (YP) compatibility is not supported.

FILES

`/etc/init.d/rpc` initialization script for NIS+

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc.nisd`(1M), `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`nispasswd`(1), `passwd`(1), `syslogd`(1M), `attributes`(5)

NAME

nispopulate – Populate the NIS+ tables in a NIS+ domain

SYNOPSIS

/usr/lib/nis/nispopulate –Y [–x] [–f] [–n] [–u] [–v] [–S *0* | *2*][–l <*network_passwd*>] [–d <*NIS+_domain*>] –h <*NIS_server_host*> [–a <*NIS_server_addr*>] –y <*NIS_domain*> [*table*] …

/usr/lib/nis/nispopulate –F [–x] [–f] [–u] [–v] [–S 0 | 2][–d <*NIS+_domain*>] [–l <*network_passwd*>] [–p <*directory_path*>] [*table*] …

/usr/lib/nis/nispopulate –C [–x] [–f] [–v] [–d <*NIS+_domain*>] [–l <*network_passwd*>] [*hosts* | *passwd*]

DESCRIPTION

The `nispopulate` shell script can be used to populate NIS+ tables in a specified domain from their corresponding files or NIS maps. `nispopulate` assumes that the tables have been created either through `nisserver`(1M) or `nissetup`(1M).

The table argument accepts standard names that are used in the administration of Solaris systems and non-standard *key-value* type tables. See `nisaddent`(1M) for more information on *key-value* type tables. If the table argument is not specified, `nispopulate` will automatically populate each of the standard tables. These standard (default) tables are: `auto_master`, `auto_home`, `ethers`, `group`, `hosts`, `networks`, `passwd`, `protocols`, `services`, `rpc`, `netmasks`, `bootparams`, `netgroup`, `aliases` and `shadow`. Note that the `shadow` table is only used when populating from files. The non-standard tables that `nispopulate` accepts are those of *key-value* type. These tables must first be created manually with the `nistbladm`(1) command.

Use the first synopsis (–Y) to populate NIS+ tables from NIS maps. `nispopulate` uses `ypxfr`(1M) to transfer the NIS maps from the NIS servers to the `/var/yp/`<*NIS_domain*> directory on the local machine. Then, it uses these files as the input source. Note that <*NIS_domain*> is case sensitive. Make sure there is enough disk space for that directory.

Use the second synopsis (–F) to populate NIS+ tables from local files. `nispopulate` will use those files that match the table name as input sources in the current working directory or in the specified directory.

Note that when populating the `hosts` and `passwd` tables, `nispopulate` will automatically create the NIS+ credentials for all users and hosts which are defined in the `hosts` and `passwd` tables, respectively. A network passwd is required to create these credentials. This network password is used to encrypt the secret key for the new users and hosts. This password can be specified using the –`l` option or it will use the default password, "`nisplus`". `nispopulate` will not overwrite any existing credential entries in the credential table. Use `nisclient`(1M) to overwrite the entries in the `cred` table. It creates both `LOCAL`

and DES credentials for users, and only DES credentials for hosts. To disable automatic credential creation, specify the "–S *0*" option.

The third synopsis (–C) is used to populate NIS+ credential table with level 2 authentication (DES) from the passwd and hosts tables of the specified domain. The valid table arguments for this operation are passwd and hosts. If this argument is not specified then it will use both passwd and hosts as the input source. If other authentication mechanisms are configured using nisauthconf(1M), the NIS+ credential table will be loaded with credentials for those mechanisms.

If nispopulate was earlier used with "–S *0*" option, then no credentials were added for the hosts or the users. If later the site decides to add credentials for all users and hosts, then this (–C) option can be used to add credentials.

OPTIONS

–a <*NIS_server_addr*>	specifies the IP address for the NIS server. This option is *only* used with the –Y option.
–C	populate the NIS+ credential table from passwd and hosts tables using DES authentication (security level 2). If other authentication mechanisms are configured using nisauthconf(1M), the NIS+ credential table will be populated with credentials for those mechanisms.
–d <*NIS+_domain.*>	specifies the NIS+ domain. The default is the local domain.
–F	populates NIS+ tables from files.
–f	forces the script to populate the NIS+ tables without prompting for confirmation.
–h <*NIS_server_host*>	specifies the NIS server hostname from where the NIS maps are copied from. This is *only* used with the –Y option. This host must be already exist in either the NIS+ hosts table or /etc/hosts file. If the hostname is not defined, the script will prompt you for its IP address, or you can use the –a option to specify the address manually.
–l <*network_passwd*>	specifies the network password for populating the NIS+ credential table. This is *only* used when you are populating the hosts and passwd tables. The default passwd is "nisplus".

–n	does not overwrite local NIS maps in /var/yp/<*NISdomain*> directory if they already exist. The default is to overwrite the existing NIS maps in the local /var/yp/<*NISdomain*> directory. This is *only* used with the –Y option.
–p <*directory_path*>	specifies the directory where the files are stored. This is *only* used with the –F option. The default is the current working directory.
–S 0 | 2	specifies the authentication level for the NIS+ clients. Level 0 is for unauthenticated clients and no credentials will be created for users and hosts in the specified domain. Level 2 is for authenticated (DES) clients and DES credentials will be created for users and hosts in the specified domain. The default is to set up with level 2 authentication (DES). There is no need to run nispopulate with –C for level 0 authentication. Also, if other authentication mechanisms are configured with nisauthconf(1M), credentials for those mechanisms will also be populated for the NIS+ clients.
–u	updates the NIS+ tables (ie., adds, deletes, modifies) from either files or NIS maps. This option should be used to bring an NIS+ table up to date when there are only a small number of changes. The default is to add to the NIS+ tables without deleting any existing entries. Also, see the –n option for updating NIS+ tables from existing maps in the /var/yp directory.
–v	runs the script in verbose mode.
–x	turns the "echo" mode on. The script just prints the commands that it would have executed. Note that the commands are not actually executed. The default is off.
–Y	populate the NIS+ tables from NIS maps.
–y <*NIS_domain*>	specifies the NIS domain to copy the NIS maps from. This is *only* used with the –Y option. The default domainname is the same as the local domainname.

EXAMPLES

EXAMPLE 1 Using `nispopulate`

To populate all the NIS+ standard tables in the domain *xyz.sun.com.* from NIS+ maps of the *yp.sun.COM* domain as input source where host *yp_host* is a YP server of *yp.sun.COM*:

```
nis_server# /usr/lib/nis/nispopulate -Y -y yp.sun.COM \
-h yp_host -d xyz.sun.com.
```

CODE EXAMPLE 1 Updating all NIS+ standard tables

To update all of the NIS+ standard tables from the same NIS domain and hosts shown above:

```
nis_server# /usr/lib/nis/nispopulate -Y -u -y yp.sun.COM -h yp_host \
 -d xyz.sun.com.
```

CODE EXAMPLE 2 Populating the `hosts` table

To populate the `hosts` table in domain *xyz.sun.com.* from the hosts file in the `/var/nis/files` directory and using "`somepasswd`" as the network password for key encryption:

```
nis_server# /usr/lib/nis/nispopulate -F -p \
 /var/nis/files -l somepasswd hosts
```

CODE EXAMPLE 3 Populating the passwd table

To populate the passwd table in domain *xyz.sun.com.* from the passwd file in the `/var/nis/files` directory without automatically creating the NIS+ credentials:

```
nis_server# /usr/lib/nis/nispopulate -F -p /var/nis/files \
 -d xys.sun.com. -S 0 passwd
```

CODE EXAMPLE 4 Populating the credential table

To populate the credential table in domain *xyz.sun.com.* for all users defined in the passwd table.

```
nis_server# /usr/lib/nis/nispopulate -C -d xys.sun.com. passwd
```

CODE EXAMPLE 5 Creating and populating a non-standard key-value type NIS+ table

To create and populate a non-standard key-value type NIS+ table, "`private`", from the file `/var/nis/files/private`: (nispopulate assumes that the private.org_dirkey-value type table has already been created).

```
nis_server# /usr/bin/nistbladm -D access=og=rmcd,nw=r \
-c private key=S,nogw= value=,nogw= private.org.dir


nis_server# /usr/lib/nis/nispopulate -F -p /var/nis/files private
```

ENVIRONMENT VARIABLES

TMPDIR

`nispopulate` normally creates temporary files in the directory `/tmp`. You may specify another directory by setting the environment variable `TMPDIR` to your chosen directory. If `TMPDIR` is not a valid directory, then `nispopulate` will use `/tmp`).

FILES

`/etc/hosts`	Local host name database
`/var/yp`	NIS(YP) domain directory
`/var/nis`	NIS+ domain directory
`/tmp`	Temporary directory.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SUMMARY OF TRUSTED SOLARIS CHANGES

`nissetup` creates the following additional tables: `tsolprof`, `tsoluser`, `tnrhdb`, `tnrhtp`, and `tntime`.

SEE ALSO

Trusted Solaris 7 Reference Manual

`nissetup`(1M)

SunOS 5.7 Reference Manual

`nis+`(1), `nistbladm`(1), `nisaddcred`(1M), `nisaddent`(1M), `nisauthconf`(1M), `nisclient`(1M), `nisserver`(1M), `rpc.nisd`(1M), `ypxfr`(1M), `attributes`(5)

NAME

nissetup – Initialize a NIS+ domain

SYNOPSIS

/usr/lib/nis/nissetup [–Y] [*domain*]

DESCRIPTION

`nissetup` is a shell script that sets up a NIS+ domain to service clients that wish to store system administration information in a domain named *domain*. This domain should already exist prior to executing this command (see `nismkdir`(1) and `nisinit`(1M)).

A NIS+ domain consists of a NIS+ directory and its subdirectories: `org_dir` and `groups_dir`. `org_dir` stores system administration information and `groups_dir` stores information for group access control.

`nissetup` creates the subdirectories `org_dir` and `groups_dir` in *domain*. Both subdirectories will be replicated on the same servers as the parent domain. After the subdirectories are created, `nissetup` creates the default tables that NIS+ serves. These are `auto_master`, `auto_home`, `bootparams`, `cred`, `ethers`, `group`, `hosts`, `mail_aliases`, `netmasks`, `networks`, `passwd`, `protocols`, `rpc`, `services`, `tsolprof`, `tsoluser`, `tnrhdb`, `tnrhtp`, `tntime`, and `timezone`. The `nissetup` script uses the `nistbladm`(1) command to create these tables. The script can be easily customized to add site specific tables that should be created at setup time.

This command is normally executed just once per domain.

OPTIONS

–Y Specify that the domain will be served as both a NIS+ domain as well as an NIS domain using the backward compatibility flag. This will set up the domain to be less secure by making all the system tables readable by unauthenticated clients as well.

SUMMARY OF TRUSTED SOLARIS CHANGES

In Trusted Solaris 2.5, 2.5.1, and 2.7, `nissetup` creates the following additional tables: `tsolprof`, `tsoluser`, `tnrhdb`, `tnrhtp`, and `tntime`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

SunOS 5.7 Reference Manual

`nis+`(1), `nismkdir`(1), `nistbladm`(1), `nisaddent`(1M), `nisinit`(1M) `nisserver`(1M), `attributes`(5)

NOTES

While this command creates the default tables, it does not initialize them with data. This is accomplished with the `nisaddent`(1M) command.

It is easier to use the `nisserver`(1M) script to create subdirectories and the default tables.

NAME

nscd – Name service cache daemon

SYNOPSIS

/usr/sbin/nscd [–f *configuration-file*] [–g] [–e *cachename*, yes | no][–i *cachename*]

DESCRIPTION

`nscd` is a process that provides a cache for the most common name service requests. It is started up during multi-user boot. The default *configuration-file* `/etc/nscd.conf` determines the behavior of the cache daemon. See `nscd.conf`(4).

`nscd` provides cacheing for the `passwd`(4), `group`(4) and `hosts`(4) databases through standard `libc` interfaces, such as `gethostbyname`(3N), `gethostbyaddr`(3N), and others. Each cache has a separate time-to-live for its data; modifying the local database (`/etc/hosts`, and so forth) causes that cache to become invalidated within ten seconds. Note that the shadow file is specifically not cached. `getspnam`(3C) calls remain uncached as a result.

`nscd` also acts as its own administration tool. If an instance of `nscd` is already running, commands are passed to the running version transparently.

In order to preserve NIS+ security, the startup script for `nscd` (`/etc/init.d/nscd`) checks the permissions on the `passwd`, `group` and `host` tables if NIS+ is being used. If those tables are not readable by unauthenticated users, then caching is disabled so that each process continues to authenticate itself as before.

`nscd` runs at the sensitivity label `ADMIN_LOW`. However, it can communicate with DNS name servers at any sensitivity label. It requires the Trusted Path attribute.

OPTIONS

Several of the options described below require a *cachename* specification. Supported values are `passwd`, `group`, and `hosts`.

–f *configuration-file*	Causes `nscd` to read its configuration data from the specified file.
–g	Prints current configuration and statistics to standard output. This is the only option executable by non-root users.
–e *cachename*, `yes`|`no`	Enables or disables the specified cache.
–i *cachename*	Invalidate the specified cache.

EXAMPLES

EXAMPLE 1 Stopping and restarting the `nscd` daemon.

```
example# /etc/init.d/nscd stop
 example# /etc/init.d/nscd start
```

SUMMARY OF TRUSTED SOLARIS CHANGES

To invoke `nscd` requires the Trusted Path attribute, a process sensitivity label of `ADMIN_LOW`, and the following privileges: `net_upgrade_sl`, `net_mac_read`, `proc_setclr`, `sys_trans_label`, `sys_net_config`, `file_dac_write`, and `file_setid`. If `nscd`'s clearance is not `ADMIN_HIGH`, it will be set to `ADMIN_HIGH`.

FILES

`/etc/nscd.conf` determines behavior of cache daemon.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`gethostbyname`(3N), `group`(4), `hosts`(4), `nscd.conf`(4), `passwd`(4), `attributes`(5)

WARNINGS

The `nscd` interface is included in this release on an uncommitted basis only, and is subject to change or removal in a future minor release.

NAME

nslookup – Query name servers interactively

SYNOPSIS

nslookup [– option] … *host* [*server*]

nslookup [– option] … – [*server*]

nslookup

DESCRIPTION

`nslookup` sends queries to Internet domain name servers. It has two modes: interactive and non-interactive. Interactive mode allows the user to contact servers for information about various hosts and domains or to display a list of hosts in a domain. Non-interactive mode is used to display just the name and requested information for a host or domain.

If the name server with which `nslookup` must communicate is on a non-trusted host, `nslookup` can communicate with that host if the host's default sensitivity label matches the `nslookup` process' sensitivity label. To communicate with a name server on a non-trusted host whose default sensitivity label does not match, `nslookup` must be run with the `net_upgrade_sl`, `net_downgrade_sl`, and `net_mac_read` privileges.

OPTIONS

–option Set the permissible options, as shown in the following list. These are the same options that the `set` command supports in interactive mode (see `set` in the `Commands` section for more complete descriptions).

`all`	List the current settings
`class=`*classname*	Restrict search according to the specified class
`d2`	Set exhaustive debug mode on
`nod2`	Set exhaustive debug mode off
`debug`	Set debug mode on
`nodebug`	Set debug mode off
`defname`	Set domain-appending mode on
`nodefname`	Set domain-appending mode off
`domain=`*string*	Establish the appendable domain
`ignoretc`	Set it to ignore packet truncation errors
`noignoretc`	Set it to acknowledge packet truncation errors

OPERANDS

host Inquires about the specified *host*. In this non-interactive command format, `nslookup` does not prompt for additional commands.

– Causes `nslookup` to prompt for more information, such as host names, before sending one or more queries.

server Directs inquiries to the name server specified here in the command line rather than the one read from the `/etc/resolv.conf` file (see `resolv.conf`(4)). *server* can be either a name or an Internet address. If the specified host cannot be reached, `nslookup` resorts to using the name server specified in `/etc/resolv.conf`.

USAGE

Non-interactive Mode

Non-interactive mode is selected when the name or Internet address of the host to be looked up is given as the first argument.

Within non-interactive mode, space-separated options can be specified. They must be entered before the host name, to be queried. Each option must be prefixed with a hyphen.

For example, to request extensive host information and to set the timeout to 10 seconds when inquiring about `gypsy`, enter:

```
example% nslookup–query=hinfo–timeout=10gypsy
```

To avoid repeated entry of an option that you almost always use, place a corresponding `set` command in a `.nslookuprc` file located inside your home directory. (See `Commands` for more information about `set`.) The `.nslookuprc` file can contain several `set` commands if each is followed by a RETURN.

Entering and Leaving Interactive Mode

Interactive mode is selected when

- No arguments are supplied.
- A '–' (hyphen) character is supplied as the *host* argument.

To exit from an interactive `nslookup` session, type Control-d or type the command `exit` followed by RETURN.

Supported Command Interactions

The commands associated with interactive mode are subject to various limitations and run-time conventions.

The maximum length of a command line is 255 characters. When the RETURN key is pressed, command-line execution begins. While a command is running, its execution can be interrupted by typing Control-c.

The first word entered on the command line must be the name of a `nslookup` command unless you wish to enter the name of a host to inquire about. Any unrecognized command is handled as a host name to inquire about. To force a command to be treated as a host name to be inquired about, precede it with a backslash character.

Commands

`exit`
Exit the `nslookup` program.

`help`
`?`
Display a brief summary of commands.

`host` [*server*]
Look up information for *host* using the current default server, or using *server* if it is specified.

If the *host* supplied is an Internet address and the query type is `A` or `1PTR`, the name of the host is returned. If the *host* supplied is a name and it does not have a trailing period, the default domain name is appended to the name. (This behavior depends on the state of the `set` options `–domain`, `–srchlist`, `–defname`, and `–search`).

To look up a host that is not in the current domain, append a period to the name.

`finger` [*name*] [>> *filename*]
Connect with the finger server on the current host, which is defined by the most recent successful host lookup.

If no *name* value is specified, a list of login account names on the current host is generated.

Similar to a shell command interpreter, output can be redirected to a file using the usual redirection symbols: `>` and `>>`.

`ls` [*–options*] *domain* [>> *filename*]
List the information available for *domain*, optionally creating or appending to *filename*. The default output contains host names and their Internet addresses.

Output can be redirected to *filename* using the `>` and `>>` redirection symbols. When output is directed to a file, hash marks are shown for every 50 records received from the server. The permissible values for *options* are:

`a`	Lists aliases of hosts in the domain. This is a synonym for the command `ls–tCNAME`.
`d`	Lists all records for the domain. This is a synonym for the command `ls–tANY`.
`h`	Lists CPU and operating system information for the domain. This is a synonym for the command `ls–tHINFO`.

`s` — Lists well-known services of hosts in the domain. This is a synonym for the command `ls–tWKS`.

`t` *querytype-value* — lists all records of the specified type (see `querytype` within the discussion of the `set` command).

`set` *token=value*
`set` *keyword*

Establish a preferred mode of search operation. Permissible *token* and *keyword* values are:

`all` — Display the current values of frequently-used options. Information about the current default server and host is also displayed.

`cl[ass]=`*classname* — Limit the search according to the protocol group (*classname*) for which lookup information is desired. Permissible *classname* values are:

`ANY`	A wildcard selecting all classes
`IN`	The Internet class (the default)
`CHAOS`	The Chaos class.
`HESIOD`	The MIT Athena Hesiod class.

`d2`
`nod2` — Enable or disable exhaustive debugging mode. Essentially all fields of every packet are displayed. By default, this option is disabled.

`deb[ug]`
`nodeb[ug]` — Enable or disable debugging mode. When debugging mode is enabled, much more information is produced about the packet sent to the server and the resulting answer. By default, this option is disabled.

`def[name]`
`nodef[name]` — Enable or disable appending the default domain name to a single-component lookup request (one that lacks a dot). By default, this option is enabled for `nslookup`. The default

value for the domain name is the value given in `/etc/resolv.conf`, unless: there is an environmental value for `LOCALDOMAIN` when `nslookup` is run; a recent value has been specified through the `srchlist` command or the `set domain` command.

`do[main]=`*string*
: Change the default domain name to be appended to all lookup requests to *string*. For this option to have any effect, the `–defname` option must also be enabled and the `–search` option must be set in a compatible way.

`ignoretc`
`noignoretc`
: The domain search list contains the parents of the default domain if it has at least two components in its name. For example, if the default domain is `CC.Berkeley.EDU`, the search list is `CC.Berkeley.EDU` and `Berkeley.EDU`. Use the `set srchlist` command to specify a different list. Use the `set all` command to display the list.

`srch[list]=`*name1/name2/...*
: Change the default domain name to *name1* and the domain search list to *name1*, *name2*, etc. A maximum of 6 names can be specified, along with slash characters to separate them. For example,

```
example% set srchlist=lcs.MIT.EDU/ai.MIT.EDU/MIT.EDU
```

sets the domain to `lcs.MIT.EDU` and the search list to all three names. This command overrides the default domain name and search list of the `set domain` command. Use the `set all` command to display the list.

`search`
`nosearch`
: Enable or disable having the domain names in the domain search list appended to the request, generating a series of lookup queries if necessary until an answer is received. To take effect, the lookup request must contain at least one dot (period); yet it must not contain a trailing period. By default, this option is enabled.

`po[rt]=`*value*
: Specify the default TCP/UDP name server port. By default, this value is `53`.

`q[uerytype]=`*value*
`ty[pe]=`*value*
: Change the type of information returned from a query to one of:

`A`	The Internet address of the host
`CNAME`	The canonical name for an alias

`HINFO` The host CPU and operating system type

`MD` The mail destination

`MX` The mail exchanger

`MB` The mailbox domain name

`MG` The mail group member

`MINFO` The mailbox or mail list information

`NS` The name server

`PTR` The host name if the query is in the form of an Internet address; otherwise the pointer to other information

`SOA` The domain's start-of-authority information

`TXT` The text information

`UINFO` The user information

`WKS` The supported well-known services

(Other types specified in the *RFC 1035* document are valid, but they are not as useful.)

`recurse`
`norecurse`
Enable or disable having to query other name servers before abandoning a search. By default, this feature is enabled.

`ret[ry]=`*count*
Set the maximum number of times to retry a request before abandoning a search. When a reply to a request is not received within a certain amount of time (changed with `set timeout`), the timeout period is doubled and the request is resent. The retry value controls how many times a request is resent before the request is aborted. The default for *count* is `4`.

`ro[ot]=`*host*
Change the name of the root server to *host*. This affects the `root` command. The default root server is `ns.internet.net`.

`t[timeout]=`*interval*
Change the amount of time to wait for a reply to *interval* seconds. Each retry doubles the timeout period. The default *interval* is `5` seconds.

`vc`
`novc`

Enable or disable the use of a virtual circuit when sending requests to the server. By default, this feature is disabled.

`root`
Change the default server to the server for the root of the domain name space. Currently, the host `ns.internic.net` is used; this command is a synonym for `server ns.internic.net`. The name of the root server can be changed with the `set root` command.

`server` *domain*
`lserver` *domain*
Change the default server to *domain*. `lserver` uses the initial server to look up information about *domain* while `server` uses the current default server. If an authoritative answer can not be found, the names of servers that might have the answer are returned.

`view` *filename*
Sort the output of previous `ls` command(s) and display it one text screenful at a time, similar to `more`(1).

EXAMPLES

EXAMPLE 1 Searching the Internet domain namespace.

To effectively search the Internet domain namespace, it helps to know its structure. At present, the Internet domain name-space is tree-structured, with one top level domain for each country except the U.S.A. There are also some traditional top level domains, not explicitly tied to any particular country. These include:

`COM` Commercial establishments

`EDU` Educational institutions

`ORG` Not-for-profit organizations

`GOV` Government agencies

`MIL` MILNET hosts

If you are looking for a specific host, you need to know something about the host's organization in order to determine the top-level domain that it belongs to. For instance, if you want to find the Internet address of a machine at UCLA, do the following:

- Connect with the root server using the `root` command. The root server of the name space has knowledge of the top-level domains.
- Since UCLA is a university, its domain name is `ucla.edu`. Connect with a server for the `ucla.edu` domain with the command **`server ucla.edu`**. The response produces the names of hosts that act as servers for that

domain. Note: the root server does not have information about `ucla.edu`, but knows the names and addresses of hosts that do. Once located by the root server, all future queries will be sent to the UCLA name server.

- To request information about a particular host in the domain (for instance, `locus`), just type the host name. To request a listing of hosts in the UCLA domain, use the `ls` command. The `ls` command requires a domain name (in this case, `ucla.edu`) as an argument.

If you are connected with a name server that handles more than one domain, all lookups for host names must be fully specified with its domain. For instance, the domain `harvard.edu` is served by `seismo.css.gov`, which also services the `css.gov` and `cornell.edu` domains. A lookup request for the host `aiken` in the `harvard.edu` domain must be specified as `aiken.harvard.edu`. However, the `set domain=`*name* and `set defname` commands can be used to automatically append a domain name to each request.

After a successful lookup of a host, use the `finger`(1) command to see who is on the system, or to finger a specific person. (`finger` requires the type to be `A`.)

To get other information about the host, use the `set querytype=`*value* command to change the type of information desired and request another lookup.

ENVIRONMENT VARIABLES

`HOSTALIASES` References the file containing host aliases

`LOCALDOMAIN` Overrides default domain

EXIT STATUS

The process returns the following values:

`0` On success.

`1` On failure.

SUMMARY OF TRUSTED SOLARIS CHANGES

If the name server with which `nslookup` must communicate is on a non-trusted host, `nslookup` can communicate with that host if the host's default sensitivity label matches the `nslookup` process' sensitivity label. To communicate with a name server on a non-trusted host whose default sensitivity label does not match, `nslookup` must be run with the `net_upgrade_sl`, `net_downgrade_sl`, and `net_mac_read` privileges.

FILES

`/etc/resolv.conf`	initial domain name and name server addresses
`$HOME/.nslookuprc`	initial option commands
`/usr/lib/nslookup.help`	summary of commands

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`nstest`(1M), `resolver`(3N)

SunOS 5.7 Reference Manual

`resolv.conf`(4), `attributes`(5)

RFC 882, RFC 883

DIAGNOSTICS

If the lookup request is successful, an error message is produced. Possible errors are:

`Timed out`
The server did not respond to a request after a certain amount of time (changed with `set timeout`=*value*) and a certain number of retries (changed with `set retry`=*value*).

`No response from server`
No name server is running on the server machine.

`No records`
The server does not have resource records of the current query type for the host, although the host name is valid. The query type is specified with the `set querytype` command.

`Non-existent domain`
The host or domain name does not exist.

`Connection refused`
`Network is unreachable`
The connection to the name or finger server can not be made at the current time. This error commonly occurs with `ls` and `finger` requests.

`Server failure`
The name server found an internal inconsistency in its database and could not return a valid answer.

`Refused`
The name server refused to service the request.

`Format error`
The name server found that the request packet was not in the proper format. This may indicate an error in `nslookup`.

NAME

nstest – DNS test shell

SYNOPSIS

nstest [–d] [–i] [–r] [–v] [–p *port*] [*inet_addr* [*logfile*]]

DESCRIPTION

`nstest` is an interactive DNS test program. Queries are formed and sent by user command; any reply received is printed on the standard output. *inet_addr* is the Internet address of the DNS resolver to which `nstest` should send its queries. If *inet_addr* is not included, `nstest` first tries to contact a DNS server on the local host; if that fails, it tries the servers listed in the `/etc/resolv.conf` file. If a *logfile* is supplied, `nstest` uses it to log the queries sent and replies received.

OPTIONS

–d Causes `nstest` to create a file named `ns_packet.dump` (if it does not exist) and write into it a raw (binary) copy of each packet sent. If `ns_packet.dump` does exist, `nstest` will truncate it.

–i Sets the `RES_IGNTC` flag on the queries it makes. See `resolver`(3N) for a description of the `RES_IGNTC` flag.

–r Turns off the `RES_RECURSE` flag on the queries it makes. See `resolver`(3N) for a description of the `RES_RECURSE` flag.

–v Turns on the `RES_USEVC` and `RES_STAYOPEN` flags on the `res_send( )` calls made. See `resolver`(3N) for a description of the `RES_USEVC` and `RES_STAYOPEN` flags.

–p Causes `nstest` to use the supplied *port* instead of the default name server port.

USAGE

When `nstest` starts, it prints a prompt (">") and waits for user input. DNS queries are formed by typing a *key letter* followed by the appropriate *argument*. Each *key letter* results in a call to `res_mkquery( )` with *op* set to either `IQUERY` or `QUERY` and *type* set to one of the type values (defined in `<arpa/nameser.h>`). (Any other `key letter` than those listed below causes `nstest` to print a summary of the following table.)

Key Letter & Argument	Op	Type
a*host*	QUERY	T_A
A*addr*	IQUERY	T_A
B*user*	QUERY	T_MG
b*user*	QUERY	T_MB
c*host*	QUERY	T_CNAME

f*host*	QUERY	T_UINFO
G*gid*	IQUERY	T_GID
g*host*	QUERY	T_GID
h*host*	QUERY	T_HINFO
i*host*	QUERY	T_MINFO
M*host*	QUERY	T_MAILB
m*host*	QUERY	T_MX
n*host*	QUERY	T_NS
p*host*	QUERY	T_PTR
r*host*	QUERY	T_MR
s*host*	QUERY	T_SOA
T*host*	QUERY	T_TXT
U*uid*	IQUERY	T_UID
u*host*	QUERY	T_UID
w*host*	QUERY	T_WKS
x*host*	QUERY	T_AXFR

After the query is successfully formed, `res_send( )` is called to send it and wait for a reply. `nstest` then prints the following on the standard output:

- a summary of the request and reply packets, including the HEADER structure (defined in <arpa/nameser.h>) used in the request
- the question being asked of the name server
- an enumeration of the name server(s) being polled
- a summary of the HEADER structure received in the reply
- the question the name server answered
- the answer itself

EXAMPLES

EXAMPLE 1 Fetching the address of host `playground.sun.com` from the Sun name server.

To fetch the address of host `playground.sun.com` from the Sun name server, the user would enter:

```
$ nstest 192.9.5.1
> aplayground.sun.com
```

The utility `nstest` would return the following:

```
res_mkquery(0, playground.sun.com, 1, 1)
res_send()
HEADER:
        opcode = QUERY, id = 1, rcode = NOERROR
        header flags:  rd
        qdcount = 1, ancount = 0, nscount = 0, arcount = 0

QUESTIONS:
        playground.sun.com, type = A, class = IN

Querying server (# 1) address = 192.9.5.1
got answer:
HEADER:
        opcode = QUERY, id = 1, rcode = NOERROR
        header flags:  qr aa rd ra
        qdcount = 1, ancount = 1, nscount = 0, arcount = 0

QUESTIONS:
        playground.sun.com, type = A, class = IN
ANSWERS:
        playground.sun.com
        type = A, class = IN, ttl = 1 day, dlen = 4
        internet address = 192.9.5.5
```

EXAMPLE 2 Looking up a PTR record.

To look up a PTR record, enter:

```
$ nstest 192.9.5.1
> p5.5.9.192.in-addr.arpa
```

The utility `nstest` would return the following:

```
res_mkquery(0, 5.5.9.192.in-addr.arpa, 1, 12)
res_send()
HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  rd
        qdcount = 1, ancount = 0, nscount = 0, arcount = 0

QUESTIONS:
        5.5.9.192.in-addr.arpa, type = PTR, class = IN

Querying server (# 1) address = 192.9.5.1
got answer:
HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  qr aa rd ra
        qdcount = 1, ancount = 1, nscount = 0, arcount = 0

QUESTIONS:
        5.5.9.192.in-addr.arpa, type = PTR, class = IN

ANSWERS:
        5.5.9.192.in-addr.arpa
        type = PTR, class = IN, ttl = 7 hours 47 mins 2 secs, dlen = 23
        domain name = playground.sun.com
```

SUMMARY OF TRUSTED SOLARIS CHANGES

`nstest` uses `resolver`(3N) instead of `resolver`(3N).

FILES

`</usr/include/arpa/nameser.h>`	include file for implementation of DNS protocol
`</usr/include/resolv.h>`	include file for the resolver daemon (in.named)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`resolver`(3N)

SunOS 5.7 Reference Manual

`nslookup`(1M), `attributes`(5)

NAME

pbind – Control and query bindings of processes to processors

SYNOPSIS

pbind –b *processor_id pid…*

pbind –u *pid…*

pbind [–q] [*pid…*]

DESCRIPTION

`pbind` controls and queries bindings of processes to processors. `pbind` binds all the LWPs (lightweight processes) of a process to a processor, or removes or displays the bindings.

When an LWP is bound to a processor, it will be executed only by that processor except when the LWP requires a resource that is provided only by another processor. The binding is not exclusive, that is, the processor is free execute other LWPs as well.

Bindings are inherited, so new LWPs and processes created by a bound LWP will have the same binding. Binding an interactive shell to a processor, for example, binds all commands executed by the shell.

This command may be used to bind or unbind any process for which the user has permission to signal — any process that has the same effective UID as the user.

OPTIONS

The following options are supported:

–b *processor_id* Binds all the LWPs of the specified processes to the processor *processor_id*. Specify *processor_id* as the processor ID of the processor to be controlled or queried. *processor_id* must be present and on-line. Use the `psrinfo` command to determine whether or not *processor_id* is present and online. See `psrinfo`(1M).

–q Displays the bindings of the specified processes, or of all processes. If a process is composed of multiple LWPs, which have different bindings, the bindings of only one of the bound LWPs will be displayed.

–u Removes the bindings of all LWPs of the specified processes, allowing them to be executed on any on-line processor.

OPERANDS

The following operands are supported:

pid The process ID of the process to be controlled or queried.

EXAMPLES

EXAMPLE 1 Binding processes

The following example binds processes 204 and 223 to processor 2.

```
example% pbind –b 2 204 223
```

This command displays the following output:

```
process id 204: was 2, now 2
process id 223: was 3, now 2
```

CODE EXAMPLE 1 Unbinding a process

The following example unbinds process 204.

```
example% pbind –u 204
```

CODE EXAMPLE 2 Querying Bindings

The following example demonstrates that process 1 is bound to processor 0, process 149 has at least one LWP bound to CPU3, and process 101 has no bound LWPs.

```
example% pbind –q 1 149 101
```

This command displays the following output:

```
process id 1: 0
process id 149: 3
process id 101: not bound
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `proc_owner` privilege is needed to bind or unbind any process with an effective user ID different from that of the user.

SEE ALSO

Trusted Solaris 7 Reference Manual

`psradm`(1M)

SunOS 5.7 Reference Manual

`psrinfo`(1M), `psrset`(1M), `processor_bind`(2), `processor_info`(2), `sysconf`(3C), `attributes`(5)

DIAGNOSTICS

`pbind: cannot query pid 31: No such process`
The process specified did not exist or has exited.

`pbind: cannot bind pid 31: Not owner`
The user does not have permission to bind the process.

`pbind: cannot bind pid 31: Invalid argument`

The specified processor is not online.

NAME

pfsh, clist – Profile shell

SYNOPSIS

pfsh [–acefhiknprstuvx] [*argument*...]

DESCRIPTION

The profile shell is a modified version of the Bourne shell, sh(1) . Based on the user's profiles, pfsh restricts the commands that can be executed. Based on the profile definitions, pfsh determines which privileges, user ID (UID), and group ID (GID) to use in executing commands.

Usage

Refer to the sh(1) man page for a complete usage description. pfsh adds the clist command.

Commands

clist	Displays a list of the commands that are permitted for the user.
[–-hpniu]	
–h	Includes a hexadecimal list of the privileges assigned to each command in the command list.
–p	Includes a list of the privileges assigned to each command in the command list. The list is in text form.
–n	Includes a comma-separated decimal list of the privileges assigned to each command in the command list.
–i	Includes the UID and GID assigned to each command in the command list.
–u	Lists only those commands that are are unusable because the profile assigned privileges that pfsh did not inherit. (See WARNINGS .)

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

tsolprof(4) , tsoluser(4)

SunOS 5.7 Reference Manual

sh(1) , attributes(5)

WARNINGS

`pfsh` must inherit privileges in order to run commands with those privileges. Privileges for a command that are defined in a profile may not be inherited when `pfsh` runs that command. If such a command is executed, a warning message is printed and the command is run with no privileges.

Profiles are searched in the order specified in the user's `tsoluser` entry. If the same command appears in more than one profile, `pfsh` uses the first entry whose label range includes the sensitivity label of the process.

When it is executed, `pfsh` builds the list of allowable commands by reading the user's profiles. If any changes are made to the profiles while `pfsh` is running, the changes will not take effect until the shell is restarted.

NOTES

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

halt, poweroff – Stop the processor

SYNOPSIS

/usr/sbin/halt [–lnqy]

/usr/sbin/poweroff [–lnqy]

DESCRIPTION

`halt` and `poweroff` write out any pending information to the disks and then stop the processor. `poweroff` will have the machine remove power, if possible.

`halt` and `poweroff` normally log the system shutdown to the system log daemon, `syslogd`(1M) , and place a shutdown record in the login accounting file `/var/adm/wtmp` . These actions are inhibited if the –n or –q options are present.

OPTIONS

The following options are supported:

–l Suppress sending a message about who executed `halt` to the system log daemon, `syslogd`(1M) .

–n Prevent the `sync`(4) before stopping.

–q Quick halt. No graceful shutdown is attempted.

–y Halt the system, even from a dialup terminal.

FILES

`/var/adm/wtmp` login accounting file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This command requires the `PRIV_SYS_BOOT` privilege and an effective uid of `0` in order to run.

SEE ALSO

Trusted Solaris 7 Reference Manual

`init`(1M) , `reboot`(1M)

SunOS 5.7 Reference Manual

`shutdown`(1M) , `sync`(1M) , `syslogd`(1M) , `attributes`(5)

NOTES

Unlike `shutdown`(1M) and `init`(1M) , `halt` does not execute the `rc0` scripts.

`poweroff` is equivalent to `init 5` .

NAME

praudit – Print contents of an audit trail file

SYNOPSIS

praudit [–lrs] [–d*del*] [*filename*…]

DESCRIPTION

praudit reads the listed *filename*s (or standard input, if no *filename* is specified) and interprets the data as audit trail records as defined in audit.log(4). By default, times, user and group IDs (UIDs and GIDs, respectively) are converted to their ASCII representation. Record type and event fields are converted to their ASCII representation. A maximum of 100 audit files can be specified on the command line.

The PAF_LABEL_VIEW process attribute flag for the current process will affect how ADMIN_HIGH or ADMIN_LOW binary labels are translated to their text equivalents. See pattr(1) and getpattr(2) for more information.

OPTIONS

–l Prints one line per record. The record type and event fields are always converted to their short text representation as is done for the –s option.

–r Print records in their raw form. Times, UIDs, GIDs, record types, and events are displayed as integers. This option and the –s option are exclusive. If both are used, a format usage error message is output.

–s Print records in their short form. All numeric fields are converted to text and displayed. The short text representations for the record type and event fields are used. This option and the –r option are exclusive. If both are used, a format usage error message is output.

–d*del* Use *del* as the field delimiter instead of the default delimiter, which is the comma. If *del* has special meaning for the shell, it must be quoted. The maximum size of a delimiter is four characters.

FILES

/etc/security/audit_event Audit event definition and class mappings.

/etc/security/audit_class Audit class definitions.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This function uses the PAF_LABEL_VIEW process attribute flag and converts security labels as well as times and IDs.

The functionality described in this man page is available only if auditing is enabled. By default, auditing is enabled in the Trusted Solaris environment.

SEE ALSO

Trusted Solaris 7 Reference Manual

`audit`(2), `getauditflags`(3), `audit.log`(4), `audit_class`(4), `audit_event`(4)

Trusted Solaris Audit Administration

SunOS 5.7 Reference Manual

`group`(4), `passwd`(4), `attributes`(5)

NAME

prtconf – Print system configuration

SYNOPSIS

SPARC **/usr/sbin/prtconf** [–V] | [–F] | [–x] | [–vpPD]

x86 **/usr/sbin/prtconf** [–V] | [–x] | [–vpPD]

DESCRIPTION

The `prtconf` command prints the system configuration information. The output includes the total amount of memory, and the configuration of system peripherals formatted as a device tree.

OPTIONS

The following options are supported:

–D For each system peripheral in the device tree, displays the name of the device driver used to manage the peripheral.

–F (SPARC only). Returns the device pathname of the console frame buffer, if one exists. If there is no frame buffer, `prtconf` returns a non-zero exit code. This flag must be used by itself. It returns only the name of the console, frame buffer device or a non-zero exit code. For example, if the console frame buffer on a SPARCstation 1 is `cgthree` in SBus slot #3, the command returns: `/sbus@1,f80000000/cgthree@3,0`. This option could be used to create a symlink for `/dev/fb` to the actual console device.

–p Displays information derived from the device tree provided by the firmware (PROM) on SPARC platforms or the booting system on x86 platforms.

–P Includes information about pseudo devices. By default, information regarding pseudo devices is omitted.

–v Specifies verbose mode.

–V Displays platform-dependent PROM (on SPARC platforms) or booting system (on x86 platforms) version information. This flag must be used by itself. The output is a string. The format of the string is arbitrary and platform-dependent.

–x Reports if the firmware on this system is 64-bit ready. Some existing platforms may need a firmware upgrade in order to run the 64-bit kernel. If the operation is not applicable to this platform or the firmware is already 64-bit ready, it exits silently with a return code of zero. If the operation is applicable to this platform and the firmware is not 64-bit ready, it displays a descriptive message on stdout and exits with a non-zero return code. The hardware platform documentation contains more information about the platforms that may need a firmware upgrade in order to run the 64-bit kernel.

This flag overrides all other flags and must be used by itself.

EXAMPLES

EXAMPLE 1 Running `prtconf` on a SPARC Sun4/65 Series Machine

Running `prtconf` on a Sun4/65 series machine produces the following sample output:

```
example% prtconf
System Configuration:  Sun Microsystems  sun4c
Memory size: 16 Megabytes
System Peripherals (Software Nodes):
Sun 4_65
    options, instance #0
    zs, instance #0
    zs, instance #1
    fd (driver not attached)
    audio (driver not attached)
    sbus, instance #0
        dma, instance #0
        esp, instance #0
            sd (driver not attached)
            st (driver not attached)
            sd, instance #0
            sd, instance #1 (driver not attached)
            sd, instance #2 (driver not attached)
            sd, instance #3
            sd, instance #4 (driver not attached)
            sd, instance #5 (driver not attached)
            sd, instance #6 (driver not attached)
        le, instance #0
        cgsix (driver not attached)
    auxiliary-io (driver not attached)
    interrupt-enable (driver not attached)
    memory-error (driver not attached)
    counter-timer (driver not attached)
    eeprom (driver not attached)
    pseudo, instance #0
```

EXAMPLE 2 Running `prtconf` on an x86 Machine

Running `prtconf` on an x86 machine produces the following sample output:

```
example% prtconf
System Configuration:  Sun Microsystems  i86pc
Memory size: 32 Megabytes
System Peripherals (Software Nodes):

i86pc
    eisa, instance #0
        kd, instance #0
        ata, instance #0
            cmdk, instance #0
        aha, instance #0
            cmdk, instance #1 (driver not attached)
            cmdk, instance #2 (driver not attached)
            cmdk, instance #3 (driver not attached)
```

```
            cmdk, instance #4 (driver not attached)
            cmdk, instance #5 (driver not attached)
            cmdk, instance #6 (driver not attached)
            cmdk, instance #7
        chanmux, instance #0
        asy, instance #0
        asy, instance #1
        elx, instance #0
        elx, instance #1 (driver not attached)
        elx, instance #2 (driver not attached)
        elx, instance #3 (driver not attached)
        fdc, instance #0
            fd, instance #0
            fd, instance #1
    options, instance #0
    objmgr, instance #0
    pseudo, instance #0
example%
```

EXIT STATUS

The following exit values are returned:

`0` No error occurred.

`non-zero` With the –`F` option (SPARC only), a non-zero return value means that the output device is not a framebuffer. With the –`X` option, a non-zero return value means that the firmware is not 64–bit ready. In all other cases, a non-zero return value means that an error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –`v` option of this command can be run from an administrative role.

The `file_mac_read` privilege is required in order to run the `prtconf` command.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit) SUNWesxu (64-bit)

SEE ALSO

`modinfo`(1M), `sysdef`(1M), `attributes`(5)

Sun Hardware Platform Guide

SPARC Only

`openprom`(7D)

NOTES

The output of the `prtconf` command is highly dependent on the version of the PROM installed in the system. The output will be affected in potentially all circumstances.

The `driver not attached` message means that no driver is currently attached to that instance of the device. In general, drivers are loaded and installed (and attached to hardware instances) on demand, and when needed, and may be uninstalled and unloaded when the device is not in use.

NAME

psradm – Change processor operational status

SYNOPSIS

psradm –f | –i | –n [–v] *processor_id*…

psradm –a –f | –i | –n [–v]

DESCRIPTION

The psradm utility changes the operational status of processors. The legal states for the processor are on-line, off-line, and no-intr.

An on-line processor processes LWPs (lightweight processes) and may be interrupted by I/O devices in the system.

An off-line processor does not process any LWPs. Usually, an off-line processor is not interruptible by I/O devices in the system. On some processors or under certain conditions, it may not be possible to disable interrupts for an off-line processor. Thus, the actual effect of being off-line may vary from machine to machine.

A no-intr processor processes LWPs but is not interruptible by I/O devices.

A processor may not be taken off-line if there are LWPs that are bound to the processor. On some architectures, it might not be possible to take certain processors off-line if, for example, the system depends on some resource provided by the processor.

At least one processor in the system must be able to process LWPs. At least one processor must also be able to be interrupted. Since an off-line processor may be interruptible, it is possible to have an operational system with one processor no-intr and all other processors off-line but with one or more accepting interrupts.

If any of the specified processors are powered off, psradm may power on one or more processors.

To succeed, this command needs the sys_config privilege.

OPTIONS

The following options are supported:

–a Perform the action on all processors, or as many as possible.

–f Take the specified processors off-line.

–i Set the specified processors no-intr.

–n Bring the specified processors on-line.

–v Output a message giving the results of each attempted operation.

OPERANDS

The following operands are supported:

processor_id The processor ID of the processor to be set on-line or off-line.

EXAMPLES

EXAMPLE 1 Set processors 2 and 3 off-line.

The following example sets processors 2 and 3 off-line.

```
psradm –f 2 3
```

EXAMPLE 2 Set processors 1 and 2 no-intr.

The following example sets processors 1 and 2 no-intr.

```
psradm –i 1 2
```

EXAMPLE 3 Set all processors on-line.

The following example sets all processors on-line.

```
psradm –a –n
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/wtmp` records logging processor status changes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the `sys_config` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`p_online`(2)

SunOS 5.7 Reference Manual

`psrinfo`(1M), `psrset`(1M), `attributes`(5)

DIAGNOSTICS

`psradm: processor 4: Invalid argument`
The specified processor does not exist in the configuration.

`psradm: processor 3: Device busy`
The specified processor could not be taken off-line because it either has LWPs bound to it, is the last on-line processor in the system, or is needed by the system because it provides some essential service.

`psradm: processor 3: Device busy`
The specified processor could not be set no-intr because it is the last interruptible processor in the system, or or it is the only processor in the system that can service interrupts needed by the system.

`psradm: processor 3: Device busy`
The specified processor is powered off, and it cannot be powered on because some platform-specific resource is unavailable.

`psradm: processor 0: Not owner`
The user does not have permission to change processor status.

`psradm: processor 2: Operation not supported`
The specified processor is powered off, and the platform does not support power on of individual processors.

NAME

in.rarpd, rarpd – DARPA Reverse Address Resolution Protocol server

SYNOPSIS

/usr/sbin/in.rarpd [–d] –a

/usr/sbin/in.rarpd [–d] *device unit*

DESCRIPTION

`in.rarpd` starts a daemon that responds to Reverse Address Resolution Protocol (RARP) requests. The daemon forks a copy of itself that runs in background. It must be started from the trusted path, with a UID of 0 and the label `ADMIN_LOW`. To succeed, it must inherit the `sys_net_conf` and `net_broadcast` privileges.

RARP is used by machines at boot time to discover their Internet Protocol (IP) address. The booting machine provides its Ethernet address in a RARP request message. Using the `ethers` and `hosts` databases, `in.rarpd` maps this Ethernet address into the corresponding IP address which it returns to the booting machine in an RARP reply message. The booting machine must be listed in both databases for `in.rarpd` to locate its IP address. `in.rarpd` issues no reply when it fails to locate an IP address.

`in.rarpd` uses the STREAMS-based Data Link Provider Interface (DLPI) message set to communicate directly with the datalink device driver.

OPTIONS

The following options are supported:

–a Get the list of available network interfaces from IP using the SIOCGIFADDR ioctl and start a RARP daemon process on each interface returned.

–d Print assorted debugging messages while executing.

EXAMPLES

EXAMPLE 1 Starting an `in.rarpd` Daemon for Each Network Interface Name Returned From `/dev/ip`:

The following command starts an `in.rarpd` for each network interface name returned from `/dev/ip`:

```
example# /usr/sbin/in.rarpd –a
```

EXAMPLE 2 Starting an `in.rarpd` Daemon on the Device `/dev/le` with the Device Instance Number `0`

The following command starts one `in.rarpd` on the device `/dev/le` with the device instance number `0`.

```
example# /usr/sbin/in.rarpd le 0
```

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.rarpd` should be started from the trusted path with a UID `0` and sensitivity label of `ADMIN_LOW`. It must inherit the `sys_net_config` and `net_broadcast` privileges.

FILES

`/etc/ethers`	File or NIS+ map of host names and ethernet addresses.
`/etc/hosts`	File or NIS+ map of Internet host names and addresses.
`/tftpboot`	Directory for remote boot scripts.
`/dev/ip`	List of available network interfaces.
`/dev/arp`	Address resolution protocol list.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`ifconfig`(1M)

SunOS 5.7 Reference Manual

`boot`(1M) , `ethers`(4) , `hosts`(4) , `netconfig`(4) , `attributes`(5) , `dlpi`(7P)

RFC-903, *A Reverse Address Resolution Protocol* , Network Information Center, SRI International.

Unix International, *Data Link Provider Interface* , Version 2, May 7, 1991, Sun Microsystems, 800-6915-01.

NAME

rdate – Set system date from a remote host

SYNOPSIS

rdate *hostname*

DESCRIPTION

`rdate` sets the local date and time from the *hostname* given as an argument. This program needs to inherit the `sys_config` privilege to run properly.Typically `rdate` can be inserted as part of a startup script.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

This program needs to inherit the `sys_config` privilege to run properly.

SEE ALSO

Trusted Solaris 7 Reference Manual

There are no Trusted Solaris-specific references for this manual page.

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

in.rdisc, rdisc – Network router discovery daemon

SYNOPSIS

/usr/sbin/in.rdisc [–a] [–f] [–s] [*send-address*] [*receive-address*]

/usr/sbin/in.rdisc –r [–p *preference*] [–T *interval*] [*send-address*] [*receive-address*]

DESCRIPTION

`in.rdisc` implements the ICMP router discovery protocol. The first form of the command is used on hosts and the second form is used on routers. On a host, `in.rdisc` is invoked at boot time to populate the network routing tables with default routes. On a router, it is also invoked at boot time in order to start advertising the router to all the hosts.

Host (First Form)

On a host, `in.rdisc` listens on the `ALL_HOSTS` (224.0.0.1) multicast address for `ROUTER_ADVERTISE` messages from routers. The received messages are handled by first ignoring those listed router addresses with which the host does not share a network. Among the remaining addresses, the ones with the highest preference are selected as default routers and a default route is entered in the kernel routing table for each one of them.

Optionally, `in.rdisc` can avoid waiting for routers to announce themselves by sending out a few `ROUTER_SOLICITATION` messages to the `ALL_ROUTERS` (224.0.0.2) multicast address when it is started.

A timer is associated with each router address. The address will no longer be considered for inclusion in the routing tables if the timer expires before a new *advertise* message is received from the router. The address will also be excluded from consideration if the host receives an *advertise* message with the preference being maximally negative.

Router (Second Form)

When `in.rdisc` is started on a router, it uses the `SIOCGIFCONF` `ioctl`(2) to find the interfaces configured into the system and it starts listening on the `ALL_ROUTERS` multicast address on all the interfaces that support multicast. It sends out `advertise` messages to the `ALL_HOSTS` multicast address advertising all its IP addresses. A few initial *advertise* messages are sent out during the first 30 seconds and after that it will transmit *advertise* messages approximately every 600 seconds.

When `in.rdisc` receives a *solicitation* message, it sends an *advertise* message to the host that sent the *solicitation* message.

When `in.rdisc` is terminated by a signal, it sends out an *advertise* message with the preference being maximally negative.

OPTIONS

–a Accept all routers independent of the preference they have in their *advertise* messages. Normally, `in.rdisc` only accepts (and enters in the kernel routing tables) the router or routers with the highest preference.

–f	Run in.rdisc forever even if no routers are found. Normally, in.rdisc gives up if it has not received any *advertise* message after soliciting three times, in which case it exits with a non-zero exit code. If –f is not specified in the first form then –s must be specified.
–r	Act as a router, rather than a host.
–s	Send three *solicitation* messages initially to quickly discover the routers when the system is booted. When –s is specified, in.rdisc exits with a non-zero exit code if it can not find any routers. This can be overridden with the –f option.
–p *preference*	Set the preference transmitted in the *solicitation* messages. The default is zero .
–T *interval*	Set the interval between transmitting the *advertise* messages. The default time is 600 seconds.

SUMMARY OF TRUSTED SOLARIS CHANGES

in.rdisc must be started from the trusted path. To access /dev/rawip , in.rdisc must be started with an effective UID of 0 , or it must have the file_dac_read and file_dac_write privileges. To modify kernel routing tables, it must inherit the sys_net_config privilege; to open a raw socket, it needs the net_rawaccess privilege; and to send multicast or broadcast packets, it needs the net_broadcast privilege.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

in.routed(1M)

SunOS 5.7 Reference Manual

ioctl(2) , attributes(5) , icmp(7P) , inet(7P)

Deering, S.E., editor, *ICMP Router Discovery Messages* , RFC 1256, Network Information Center, SRI International, Menlo Park, California, September 1991.

NAME

reboot – Restart the operating system

SYNOPSIS

/usr/sbin/reboot [–dlnq] [*boot arguments*]

DESCRIPTION

`reboot` restarts the kernel. The kernel is loaded into memory by the PROM monitor, which transfers control to the loaded kernel.

Although `reboot` can be run by an administrative role with appropriate privilege and authorization at any time, `shutdown`(1M) is normally used first to warn all users logged in of the impending loss of service. See `shutdown`(1M) for details.

`reboot` performs a `sync`(1M) operation on the disks, and then a multi-user reboot is initiated. See `init`(1M) for details.

`reboot` normally logs the reboot to the system log daemon, `syslogd`(1M), and places a shutdown record in the login accounting file `/var/adm/wtmp`. These actions are inhibited if the –`n` or –`q` options are present.

Normally, the system will reboot itself at power-up or after crashes.

OPTIONS

–`d`	Dump system core before rebooting. This option is provided for compatibility, but is not supported by the underlying `reboot`(3C) call.
–`l`	Suppress sending a message to the system log daemon, `syslogd`(1M) about who executed `reboot`.
–`n`	Avoid the `sync`(1M) operation. Use of this option can cause file system damage.
–`q`	Quick. Reboot quickly and ungracefully, without shutting down running processes first.
`boot` *arguments*	These arguments are accepted for compatibility, and are passed unchanged to the `uadmin`(2) system call.

EXAMPLES

EXAMPLE 1 Example of the reboot command.

In the example below, the delimiter '—' (two hyphens) must be used to separate the options of `reboot` from the arguments of `boot`(1M).

```
example# reboot –dl — –rv
```

FILES

`/var/adm/wtmp` Login accounting file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`reboot` requires the `sys_boot` privilege in order to run. Shutting down a computer requires authorization.

SEE ALSO

Trusted Solaris 7 Reference Manual

`halt`(1M), `init`(1M), `uadmin`(2)

SunOS 5.7 Reference Manual

`boot`(1M), `crash`(1M), `fsck`(1M), `shutdown`(1M), `sync`(1M), `syslogd`(1M), `reboot`(3C), `attributes`(5)

NAME

accept, reject – Accept or reject print requests

SYNOPSIS

accept *destination…*

reject [–r *reason*] *destination…*

DESCRIPTION

`accept` allows the queueing of print requests for the named destinations.

`reject` prevents queueing of print requests for the named destinations.

Use `lpstat` **–a** to check if destinations are accepting or rejecting print requests.

`accept` and `request` must be run on the print server; they have no meaning on a client system.

OPTIONS

The following options are supported for `reject` .

–r *reason* — Assigns a reason for rejection of print requests for *destination* . Enclose *reason* in quotes if it contains blanks. *reason* is reported by `lpstat` **–a** . By default, *reason* is `unknown reason` for existing destinations, and `new destination` for destinations added to the system but not yet accepting requests.

OPERANDS

The following operands are supported.

destination — The name of the destination accepting or rejecting print requests. Destination specifies the name of a printer or class of printers [see `lpadmin`(1M)]. Specify *destination* using atomic name. See `printers.conf`(4) for information regarding the naming conventions for atomic names.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/var/spool/lp/*` — LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
CSI	Enabled (see `NOTES`)

SEE ALSO

Trusted Solaris 7 Reference Manual

`enable`(1) , `lp`(1) , `lpstat`(1) , `lpadmin`(1M) , `lpsched`(1M)

SunOS 5.7 Reference Manual

`printers.conf` (4), `attributes`(5)

NOTES

`accept` and `reject` only affect queuing on the print server's spooling system. Requests made from a client system remain queued in the client system's queuing mechanism until they are cancelled or accepted by the print server's spooling system.

`accept` is CSI -enabled except for the *destination* name.

NAME

rem_drv – Remove a device driver from the system

SYNOPSIS

rem_drv [–b *basedir*] *device_driver*

DESCRIPTION

The `rem_drv` command informs the system that the device driver *device_driver* is no longer valid. If possible, `rem_drv` unloads *device_driver* from memory. Entries for the device in the `/devices` namespace are removed. `rem_drv` also updates the system driver configuration files.

If `rem_drv` has been executed, the next time the system is rebooted it will automatically perform a reconfiguration boot (see `kernel`(1M)).

OPTIONS

–b *basedir* Sets the path to the root directory of the diskless client. Used on the server to execute `rem_drv` for a client. The client machine must be rebooted to unload the driver.

EXAMPLES

EXAMPLE 1 Examples of rem_drv.

The following example removes the `sd` driver from use:

```
example% rem_drv sd
```

The next example removes the driver from the `sun1` diskless client. The driver will not be uninstalled nor unloaded until the client machine is rebooted.

```
example% rem_drv –b /export/root/sun1 sd
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the `sys_devices` privilege. This command is intended to be invoked at `ADMIN_LOW` with effective user ID `0`; if invoked by other users, this command needs the `file_dac_write` privilege.

SEE ALSO

Trusted Solaris 7 Reference Manual

`add_drv`(1M), `drvconfig`(1M)

SunOS 5.7 Reference Manual

`kernel`(1M), `attributes`(5)

NAME

remove_allocatable – Remove entries from allocation databases and delete ancillary file

SYNOPSIS

/usr/sbin/remove_allocatable [–f] –n *name*

DESCRIPTION

`remove_allocatable` removes database entries for allocatable devices and certain non-allocatable devices and deletes an ancillary file used by the `allocate`(1M) command to control access to an allocatable device. `remove_allocatable` removes the device's entries from the `device_allocate`(4) and `device_maps`(4) databases, and it deletes the ancillary file in `/etc/security/dev` for the specified device.

OPTIONS

–f Force the removal of an entry. `remove_allocatable` exits with an error if this option is not specified when an entry with the specified device name no longer exists.

–n *name* Removes the entry for a device named *name* from the `device_allocate` and `device_maps` and deletes the ancillary file of that name in `/etc/security/dev`.

ERRORS

When successful, `remove_allocatable` returns an exit status of `0` (true). `remove_allocatable` returns a nonzero exit status in the event of an error. The exit codes are as follows:

`1` Invocation syntax error

`2` Unknown system error

`3` Device *name* not found. This error occurs only when the –f option is not specified.

`4` Permission denied. User does not have DAC or MAC access to database.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

FILES

`/etc/security/device_allocate`
Mandatory access control file for devices

`/etc/security/tsol/device_maps`
List of physical devices associated with a device name and type

SEE ALSO

Trusted Solaris 7 Reference Manual

`allocate`(1M), `device_allocate`(4), `device_clean`(1M), `device_maps`(4), `add_allocatable`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

in.rexecd, rexecd – Remote execution server

SYNOPSIS

in.rexecd

DESCRIPTION

`in.rexecd` is the server for the `rexec`(3N) routine. The server provides remote execution facilities with authentication based on user names and passwords. It is invoked automatically as needed by `inetd`(1M) , and then executes the following protocol:

1) The server reads characters from the socket up to a null (`\\0`) byte. The resultant string is interpreted as an ASCII number, base 10.

2) If the number received in step 1 is non-zero, it is interpreted as the port number of a secondary stream to be used for the `stderr` . A second connection is then created to the specified port on the client's machine.

3) A null terminated user name of at most 16 characters is retrieved on the initial socket.

4) A null terminated password of at most 16 characters is retrieved on the initial socket.

5) A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.

6) `rexecd` then validates the user as is done at login time and, if the authentication was successful, changes to the user's home directory, and establishes the user and group protections of the user. Access is denied unless the user has the remote login authorization. If the `/etc/nologin` file exists, access is denied. If any of these steps fail the connection is aborted and a diagnostic message is returned.

7) A null byte is returned on the connection associated with the `stderr` and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rexecd` .

SUMMARY OF TRUSTED SOLARIS CHANGES

Login is not allowed unless the user has the `remote login` authorization. If the `/etc/nologin` file exists, the user is not allowed to log in.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`rexec`(3N) , `attributes`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr` , after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

`username too long`	The name is longer than 16 characters.
`password too long`	The password is longer than 16 characters.
`command too long`	The command line passed exceeds the size of the argument list (as configured into the system).
`Login incorrect`	No password file entry for the user name existed.
`Password incorrect`	The wrong password was supplied.
`No remote directory`	The `chdir` command to the home directory failed.
`/usr/bin/sh: ...`	The user's login shell could not be started.

NAME

in.rlogind, rlogind – Remote login server

SYNOPSIS

/usr/sbin/in.rlogind –U –T

DESCRIPTION

`in.rlogind` is the server for the `rlogin`(1) program. The server provides a remote login facility with authentication based on privileged port numbers.

`in.rlogind` is invoked by `inetd`(1M) when a remote login connection is established, and executes the following protocol:

- The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection.
- The server checks the client's source address. If an entry for the client exists in both `/etc/hosts` and `/etc/hosts.equiv` , a user logging in from the client is not prompted for a password. If the address is associated with a host for which no corresponding entry exists in `/etc/hosts` , the user is prompted for a password, regardless of whether an entry for the client is present in `/etc/hosts.equiv` . See `hosts`(4) and `hosts.equiv`(4) .

Once the source port and address have been checked, `in.rlogind` allocates a pseudo-terminal and manipulates file descriptors so that the slave half of the pseudo-terminal becomes the `stdin` , `stdout` , and `stderr` for a login process. The login process is an instance of the `login`(1) program, invoked with the –`r` .

The login process then proceeds with the `in.rshd`(1M) authentication process.

The –`U` option is used to pass the UID of the client to `login`(1) . The –`T` option is used if the client has the trusted path attribute.

The parent of the login process manipulates the master side of the pseudo-terminal, operating as an intermediary between the login process and the client instance of the `rlogin` program. In normal operation, a packet protocol is invoked to provide Ctrl-S and Ctrl-Q type facilities and propagate interrupt signals to the remote programs. The login process propagates the client terminal's baud rate and terminal type, as found in the environment variable, `TERM` ; see `environ`(4) .

SUMMARY OF TRUSTED SOLARIS CHANGES

Two new options (–`U` and –`T`) are used in the call to `login`(1) .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`login`(1) , `in.rshd`(1M) , `inetd`(1M) , `inetd.conf`(4)

SunOS 5.7 Reference Manual

`rlogin`(1) , `environ`(4) , `hosts`(4) , `hosts.equiv`(4) , `attributes`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr` , after which any network connections are closed. An error is indicated by a leading byte with a value of 1.

`Hostname for your address unknown.`	No entry in the host name database existed for the client's machine.
`Try again.`	A *fork* by the server failed.
`/usr/bin/sh: ...`	The user's login shell could not be started.

NOTES

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but it is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

NAME

route – Manually manipulate the routing tables

SYNOPSIS

route [–fnvq] *command* [[*modifiers*] *args*]

route [–fnvq] [–t *file1*] add | change | delete | get [host | net]*destination* [*gateway* [*args*]] [*extended_metric*]

route [–n] monitor

route [–n] flush

DESCRIPTION

route manually manipulates the network routing tables. These tables are normally maintained by the system routing daemon, by routed(1M), or through default routes and redirect messages from routers.

This utility supports a limited number of general options, but a rich command language. It enables the user to specify any arbitrary request that could be delivered via the programmatic interface discussed in route(7P).

route uses a routing socket and the new message types RTM_ADD, RTM_DELETE, RTM_GET, and RTM_CHANGE.

route must inherit the sys_net_config privilege to operate directly on the routing table for the specific host or network indicated by *destination*. It also must run with a uid of 0, or have the file_dac_read and file_dac_write privileges.

Trusted Solaris Routing

In the Trusted Solaris environment, as in the Solaris operating environment, routing can be configured by the administrator to be *static*, that is, determined by a router or route description in a static file, or *dynamic*, that is, determined at the time of routing. By default, dynamic routing is in effect. The static routing file to establish default gateways (routers) in both environments is /etc/defaultrouter.

In the Trusted Solaris environment, an administrator can establish gateways for specific networks and default gateways in the file /etc/tsolgateways. Routing decisions are made in the order:

1. Get routing information from the file /etc/tsolgateways. If it does not exist,
2. Get routing information from the file /etc/defaultrouter. If it does not exist,
3. Start dynamic routing.

For trusted routing, security attributes must also be associated with a route. The additional security routing information (SRI) includes sensitivity label range, CIPSO DOI, RIPSO label, and RIPSO error. The SRI and the simple metric

together compose the extended metric (*extended_metric*), which is necessary for trusted routing.

If –e is specified in *extended_metric*, the metric and SRI are obtained from a file composed of a series of lines, each specifying an extended metric value. Both the –e and –m options use the same format. For readability only, the one-line format is shown here as two lines:

```
metric= val,min_sl=val,max_sl=val,doi= val,
ripso_label= val,ripso_error=val,ripso_only,cipso_only,msix_only
```

The *val* for metric is an integer from 0 to 15. The *val* for `min_sl` and `max_sl` is a sensitivity label in either hex or string form. The *val* for `doi` is a nonzero integer. The *val* for `ripso_label` is the classification, followed by a space, followed by a list of protections separated by semicolons (`;`). Both the classification and the protections are specified either in hex or string form. The *val* for `ripso_error` is a list of protections separated by semicolons (`;`). They are specified in either hex or string form. Note that if *val* contains a space, it must be protected by double quotes.

The three keywords, `ripso_only`, `cipso_only`, and `msix_only`, do not have values. They indicate that a route can only forward packets having `RIPSO`, `CIPSO`, or `MSIX` labels. They must be specified if a `SUN_RIPSO`, `SUN_CIPSO`, or `SUN_MSIX` gateway is involved in a route.

Some keywords are necessary, and others are optional. The following rules apply when specifying the extended metric information.

- `metric`, `min_sl`, and `max_sl` must be specified.
- `ripso_label` and `ripso_error` must both be present or both be absent.
- If `cipso_only` is specified, `doi` must be specified; and no `ripso_label`, `ripso_error`, `ripso_only`, or `msix_only` can be specified.
- If `ripso_only` is specified, `ripso_label` and `ripso_error` must be specified; and no `doi`, `cipso_only`, or `msix_only` can be specified.
- If `msix_only` is specified, no `doi`, `ripso_label`, `ripso_error`, `cipso_only`, or `ripso_only` can be specified.

When the –e option is used, emetrics are generated for a route. These emetrics are used for accreditation checks when selecting a route.

Without the –e option, no emetric is generated. If the command adds a remote route, the template of the gateway will be used for accreditation checks when selecting a route, since no emetric is available.

OPTIONS

`–f` Flush the routing tables of all gateway entries. If this is used in conjunction with one of the commands described above, `route` flushes the gateways before performing the command.

`–n` Prevent attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down on your local net, and you need a route before you can contact the name server.

`–v` (Verbose) Print additional details.

`–q` Suppress all output.

`–t` *file1* Obtain extended metric information from *file1*, where *file1* has the same format as `/etc/tsolgateways`.

Commands

`route` executes one of four *commands* on a route to a *destination*. Two additional *commands* operate globally on all routing information. The (six) commands are:

`add` Add a route.

`change` Change aspects of a route (such as its gateway).

`delete` Delete a specific route.

`flush` Remove all gateway entries from the routing table.

`get` Look up and display the route for a destination.

`monitor` Continuously report any changes to the routing information base, routing lookup misses, or suspected network partitionings.

The add, delete, and change commands have the following syntax:

`route` [`–fnvq`] *command* [`–net | –host`] *destination gateway* [*extended_metric*]

where

destination Is the destination host or network.

gateway Is the next-hop intermediary via where packets should be routed.

extended_metric Is one of:

`–e` *file*
or
`–m` *emetric_val* ... `–m` *emetric_val*

The –e or –m option is required for add commands, and must be nonzero if the route utilizes one or more gateways. These options are used to specify extended metric information associated with a route. See the explanation in the section Trusted Solaris Routing under DESCRIPTION.

OPERANDS

route executes its commands on routes to destinations.

Destinations

All symbolic names specified for a *destination* or *gateway* are looked up first as a host name, using gethostbyname(3N). If this lookup fails, getnetbyname(3N) is used to interpret the name as that of a network.

An optional modifier may be included on the command line before a *destination*, to force how route interprets a destination:

–host Forces the destination to be interpreted as a host.

–net Forces the destination to be interpreted as a network.

Routes to a particular host may be distinguished from those to a network by interpreting the Internet address specified as the *destination*. If the *destination* has a "local address part" of INADDR_ANY, or if the *destination* is the symbolic name of a network, then the route is assumed to be to a network; otherwise, it is presumed to be a route to a host.

For example, the route:

```
128.32 is interpreted as –host 128.0.0.32
128.32.130 is interpreted as –host 128.32.0.130
–net 128.32 is interpreted as 128.32.0.0
–net 128.32.130 is interpreted as 128.32.130.0
```

If the destination is directly reachable by way of an interface requiring no intermediary system to act as a gateway, this can be indicated by including one of two optional modifiers after the destination: The interface modifier can be included or a *metric* of 0 can be specified. These modifiers are illustrated in the following alternative examples:

```
route add default hostname interface
route add default hostname 0
```

hostname is the name or IP address associated with the network interface all packets should be sent over. On a host with a single network interface, *hostname* is normally the same as the nodename returned by uname –n (see uadmin(1M)).

In the above examples, the route does not refer to a gateway, but rather to one of the machine's interfaces. Destinations matching such a route are sent out on the interface identified by the *gateway* address. For interfaces using the ARP protocol, this type of route is used to specify *all destinations are local*. That is, a host should use ARP for all addresses by adding a default route using one of the two commands listed above.

The optional –netmask qualifier is intended to manually add subnet routes with netmasks different from that of the implied network interface. The implicit network mask generated in the AF_INET case can be overridden by making sure this option, and an ensuing address parameter (to be interpreted as a network mask), follows the destination parameter.

Routing Flags

Routes have associated flags which influence operation of the protocols when sending to destinations matched by the routes. These flags may be set (or sometimes cleared) by including the following corresponding modifiers on the command line:

Modifier	Flag	Description
–cloning	RTF_CLONING	generates a new route on use
–xresolve	RTF_XRESOLVE	emit mesg on use (for external lookup)
–iface	~RTF_GATEWAY	destination is directly reachable
–static	RTF_STATIC	manually added route
–nostatic	~RTF_STATIC	pretend route added by kernel or daemon
–reject	RTF_REJECT	emit an ICMP unreachable when matched
–blackhole	RTF_BLACKHOLE	silently discard pkts (during updates)
–proto1	RTF_PROTO1	set protocol specific routing flag #1
–proto2	RTF_PROTO2	set protocol specific routing flag #2
–llinfo	RTF_LLINFO	validly translates proto addr to link addr

The optional modifiers:

```
–rtt,
–rttvar,
–sendpipe,
–recvpipe,
–mtu,
–hopcount,
–expire,
–ssthresh
```

provide initial values to quantities maintained in the routing entry by transport level protocols, such as TCP. These may be individually locked by preceding each such modifier to be locked by the –`lock` meta-modifier, or one can specify that all ensuing metrics may be locked by the –`lockrest` meta-modifier.

In a `change` or `add` command where the destination and gateway are not sufficient to specify the route (e.g., when several interfaces have the same address), the –`ifp` or –`ifa` modifiers may be used to determine the interface or interface address.

FILES

`/etc/hosts`	List of host names and net addresses.
`/etc/networks`	List of network names and addresses.
`/etc/defaultrouter`	List of default routes.
`/etc/tsolgateways`	List of trusted gateways and metrics.
`/etc/security/tsol/device_policy`	Policy for trusted devices.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`route` must inherit the `sys_net_config` privilege to operate directly on the routing table for the specific host or network indicated by *destination*. It also must run with a uid of `0`, or have the `file_dac_read` and `file_dac_write` privileges. The file `/etc/security/tsol/device_policy` specifies the access policy for the device special files used by `route`.

The Trusted Solaris environment adds the file `/etc/tsolgateways` and the –`t` option to read it. It also adds the *extended_metric* arguments to handle security routing information.

SEE ALSO

Trusted Solaris 7 Reference Manual

`uadmin`(1M), `in.rdisc`(1M), `netstat`(1M), `routed`(1M), `device_policy`(4), `tsolgateways`(4)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`get`(1), `ioctl`(2), `gethostbyname`(3N), `getnetbyname`(3N), `hosts`(4), `networks`(4), `attributes`(5), ARP(7P), `route`(7P), `routing`(7P)

DIAGNOSTICS

`add` [`host` | `network`] *destination*:*gateway flags*
The specified route is being added to the tables. The values printed are from the routing table entry supplied in the `ioctl`(2) call. If the gateway address used was not the primary address of the gateway (the first one returned by `gethostbyname`(3N)) the gateway address is printed numerically as well as symbolically.

`delete` [`host` | `network`] *destination*:*gateway flags*
As above, but when deleting an entry.

destination `done`
When the –`f` flag is specified, or in the `flush` command, each routing table entry deleted is indicated with a message of this form.

`Network is unreachable`
An attempt to add a route failed because the gateway listed was not on a directly-connected network. Give the next-hop gateway instead.

`not in table`
A delete operation was attempted for an entry that is not in the table.

`routing table overflow`
An add operation was attempted, but the system was unable to allocate memory to create the new entry.

NOTES

All destinations are local assumes that the routers implement the protocol, `proxy arp`. Normally, using `router discovery` (see `in.rdisc`(1M)) is more reliable than using `proxy arp`.

Combining the *all destinations are local* route with subnet or network routes can lead to unpredictable results: the search order as it relates to the *all destinations are local* route are undefined and may vary from release to release.

NAME

in.routed, routed – Network routing daemon

SYNOPSIS

/usr/sbin/in.routed [–s] [–q] [–t] [–g] [–S] [–v] [*logfile*]

DESCRIPTION

`in.routed` is invoked at boot time to manage the network routing tables. The routing daemon uses a variant of the Xerox NS Routing Information Protocol in maintaining up-to-date kernel routing table entries.

In normal operation, `in.routed` listens on `udp`(7P) socket 520 (decimal) for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

When `in.routed` is started, it uses the `SIOCGIFCONF` `ioctl`(2) to find those directly connected interfaces configured into the system and marked "up" (the software loopback interface is ignored). If multiple interfaces are present, it is assumed the host will forward packets between networks. `in.routed` then transmits a *request* packet on each interface (using a broadcast packet if the interface supports it) and enters a loop, listening for *request* and *response* packets from other hosts.

For trusted routing, extended security attributes must be associated with a route along with the simple metric that indicates the number of hops to the destination. The additional security routing information (SRI) includes a sensitivity label range, and can include a CIPSO domain of interpretation, a RIPSO label, and a RIPSO error, and some additional keywords: `ripso_only` , `cipso_only` , and `msix_only` . The SRI combined with the simple metric is called the extended metric, or emetric.

For Trusted Solaris 7 systems, two additional types of packets are exchanged. The first one is *sec_response* , which is like the *response* packet but also carries the SRI for the routes. Similar to the *response* packet, the *sec_response* packet propagates a route while adjusting its metric and SRI one hop at a time. The SRI that is carried in *sec_response* packets cannot be propagated through non-Trusted Solaris gateways.

The second additional packet type is *sec_t_response* , which has the exact format as *sec_response* but with a different command number. The *sec_t_response* packets are used for tunneling. Every time a *response* is sent, a *sec_response* and a *sec_t_response* packet are also sent.

Tunneling can be set up for trusted routing between Trusted Solaris 7 gateways when non-Trusted Solaris gateways exist between the Trusted Solaris 7 gateways. For tunneling to work, all Trusted Solaris gateways must be running Trusted Solaris 2.5.1 or 7, and they must be using the extended `in.routed`(1M) for dynamic routing. Also, the non-Trusted Solaris gateways must be using the standard `in.routed`(1M) for dynamic routing. All gateways must be in the same Intranet. To forward SRI s through non-Trusted Solaris gateways to a target

(sub)network, a Trusted Solaris system sends an unlabeled *sec_t_response* packet in a (sub)network directed broadcast to the target (sub)network on behalf of the non-Trusted Solaris gateway connected to that (sub)network. Trusted Solaris systems on the (sub)network can use the SRI to configure their routing tables correctly, and Trusted Solaris 7 gateways on that (sub)network can propagate the SRI to other (sub)networks. A machine that does tunneling is called the forwarding machine; any Trusted Solaris gateway can be a forwarding machine.

Tunneling is enabled by the existence of the file `/etc/security/tsol/tunnel` , and the target (sub)network addresses are obtained from this file. A Trusted Solaris gateway can be responsible for tunneling to more than one (sub)network. The file is composed of a series of lines, each in the following format:

broadcast_addr

A Trusted Solaris gateway can be responsible for tunneling to more than one (sub)network.

A Trusted Solaris system ignores a *response* packet if it is sent by another Trusted Solaris gateway, because in this case, *sec_response* packets should be used in place of *response* packets. A Trusted Solaris system processes a *response* packet if it is sent by a non-Trusted Solaris gateway. If tunneling is done on behalf of that non-Trusted Solaris gateway, it will process both the *response* packets sent by the non-Trusted Solaris gateway and the *sec_response* packets sent by a remote Trusted Solaris gateway on behalf of the non-Trusted Solaris gateway.

When a *request* packet is received, `in.routed` formulates a reply based on the information maintained in its internal tables. The *response* packet contains a list of known routes, each marked with a “hop count” metric (a count of 16, or greater, is considered “infinite”). The metric associated with each route returned, provides a metric relative to the sender.

sec_response and *sec_t_response* packets are formulated by `AND` ing the emetric of the route with the emetric derived from the outgoing interface. Before the *response* packet is sent, a *sec_response* and a *sec_t_response* packet are sent to the same destination with the same metric and additional SRI .

response , *sec_response* , and *request* packets received by `in.routed` are used to update the routing tables if one of the following conditions is satisfied:

- No routing table entry exists for the destination network or host, and the metric indicates the destination is “reachable” (that is, the hop count is not infinite). For *sec_response* and *sec-t_response* packets, a destination is also unreachable if its SRI restricts all possible packets.

- The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed. The only exception occurs when `in.routed` is supposed to process both the *response* packet from a non-Trusted Solaris gateway and the *sec_response* packet tunneled on behalf of that non-Trusted Solaris gateway. In this situation, if both packets carry routing information for the same route, the SRI from the tunneled *sec_response* packet and the metric from the *response* packet are used.
- The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost effective as the current route.
- The new route describes a shorter route to the destination than the one currently stored in the routing tables; the metric of the new route is compared against the one stored in the table to decide this.

For *sec_response* and *sec_t_response* packets, the last rule above is changed to compare the SRI s as well as the metrics. One route is better than another if (a) its metric is smaller; and (b) its SRI is more relaxed than or equal to that of the other. Note that when comparing the SRI s of two routes, one route cannot always serve as a substitute for the other. For example, if the SRI s of two routes have different sensitivity labels, one SRI cannot be said to be more restrictive, because they restrict different sensitivity label ranges.

If two routes cannot be compared, both routes are kept in the routing table, because they represent two routes to the same destination although with different security characteristics; and both routes are needed.

When an update is applied, `in.routed` records the change in its internal tables and generates a *sec_response* packet and a *response* packet to all directly connected hosts and networks. `in.routed` waits a short period of time (no more than 30 seconds) before modifying the kernel's routing tables to allow possible unstable situations to settle.

In addition to processing incoming packets, `in.routed` also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks.

In addition to the facilities described above, `in.routed` supports the notion of "distant" `passive` and `active` gateways. When `in.routed` is started up, it reads the file `gateways` to find gateways which may not be identified using the

SIOCGIFCONF ioctl . Gateways specified in this manner should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange routing information (that is, they should have a in.routed process running on the machine). Passive gateways are maintained in the routing tables forever. Information regarding their existence is not included in any routing information transmitted. Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway and if no routing information is received for a period of time, the associated route is deleted.

The gateways is comprised of a series of lines, each in the following format:

< net | host> *filename1* gateway *filename2* metric *value* < passive | active >

The net or host keyword indicates if the route is to a network or specific host.

filename1 is the name of the destination network or host. This may be a symbolic name located in networks or hosts , or an Internet address specified in "dot" notation; see inet(3N) .

filename2 is the name or address of the gateway to which messages should be forwarded.

value is a metric indicating the hop count to the destination host or network.

The keyword passive or active indicates if the gateway should be treated as passive or active (as described above).

For both the passive and active gateway, the SRI s of their routes are obtained initially from their remote host template. For an active gateway, further routing information will be exchanged with this machine. If later a *sec_response* packet is received from the active gateway or a *sec_t_response* tunneled on its behalf is received, the initial SRI will be updated. If no *sec_response* packet is ever received for this active gateway, use of the initial SRI is continued. For a passive gateway, no further routing information will be exchanged; therefore, the initial SRI is continuously used.

in.routed must be started from the Trusted path at ADMIN_HIGH . It must inherit the net_mac_read , net_privaddr , net_broadcast , and sys_net_config privileges. If a log file is specified, in.routed must also inherit the file_mac_write privilege.

OPTIONS

–g Is used on internetwork routers to offer a route to the "default" destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers.

–q Is the opposite of the –s option.

–s Forces `in.routed` to supply routing information whether it is acting as an internetwork router or not.

–S If `in.routed` is not acting as an internetwork router it will, instead of entering the whole routing table in the kernel, only enter a default route for each internetwork router. This reduces the the memory requirements without losing any routing reliability.

–t All packets sent or received are printed on standard output. In addition, `in.routed` will not divorce itself from the controlling terminal so that interrupts from the keyboard will kill the process. Any other argument supplied is interpreted as the name of the file in which `in.routed` 's actions should be logged. This log contains information about any changes to the routing tables and a history of recent messages sent and received which are related to the changed route.

–v Allows a logfile (whose name must be supplied) to be created showing the changes made to the routing tables with a timestamp.

FILES

`/etc/gateways`	For distant gateways
`/etc/networks`	Associations of Internet Protocol network numbers with network names
`/etc/hosts`	Internet host table
`/etc/security/tsolgateways`	For trusted routing through listed gateways
`/etc/security/tsol/tunnel`	Tunneling information table for Trusted Solaris hosts

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.routed` should be started at `ADMIN_HIGH` . It must inherit the `net_mac_read` , `net_privaddr` , `net_broadcast` , and `sys_net_config` privileges. If a log file is specified, `in.routed` must also inherit the `file_mac_write` privilege. Because trusted routing considers the security of the route along with the route's metric when making routing decisions, `in.routed` sends two additional types of response packets containing security information for routes: *sec_response* packets for communications with connected Trusted Solaris gateways, and *sec_t_response* packets for tunneling to Trusted Solaris gateways on the other side of non-Trusted Solaris gateways.

SEE ALSO

Trusted Solaris 7 Reference Manual

`route`(1M)

SunOS 5.7 Reference Manual

`ioctl`(2) , `inet`(3N) , `attributes`(5) , `inet`(7P) , `udp`(7P)

NOTES

The kernel's routing tables may not correspond to those of `in.routed` for short periods of time while processes that utilize existing routes exit; the only remedy for this is to place the routing process in the kernel.

`in.routed` should listen to intelligent interfaces, such as an IMP , and to error protocols, such as ICMP , to gather more information.

`in.routed` initially obtains a routing table by examining the interfaces configured on a machine and the `gateways` file. It then sends a request on all directly connected networks for more routing information. `in.routed` does not recognize or use any routing information already established on the machine prior to startup. With the exception of interface changes, `in.routed` does not see any routing table changes that have been done by other programs on the machine, for example, routes added, deleted or flushed by way of the `route`(1M) command. Therefore, these types of changes should not be done while `in.routed` is running. Rather, shut down `in.routed` , make the changes required, and then restart `in.routed` .

NAME

rpcbind – Universal addresses to RPC program number mapper

SYNOPSIS

rpcbind [–d] [–w]

DESCRIPTION

`rpcbind` is a server that converts RPC program numbers into universal addresses. It must be running on the host to be able to make RPC calls on a server on that machine.

When an RPC service is started, it tells `rpcbind` the address at which it is listening, and the RPC program numbers it is prepared to serve. When a client wishes to make an RPC call to a given program number, it first contacts `rpcbind` on the server machine to determine the address where RPC requests should be sent.

`rpcbind` should be started before any other RPC service. Normally, standard RPC servers are started by port monitors, so `rpcbind` must be started before port monitors are invoked.

When `rpcbind` is started, it checks that certain name-to-address translation-calls function correctly. If they fail, the network configuration databases may be corrupt. Since RPC services cannot function correctly in this situation, `rpcbind` reports the condition and terminates.

OPTIONS

The following options are supported:

`–d` Run in debug mode. In this mode, `rpcbind` will not fork when it starts, will print additional information during operation, and will abort on certain errors. With this option, the name-to-address translation consistency checks are shown in detail.

`–w` Do a warm start. If `rpcbind` aborts or terminates on `SIGINT` or `SIGTERM`, it will write the current list of registered services to `/tmp/portmap.file` and `/tmp/rpcbind.file`. Starting `rpcbind` with the `–w` option instructs it to look for these files and start operation with the registrations found in them. This allows `rpcbind` to resume operation without requiring all RPC services to be restarted.

SUMMARY OF TRUSTED SOLARIS CHANGES

`rpcbind` should be run at a sensitivity label of `ADMIN_HIGH` and must be run from the trusted path. `rpcbind` should be run with all privileges. Note, however, that these privileges are made effective only when required for `rpcbind`'s operation. Most are used only when `rpcbind` makes an RPC call on behalf of a privileged `RPCBPROC_CALLIT` client.

FILES

`/tmp/portmap.file` File of registered services used during a warm start.

`/tmp/rpcbind.file` File of registered services used during a warm start.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpcinfo`(1M), `rpcbind`(3N)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

Terminating `rpcbind` with `SIGKILL` will prevent the warm-start files from being written.

All RPC servers must be restarted if the following occurs: `rpcbind` crashes (or is killed with `SIGKILL`) and is unable to to write the warm-start files; `rpcbind` is started without the `–w` option after a graceful termination; or, the warm-start files are not found by `rpcbind`.

NAME

rpc.bootparamd, bootparamd – Boot parameter server

SYNOPSIS

/usr/sbin/rpc.bootparamd [–d]

DESCRIPTION

`rpc.bootparamd` is a server process that provides information from a `bootparams` database to diskless clients at boot time. See `bootparams`(4)

The source for the `bootparams` database is determined by the `nsswitch.conf`(4) file (on the machine running the `rpc.bootparamd` process).

The `rpc.bootparamd` program can be invoked either by `inetd`(1M) or directly from the command line.

OPTIONS

–d Display debugging information.

SUMMARY OF TRUSTED SOLARIS CHANGES

`rpc.bootparamd` requires the trust path attribute with a UID of `0` , and the sensitivity label `ADMIN_LOW` .

FILES

`/etc/bootparams` Boot parameter database.

`/etc/nsswitch.conf` Configuration file for the name-service switch.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M) , `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`bootparams`(4) , `attributes`(5)

NOTES

A diskless client requires service from at least one `rpc.bootparamd` process running on a server that is on the same IP subnetwork as the diskless client.

Some routines that compare hostnames use case-sensitive string comparisons; some do not. If an incoming request fails, verify that the case of the hostname in the file to be parsed matches the case of the hostname called for, and attempt the request again.

NAME

rpc.getpeerinfod – Getpeerinfo service daemon

SYNOPSIS

/usr/sbin/rpc.getpeerinfod

DESCRIPTION

`rpc.getpeerinfo` is an RPC server that returns process attributes for peer processes. It is used to obtain values used for process creation and audit characteristic propagation. The `rpc.getpeerinfo` daemon is normally started through `rc` scripts.

NAME

rpcinfo – Report RPC information

SYNOPSIS

rpcinfo [–m | –s][*host*]

rpcinfo –p [*host*]

rpcinfo –T *transport host prognum* [*versnum*]

rpcinfo [–M] [–s] [*host*]

rpcinfo –l [–T *transport*] *host prognum versnum*

rpcinfo [–n *portnum*] –u *host prognum* [*versnum*]

rpcinfo [–n *portnum*] –t *host prognum* [*versnum*]

rpcinfo –a *serv_address* –T *transport prognum* [*versnum*]

rpcinfo –b [–T *transport*] *prognum versnum*

rpcinfo –d [–T *transport*] *prognum versnum*

DESCRIPTION

`rpcinfo` makes an RPC call to an RPC server and reports what it finds.

In the first synopsis, `rpcinfo` lists all the registered RPC services with `rpcbind` on *host*. If *host* is not specified, the local host is the default. If –s is used, the information is displayed in a concise format.

In the second synopsis, `rpcinfo` lists all the RPC services registered with `rpcbind`, version 2. Note that the format of the information is different in the first and the second synopsis. This is because the second synopsis is an older protocol used to collect the information displayed (version 2 of the `rpcbind` protocol).

The third synopsis makes an RPC call to procedure 0 of *prognum* and *versnum* on the specified *host* and reports whether a response was received. *transport* is the transport which has to be used for contacting the given service. The remote address of the service is obtained by making a call to the remote `rpcbind`.

The fourth synopsis is an extended version of the first. While the default report lists the RPC services that are registered for the user's sensitivity label (including multilevel services), the –M option lists all RPC services that are registered at or below the sensitivity label of the user. If the process has the `net_mac_read` privilege, the list includes all RPC services. These reports include the same information as that produced by the default report plus a multilevel mapping indicator or the sensitivity label at which the RPC service is registered.

The *prognum* argument is a number that represents an RPC program number (see `rpc`(4)).

If a *versnum* is specified, `rpcinfo` attempts to call that version of the specified *prognum*. Otherwise, `rpcinfo` attempts to find all the registered version numbers for the specified *prognum* by calling version 0, which is presumed not to

exist; if it does exist, `rpcinfo` attempts to obtain this information by calling an extremely high version number instead, and attempts to call each registered version. Note that the version number is required for `–b` and `–d` options.

The `EXAMPLES` section describe other ways of using `rpcinfo`.

OPTIONS

`–T` *transport*
: Specify the transport on which the service is required. If this option is not specified, `rpcinfo` uses the transport specified in the `NETPATH` environment variable, or if that is unset or `NULL`, the transport in the `netconfig`(4) database is used. This is a generic option, and can be used in conjunction with other options as shown in the `SYNOPSIS`.

`–a` *serv_address*
: Use *serv_address* as the (universal) address for the service on *transport* to ping procedure 0 of the specified *prognum* and report whether a response was received. The `–T` option is required with the `–a` option.

`–b`
: Make an RPC broadcast to procedure 0 of the specified *prognum* and *versnum* and report all hosts that respond. If *transport* is specified, it broadcasts its request only on the specified transport. If broadcasting is not supported by any transport, an error message is printed. Use of broadcasting requires the `net_broadcast` privilege.

`–d`
: Delete registration for the RPC service of the specified *prognum* and *versnum*. If *transport* is specified, unregister the service on only that transport, otherwise unregister the service on all the transports on which it was registered. Only the owner of a service or a process with the `net_setid` privilege can delete a registration. The `net_mac_read` privilege is required to delete a multilevel mapping. The `net_privaddr` privilege is required to delete a mapping to a transport that uses a privileged address.

`–l`
: Display a list of entries with a given *prognum* and *versnum* on the specified *host*. Entries are returned for all transports in the same protocol family as that used to contact the remote `rpcbind`.

If *versnum* is not specified, `rpcinfo` tries to ping all available version numbers for that program number. This option avoids calls to remote `rpcbind` to find the address of the service. The *serv_address* is specified in universal address format of the given transport.

–m	Display a table of statistics of `rpcbind` operations on the given *host*. The table shows statistics for each version of `rpcbind` (versions 2, 3 and 4), giving the number of times each procedure was requested and successfully serviced, the number and type of remote call requests that were made, and information about RPC address lookups that were handled. This is useful for monitoring RPC activities on *host*.
–M	This extended reporting option lists all RPC services that are registered at or below the sensitivity label of the process. If the process has the `net_mac_read` privilege, the list includes all RPC services regardless of sensitivity label. These reports include the same information as that produced by the default report plus a multilevel mapping indicator or the sensitivity label of the RPC service. Note that the process will require the `sys_trans_label` privilege in order to display the names of sensitivity labels not dominated by the process.
–n *portnum*	Use *portnum* as the port number for the –`t` and –`u` options instead of the port number given by `rpcbind`. Use of this option avoids a call to the remote `rpcbind` to find out the address of the service. This option is made obsolete by the –`a` option.
–p	Probe `rpcbind` on *host* using version 2 of the `rpcbind` protocol, and display a list of all registered RPC programs. If *host* is not specified, it defaults to the local host. Note that version 2 of the `rpcbind` protocol was previously known as the portmapper protocol.
–s	Display a concise list of all registered RPC programs on *host*. If *host* is not specified, it defaults to the local host.
–t	Make an RPC call to procedure 0 of *prognum* on the specified *host* using TCP, and report whether a response was received. This option is made obsolete by the –`T` option as shown in the third synopsis.

–u	Make an RPC call to procedure 0 of *prognum* on the specified *host* using UDP, and report whether a response was received. This option is made obsolete by the –T option as shown in the third synopsis.

EXAMPLES

EXAMPLE 1 RPC services.

To show all of the RPC services registered on the local machine use:

```
example% rpcinfo
```

To show all of the RPC services registered with `rpcbind` on the machine named `klaxon` use:

```
example% rpcinfo klaxon
```

The information displayed by the above commands can be quite lengthy. Use the –s option to display a more concise list:

```
example% rpcinfo –s klaxon
```

program	versio	metid(s)	service	owner
100000	2,3,4	tcp,udp,ticlts,ticots,ticotsord	rpcbind	superuser
100008	1	ticotsord,ticots,ticlts,udp,tcp	walld	superuser
100002	2,1	ticotsord,ticots,ticlts,udp,tcp	rusersd	superuser
100001	2,3,4	ticotsord,ticots,tcp,ticlts,udp	rstatd	superuser
100012	1	ticotsord,ticots,ticlts,udp,tcp	sprayd	superuser
100007	3	ticotsord,ticots,ticlts,udp,tcp	ypbind	superuser
100029	1	ticotsord,ticots,ticlts	keyserv	superuser
100078	4	ticotsord,ticots,ticlts	kerbd	superuser
100024	1	ticotsord,ticots,ticlts,udp,tcp	status	superuser
100021	2,1	ticotsord,ticots,ticlts,udp,tcp	nlockmgr	superuser
100020	1	ticotsord,ticots,ticlts,udp,tcp	llockmgr	superuser

To show whether the RPC service with program number *prognum* and version *versnum* is registered on the machine named `klaxon` for the transport TCP use:

```
example% rpcinfo –T tcp klaxon prognum versnum
```

To show all RPC services registered with version 2 of the `rpcbind` protocol on the local machine use:

```
example% rpcinfo –p
```

To delete the registration for version 1 of the `walld` (program number `100008`) service for all transports use:

```
example# rpcinfo –d 100008 1
```

or

```
example# rpcinfo –d walld 1
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The –`M` option is added, and it requires privilege. The –`b` and –`d` options require privilege. See the `OPTIONS` definitions for details.

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpcbind`(1M), `rpc`(3N)

SunOS 5.7 Reference Manual

`netconfig`(4), `rpc`(4), `attributes`(5)

NAME

rpc.nisd, nisd – NIS+ service daemon

SYNOPSIS

/usr/sbin/rpc.nisd [–ACDFhlv] [–Y [–B [–t *netid*]]] [–d *dictionary*] [–L *load*] [–S *level*]

DESCRIPTION

The rpc.nisd daemon is an RPC service that implements the NIS+ service. This daemon must be running on all machines which serve a portion of the NIS+ namespace. A Trusted Solaris 7 system must be the root master in the NIS+ configuration.

rpc.nisd is usually started from a system startup script. It must be started through a role that has a UID of 0 and run with a sensitivity label of ADMIN_LOW . (For example, the role might be assigned the predefined NIS+ security administration and NIS+ administration profiles.) Upon startup, rpc.nisd must inherit the net_mac_read , net_upgrade_sl , and proc_setsl privileges.

OPTIONS

–A Authentication verbose mode. The daemon logs all the authentication related activities to syslogd(1M) with LOG_INFO priority.

–C Open diagnostic channel on /dev/console .

–D Debug mode (don't fork).

–F Force the server to do a checkpoint of the database when it starts up. Forced checkpoints may be required when the server is low on disk space. This option removes updates from the transaction log that have propagated to all of the replicas.

–h Print list of options.

–u Allow updates from non-Trusted Solaris TCB clients.

–v Verbose. With this option, the daemon sends a running narration of what it is doing to the syslog daemon (see syslogd(1M)) at LOG_INFO priority. This option is most useful for debugging problems with the service (see also –A option).

–Y ypserve and other NIS (YP) compatibility is not supported in Trusted Solaris. Using this option may put the daemon in an unknown state.

–B ypserve and other NIS (YP) compatibility is not supported in Trusted Solaris. Using this option may put the daemon in an unknown state.

–t *netid* Use *netid* as the transport for communication between rpc.nisd and rpc.nisd_resolv . The default transport is ticots(7D) (tcp on SunOS 4.x systems).

–d *dictionary* Specify an alternate dictionary for the NIS+ database. The primary use of this option is for testing. Note that the

string is not interpreted, rather it is simply passed to the `db_initialize( )` function. See `nis_db`(3N) .

–L *number*
: Specify the "load" the NIS+ service is allowed to place on the server. The load is specified in terms of the *number* of child processes that the server may spawn. This *number must* be at least 1 for the callback functions to work correctly. The default is 128.

–S *level*
: Set the authorization security level of the service. The argument is a number between 0 and 2. By default, the daemon runs at security level 2.

  `0` Security level 0 is designed to be used for testing and initial setup of the NIS+ namespace. When running at level 0, the daemon does not enforce any access controls. Any client is allowed to perform any operation, including updates and deletions.

  `1` At security level 1, the daemon accepts both `AUTH_SYS` and `AUTH_DES` credentials for authenticating clients and authorizing them to perform NIS+ operations. This is not a secure mode of operation since `AUTH_SYS` credentials are easily forged. It should not be used on networks in which any untrusted users may potentially have access.

  `2` At security level 2, the daemon only accepts authentication using the security mechanisms configured by `nisauthconf`(1M) . The default security mechanism is `AUTH_DES` . Security level 2 is the default if the –S option is not used.

EXAMPLES

EXAMPLE 1 Setting up the NIS+ service.

The following example sets up the NIS+ service.

```
example% rpc.nisd
```

EXAMPLE 2 Setting Up NIS+ Service Emulating YP With DNS Forwarding

The following example sets up the NIS+ service, emulating YP with DNS forwarding.

```
example% rpc.nisd -YB
```

ENVIRONMENT VARIABLES

`NETPATH` The transports that the NIS+ service will use can be limited by setting this environment variable (see `netconfig`(4)).

SUMMARY OF TRUSTED SOLARIS CHANGES

A Trusted Solaris 7 system must be the root master of the NIS+ configuration. The `rpc.nisd` daemon must inherit the `net_mac_read` , `net_upgrade_sl` , and `proc_setsl` privileges upon startup. The daemon must be started by a role with a UID of `0` and run with a sensitivity label of `ADMIN_LOW` . ypserver and other NIS (YP) compatibility is not supported.

FILES

`/var/nis/data/parent.object`
This file describes the namespace that is logically above the NIS+ namespace. The most common type of parent object is a DNS object. This object contains contact information for a server of that domain.

`/var/nis/data/root.object`
This file describes the root object of the NIS+ namespace. It is a standard XDR -encoded NIS+ directory object that can be modified by authorized clients using the `nis_modify`(3N) interface.

`/etc/init.d/rpc`
Initialization script for NIS+ .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`nis_cachemgr`(1M) , `rpc.nisd_resolv`(1M) , `rpc.nispasswdd`(1M) , `resolv.conf`(4)

SunOS 5.7 Reference Manual

`nisinit`(1M) , `nissetup`(1M) , `nslookup`(1M) , `syslogd`(1M) , `nis_db`(3N) , `netconfig`(4) , `nisfiles`(4) , `attributes`(5) , `ticots`(7D)

NAME

rpc.nisd_resolv, nisd_resolv – NIS+ service daemon

SYNOPSIS

rpc.nisd_resolv [–v | –V][–F [–C *fd*]] [–t *xx*] [–p *yy*]

DESCRIPTION

`rpc.nisd_resolv` is an auxiliary process which provides DNS forwarding service for NIS hosts requests to both `ypserv` and `rpc.nisd` that are running in the NIS compatibility mode. It is generally started by invoking `rpc.nisd`(1M) with the –B option or `ypserv`(1M) with the –d option. Although it is not recommended, `rpc.nisd_resolv` can also be started independently with the following options.

This command is not supported in the Trusted Solaris environment because `ypserv` and other NIS(YP) compatibility is unsupported.

OPTIONS

–F Run in foreground.

–C Use *fd* for service `xprt` (from `nisd`).

fd

–v Verbose. Send output to the `syslog` daemon.

–V Verbose. Send output to `stdout` .

–t Use transport *xx* .

xx

–p Use transient program# *yy* .

yy

SUMMARY OF TRUSTED SOLARIS CHANGES

This command is not supported in the Trusted Solaris environment.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc.nisd`(1M)

SunOS 5.7 Reference Manual

`nslookup`(1M) , `resolv.conf`(4) , `attributes`(5)

NOTES This command requires that the `/etc/resolv.conf` file be setup for communication with a DNS nameserver. The `nslookup` utility can be used to verify communication with a DNS nameserver. See `resolv.conf`(4) and `nslookup`(1M) .

NAME

rpc.nispasswdd, nispasswdd – NIS+ password update daemon

SYNOPSIS

/usr/sbin/rpc.nispasswdd [–a *attempts*] [–c *minutes*] [–D] [–g] [–v]

DESCRIPTION

`rpc.nispasswdd` daemon is an ONC+ RPC service that services password update requests from `nispasswd`(1) . It updates password entries in the NIS+ `passwd` table.

`rpc.nispasswdd` is normally started from a system startup script after the NIS+ server (`rpc.nisd`(1M)) has been started. `rpc.nispasswdd` will determine whether it is running on a machine that is a master server for one or more NIS+ directories. If it discovers that the host is not a master server, then it will promptly exit. It will also determine if `rpc.nisd`(1M) is running in NIS(YP) compatibility mode (the –Y option) and will register as `yppasswdd` for NIS(YP) clients as well.

ypserv and other NIS (YP) compatibility is not supported.

`rpc.nispasswdd` will syslog all failed password update attempts, which will allow an administrator to determine whether someone was trying to "crack" the passwords.

`rpc.nispasswdd` has to be run by a superuser.

OPTIONS

–a *attempts*	Set the maximum number of attempts allowed to authenticate the caller within a password update request session. Failed attempts are `syslogd`(1M) and the request is cached by the daemon. After the maximum number of allowed attempts the daemon severs the connection to the client. The default value is set to `3` .
–c *minutes*	Set the number of minutes a failed password update request should be cached by the daemon. This is the time during which if the daemon receives further password update requests for the same user and authentication of the caller fails, then the daemon will simply not respond. The default value is set to `30` minutes.
–D	Debug. Run in debugging mode.
–g	Generate DES credential. By default the DES credential is not generated for a user if who does not have one. By specifying this option, if a user does not have a credential, then one will be generated and stored in the NIS+ cred table.
–v	Verbose. With this option, the daemon sends a running narration of what it is doing to the `syslog` daemon. This option is useful for debugging problems.

EXIT STATUS

`0` success

`1` an error has occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

`rpc.nispasswdd` must be run with a UID of `0` and with a sensitivity label of `ADMIN_LOW`. On startup, `rpc.nispasswdd` must inherit the `net_mac_read` and `net_upgrade_sl` privileges. ypserv and other NIS (YP) compatibility is not supported.

FILES

`/etc/init.d/rpc` initialization script for NIS+

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc.nisd`(1M), `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`nispasswd`(1), `passwd`(1), `syslogd`(1M), `attributes`(5)

NAME

rpc.tbootparamd – Trusted Solaris boot parameter server

SYNOPSIS

/usr/sbin/rpc.tbootparamd

DESCRIPTION

`rpc.tbootparamd` is a server process that monitors when clients change their state from the booting state to normal state and back.

During booting, a diskless client changes from an unlabeled to a labeled machine. When the change occurs, the client sends out an RPC broadcast message informing its server of the change. Upon receipt of the message, the `rpc.tbootparamd` process running on the server calls `chstate( )` to inform the kernel of the change.

`rpc.tbootparamd` should be started with a uid 0 and a sensitivity label of `ADMIN_LOW`; and it must inherit the `sys_net_config` and `net_mac_read` privileges.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`tbootparam`(1M), `chstate`(2)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

runpd – Run a command for privilege debugging

SYNOPSIS

/usr/sbin/runpd [–p] [–a | –f] *command* [*args*]

DESCRIPTION

The `runpd` command is a debugging utitlity intended for use by administrators and developers. `runpd` turns on the priv_debug process attribute and executes the program specified by *command*. The *command* process inherits the priv_debug process attribute from `runpd`, and privilege-checking logs are generated for it. The logs list privileges that *command* needed to succeed, but lacked. *args* is the optional set of arguments passed as input to *command*.

`runpd` must be invoked from the Trusted Path.

To enable privilege debugging with `runpd`, the `tsol_privs_debug` kernel variable in `/etc/system` must be set to `1`, and entries for `kern.debug`, `daemon.debug`, and `local0.debug` must be uncommented in the `/etc/syslog.conf` file, as in:

```
kern.debug;daemon.debug;local0.debug     /var/log/privdebug.log
```

The string `kern.debug` enables privilege debugging of an application's use of system calls. The `local0.debug` and `daemon.debug` strings enable debugging of privileges interpreted by system daemons (for example, the `sys_trans_label` privilege and X window calls). Multiple strings are separated by semicolons.

OPTIONS

–p Execute *command* with the `trusted_path` process attribute. This option is useful when testing a program (*command*) that requires the attribute.

–a The log will include all privilege debugging records from this and previous executions of `runpd`.

–f The log will include any privilege debugging records generated by *command* or its descendants. `runpd` looks for all process IDs that are greater than or equal to that of *command*. Since process IDs can wrap and child processes may not terminate before *command* terminates, some entries may not be displayed. Use –a to display all records.

EXIT STATUS

`runpd` returns the exit code it receives from *command*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`pattr`(1)

SunOS 5.7 Reference Manual

`syslog.conf`(4), `system`(4), `attributes`(5)

NOTES

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

rwall – Write to all users over a network

SYNOPSIS

/usr/sbin/rwall *hostname…*

/usr/sbin/rwall –n *netgroup…*

/usr/sbin/rwall –h *hostname* –n *netgroup*

DESCRIPTION

`rwall` reads a message from standard input until EOF. It then sends this message, preceded by the line:

```
Broadcast Message …
```

to all users logged in on the specified host machines. With the –n option, it sends to the specified network groups.

OPTIONS

–n *netgroup* Send the broadcast message to the specified network groups.

–h *hostname* Specify the `hostname`, the name of the host machine.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

When it is used to send messages to broadcast addresses rather than to specific hosts, this program needs to inherit the `net_broadcast` privilege to run properly.

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M)

SunOS 5.7 Reference Manual

`listen`(1M), `pmadm`(1M), `sacadm`(1M), `wall`(1M), `attributes`(5)

NOTES

The timeout is fairly short to allow transmission to a large group of machines (some of which may be down) in a reasonable amount of time. Thus the message may not get through to a heavily loaded machine.

NAME

sendmail – Send mail over the internet

SYNOPSIS

/usr/lib/sendmail [–ba] [–bD] [–bd] [–bi] [–bm] [–bp] [–bs] [–bt] [–bv] [–B *type*] [–C *file*] [–d *X*] [–F *fullname*] [–f *name*] [–h *N*] [–M x*value*] [–N*notifications*] [–n] [–O*option =value*] [–o x*value*] [–p *protocol*] [–q [*time*]] [–q *Xstring*] [–R *ret*] [–r *name*] [–t] [–V *envid*] [–v] [–X *logfile*] [*address*…]

DESCRIPTION

`sendmail` sends a message to one or more people, routing the message over whatever networks are necessary. `sendmail` does internetwork forwarding as necessary to deliver the message to the correct place.

`sendmail` is not intended as a user interface routine; other programs provide user-friendly front ends. `sendmail` is used only to deliver pre-formatted messages.

With no flags, `sendmail` reads its standard input up to an EOF, or a line with a single dot, and sends a copy of the letter found there to all of the addresses listed. It determines the network to use based on the syntax and contents of the addresses.

Local addresses are looked up in the local `aliases`(4) file, or in a name service as defined by the `nsswitch.conf`(4) file, and aliased appropriately. In addition, if there is a `.forward` file in a recipient's home directory, `sendmail` forwards a copy of each message to the list of recipients that file contains. Refer to the `NOTES` section for more information about `.forward` files. Aliasing can be prevented by preceding the address with a backslash. Normally the sender is not included in alias expansions. For example, if "john" sends to "group", and "group" includes "john" in the expansion, then the message will not be delivered to "john". See the `MeToo` Processing Option for more information.

There are several conditions under which the expected behavior is for the alias database to be either built or rebuilt. It is important to note that this cannot occur under any circumstances unless root owns *and* has exclusive write permission to the `/etc/mail/aliases*` files.

If a message is found to be undeliverable, it is returned to the sender with diagnostics that indicate the location and nature of the failure; or, the message is placed in a `dead.letter` file in the sender's home directory.

OPTIONS

–ba — Go into `ARPANET` mode. All input lines must end with a `RETURN-LINEFEED`, and all messages will be generated with a `RETURN-LINEFEED` at the end. Also, the `From:` and `Sender:` fields are examined for the name of the sender.

–bd — Run as a daemon in the background, waiting for incoming SMTP connections.

–bD	Run as a daemon in the foreground, waiting for incoming SMTP connections.
–bi	Initialize the aliases(4) database. Root must own *and* have exclusive write permission to the /etc/mail/aliases* files for successful use of this option.
–bm	Deliver mail in the usual way (default).
–bp	Print a summary of the mail queue.
–bs	Use the SMTP protocol as described in RFC 821. This flag implies all the operations of the –ba flag that are compatible with SMTP.
–bt	Run in address test mode. This mode reads addresses and shows the steps in parsing; it is used for debugging configuration tables.
–bv	Verify names only; do not try to collect or deliver a message. Verify mode is normally used for validating users or mailing lists.
–B *type*	Indicate body *type* (7BIT or 8BITMIME).
–C *file*	Use alternate configuration file.
–d *X*	Set debugging value to *X*.
–F *fullname*	Set the full name of the sender.
–f *name*	Sets the name of the "from" person (that is, the sender of the mail).
–h *N*	Set the hop count to *N*. The hop count is incremented every time the mail is processed. When it reaches a limit, the mail is returned with an error message, the victim of an aliasing loop.
–M*xvalue*	Set macro *x* to the specified *value*.
–n	Do not do aliasing.
–N *notifications*	Tag all addresses being sent as wanting the indicated *notifications*, which consists of the word "NEVER" or a comma-separated list of "SUCCESS", "FAILURE", and "DELAY" for successful delivery, failure and a message that is stuck in a queue somwhere. The default is "FAILURE,DELAY".

–o*xvalue*	Set option *x* to the specified *value*. `Processing Options` are described below.
–O*option=value*	Set *option* to the specified *value* (for long from names). `Processing Options` are described below.
–p *protocol*	Set the sending protocol. The *protocol* field can be in form *protocol*`:`*host* to set both the sending protocol and the sending host. For example: –`pUUCP:uunet` sets the sending *protocol* to `UUCP` and the sending host to `uunet`. (Some existing programs use –`oM` to set the *r* and *s* macros; this is equivalent to using –`p`).
–q[*time*]	Process saved messages in the queue at given intervals. If *time* is omitted, process the queue once. *time* is given as a tagged number, with *s* being seconds, *m* being minutes, *h* being hours, *d* being days, and *w* being weeks. For example, –`q1h30m` or –`q90m` would both set the timeout to one hour thirty minutes.
–q *Xstring*	Run the queue once, limiting the jobs to those matching *Xstring*. The key letter *X* can be: `I` to limit based on queue identifier. `R` to limit based on recipient. `S` to limit based on sender. A particular queued job is accepted if one of the corresponding addresses contains the indicated *string*.
–r *name*	An alternate and obsolete form of the –`f` flag.
–R *ret*	Identify the information you want returned if the message bounces; *ret* can be "`HDRS`" for headers only or "`FULL`" for headers plus body.
–t	Read message for recipients. `To:`, `Cc:`, and `Bcc:` lines will be scanned for people to send to. The `Bcc:` line will be deleted before transmission. Any addresses in the argument list will be suppressed. The `NoRecipientAction` Processing Option can be used to change the behaviour when no legal recipients are included in the message.
–v	Go into verbose mode. Alias expansions will be announced, and so forth.

–V *envid* The indicated *envid* is passed with the envelope of the message and returned if the message bounces.

–X *logfile* Log all traffic in and out of `sendmail` in the indicated *logfile* for debugging mailer problems. This produces a lot of data very quickly and should be used sparingly.

Processing Options

There are a number of "random" options that can be set from a configuration file. Options are represented by a single character or by multiple character names. The syntax for the single character names of is:

–O*xvalue*

This sets option *x* to be *value.* Depending on the option, *value* may be a string, an integer, a boolean (with legal values `t`, `T`, `f`, or `F`; the default is `TRUE`), or a time interval.

The multiple character or long names use this syntax:

–O *Longname=argument*

This sets the option *Longname* to be *argument.* The long names are beneficial because they are easier to interpret than the single character names.

Not all processing options have single character names associated with them. In the list below the multiple character name is presented first followed by the single character syntax enclosed in parentheses.

`AliasFile (A`*file*`)`
Specify possible alias file(s).

`AliasWait (a` *N*`)`
If set, wait up to *N* minutes for an "@:@" entry to exist in the `aliases`(4) database before starting up. If it does not appear in *N* minutes, rebuild the database (if the `AutoRebuildAliases` option is also set) or issue a warning. Defaults to `10` minutes.

`AllowBogusHELO`
Allow a `HELO SMTP` command that does not include a host name. By default this option is disabled.

`AutoRebuildAliases (D)`
If set, rebuild the `/etc/mail/aliases` database if necessary and possible. If this option is not set, `sendmail` will never rebuild the `aliases` database unless explicitly requested using –`bi`, or `newaliases`(1) is invoked. Note that in order for the database to be rebuilt, root must own *and* have exclusive write permission to the `/etc/mail/aliases*` files.

`BlankSub (B`*c*`)`
Set the blank substitution character to *c*. Unquoted spaces in addresses are replaced by this character. Defaults to `SPACE` (that is, no change is made).

`CheckAliases (n)`
Validate the RHS of aliases when rebuilding the `aliases`(4) database.

`CheckpointInterval (C`*N*`)`
Checkpoints the queue every *N* (default `10`) addresses sent. If your system crashes during delivery to a large list, this prevents retransmission to any but the last *N* recipients.

`ClassFactor (z`*fact*`)`
The indicated factor *fact* is multiplied by the message class (determined by the `Precedence:` field in the user header and the `P` lines in the configuration file) and subtracted from the priority. Thus, messages with a higher `Priority:` will be favored. Defaults to `1800`.

`ColonOkInAddr`
If set, colons are treated as a regular character in addresses. If not set, they are treated as the introducer to the RFC 822 "group" syntax. This option is on for version 5 and lower configuration files.

`ConnectionCacheSize (k`*N*`)`
The maximum number of open connections that will be cached at a time. The default is `1`. This delays closing the current connection until either this invocation of `sendmail` needs to connect to another host or it terminates. Setting it to `0` defaults to the old behavior, that is, connections are closed immediately.

`ConnectionCacheTimeout (K`*timeout*`)`
The maximum amount of time a cached connection will be permitted to idle without activity. If this time is exceeded, the connection is immediately closed. This value should be small (on the order of ten minutes). Before `sendmail` uses a cached connection, it always sends a `NOOP` (no operation) command to check the connection; if this fails, it reopens the connection. This keeps your end from failing if the other end times out. The point of this option is to be a good network neighbor and avoid using up excessive resources on the other end. The default is `five` minutes.

`ConnectionRateThrottle`
The maximum number of connections permitted per second. After this many connections are accepted, further connections will be delayed. If not set or <= 0, there is no limit.

`DaemonPortOptions (O`*options*`)`
Set server SMTP options. The options are *key=value* pairs. Known keys are:

`Addr`	Address mask (defaults `INADDR_ANY`) The address mask may be a numeric address in dot notation or a network name.
`Family`	Address family (defaults to `INET`)
`Listen`	Size of listen queue (defaults to `10`)
`Port`	Name/number of listening port (defaults to `smtp`)
`ReceiveSize`	The size of the TCP/IP receive buffer.
`SendSize`	The size of the TCP/IP send buffer.

`DefaultCharSet`
Set the default character set to use when converting unlabeled 8 bit input to MIME.

`DefaultUser (g`*gid*`) or (u`*uid*`)`
Set the default group ID for mailers to run in to *gid* or set the default userid for mailers to *uid*. Defaults to `1`. The value can also be given as a symbolic group or user name.

`DeliveryMode (d`*x*`)`
Deliver in mode *x*. Legal modes are:

`i`	Deliver interactively (synchronously).
`b`	Deliver in background (asynchronously).
`d`	Deferred mode — database lookups are deferred until the actual queue run.
`q`	Just queue the message (deliver during queue run).

Defaults to `b` if no option is specified, `i` if it is specified but given no argument (that is, `Od` is equivalent to `Odi`).

`DialDelay`
If a connection fails, wait this many seconds and try again. Zero means "do not retry".

`DontBlameSendmail`
If set, override the file safety checks. This compromises system security and should not be used. See

`http://www.sendmail.org/tips/DontBlameSendmail.html` for more information.

`DontExpandCnames`
If set, $[... $] lookups that do DNS-based lookups do not expand `CNAME` records.

`DontInitGroups`
If set, the `initgroups`(3C) routine will never be invoked. If you set this, agents run on behalf of users will only have their primary (`/etc/passwd`) group permissions.

`DontProbeInterfaces`
If set, `sendmail` will not insert the names and addresses of any local interfaces into the $=w class. If set, you must also include support for these addresses, otherwise mail to addresses in this list will bounce with a configuration error.

`DontPruneRoutes (R)`
If set, do not prune route-addr syntax addresses to the minimum possible.

`DoubleBounceAddress`
If an error occurs when sending an error message, send that "double bounce" error message to this address.

`EightBitMode (8)`
Use 8–bit data handling. This option requires one of the following keys. The key can selected by using just the first character, but using the full word is better for clarity.

`mimify`	Do any necessary conversion of `8BITMIME` to 7–bit.
`pass`	Pass unlabeled 8–bit input through as is.
`strict`	Reject unlabeled 8–bit input.

`ErrorHeader (E`*file/message*`)`
Append error messages with the indicated message. If it begins with a slash, it is assumed to be the pathname of a file containing a message (this is the recommended setting). Otherwise, it is a literal message. The error file might contain the name, email address, and/or phone number of a local postmaster who could provide assistance to end users. If the option is missing or `NULL`, or if it names a file which does not exist or which is not readable, no message is printed.

`ErrorMode (e`*x*`)`
Dispose of errors using mode *x*. The values for *x* are:

`e` Mail back errors and give `0` exit status always.

`m` Mail back errors.

`p` Print error messages (default).

`q` No messages, just give exit status.

`w` Write back errors (mail if user not logged in).

`FallbackMXhost (V`*fallbackhost*`)`
If specified, the *fallbackhost* acts like a very low priority MX on every host. This is intended to be used by sites with poor network connectivity.

`ForkEachJob (Y)`
If set, deliver each job that is run from the queue in a separate process. Use this option if you are short of memory, since the default tends to consume considerable amounts of memory while the queue is being processed.

`ForwardPath (J`*path*`)`
Set the path for searching for users' `.forward` files. The default is `$z/.forward`. Some sites that use the automounter may prefer to change this to `/var/forward/$u` to search a file with the same name as the user in a system directory. It can also be set to a sequence of paths separated by colons; `sendmail` stops at the first file it can successfully and safely open. For example, `/var/forward/$u:$z/.forward` will search first in `/var/forward/` *username* and then in `~`*username*`/.forward` (but only if the first file does not exist). Refer to the `NOTES` section for more information.

`HelpFile (H`*file*`)`
Specify the help file for SMTP.

`HoldExpensive (c)`
If an outgoing mailer is marked as being expensive, don't connect immediately.

`HostsFile`
Set the file to use when doing "file" type access of host names.

`HostStatusDirectory`
If set, host status is kept on disk between `sendmail` runs in the named directory tree. If a full path is not used, then the path is interpreted relative to the queue directory.

`IgnoreDots (i)`
Ignore dots in incoming messages. This is always disabled (that is, dots are always accepted) when reading SMTP mail.

`LogLevel (L`*n*`)`
Set the default log level to *n*. Defaults to `9`.

`(M`*x value*`)`
Set the macro *x* to *value*. This is intended only for use from the command line.

`MatchGECOS (G)`
Try to match recipient names using the GECOS field. This allows for mail to be delivered using names defined in the GECOS field in `/etc/passwd` as well as the login name.

`MaxDaemonChildren`
The maximum number of children the daemon will permit. After this number, connections are rejected. If not set or <=0, there is no limit.

`MaxHopCount (h`*N*`)`
The maximum hop count. Messages that have been processed more than *N* times are assumed to be in a loop and are rejected. Defaults to `25`.

`MaxMessageSize`
The maximum size of messages that will be accepted (in bytes).

`MaxMimeHeaderLength=`*M*`[/`*N*`]`
Sets the maximum length of certain MIME header field values to *M* characters. For some of these headers which take parameters, the maximum length of each parameter is set to *N* if specified. If `/`*N* is not specified, one half of *M* will be used. By default, these values are `0`, meaning no checks are done.

`MaxQueueRunSize`
If set, limit the maximum size of any given queue run to this number of entries. This stops reading the queue directory after this number of entries is reached; job priority is not used. If not set, there is no limit.

`MeToo (M)`
Send to me too, even if I am in an alias expansion.

`MaxRecipientsPerMessage`
If set, allow no more than the specified number of recipients in an SMTP envelope. Further recipients receive a `452` error code and are deferred for the next delivery attempt.

`MinFreeBlocks (b`*N/M*`)`
Insist on at least *N* blocks free on the file system that holds the queue files before accepting email via SMTP. If there is insufficient space, `sendmail` gives a `452` response to the `MAIL` command. This invites the sender to try

again later. The optional *M* is a maximum message size advertised in the ESMTP `EHLO` response. It is currently otherwise unused.

`MinQueueAge`
The amount of time a job must sit in the queue between queue runs. This allows you to set the queue run interval low for better responsiveness without trying all jobs in each run. The default value is `0`.

`MustQuoteChars`
Characters to be quoted in a full name phrase. `&,;:\()[]` are quoted automatically.

`NoRecipientAction`
Set action if there are no legal recipient files in the message. The legal values are:

`add-apparently-to`	Add an `Apparently-to:` header with all the known recipients (which may expose blind recipients).
`add-bcc`	Add an empty `Bcc:` header.
`add-to`	Add a `To:` header with all the known recipients (which may expose blind recipients).
`add-to-undisclosed`	Add a `To: undisclosed-recipients:` header.
`none`	Do nothing, leave the message as it is.

`OldStyleHeaders (o)`
Assume that the headers may be in old format, that is, spaces delimit names. This actually turns on an adaptive algorithm: if any recipient address contains a comma, parenthesis, or angle bracket, it will be assumed that commas already exist. If this flag is not on, only commas delimit names. Headers are always output with commas between the names.

`OperatorChars or $o`
Defines the list of characters that can be used to separate the components of an address into tokens.

`PostmasterCopy (P`*`postmaster`*`)`
If set, copies of error messages will be sent to the named *postmaster*. Only the header of the failed message is sent. Since most errors are user problems, this is probably not a good idea on large sites, and arguably contains all

sorts of privacy violations, but it seems to be popular with certain operating systems vendors.

`PrivacyOptions (p`*opt,opt,...*`)`
Set privacy options. Privacy is really a misnomer; many of these are just a way of insisting on stricter adherence to the SMTP protocol.

The `goaway` pseudo-flag sets all flags except `restrictmailq` and `restrictqrun`. If `mailq` is restricted, only people in the same group as the queue directory can print the queue. If queue runs are restricted, only root and the owner of the queue directory can run the queue. `authwarnings` add warnings about various conditions that may indicate attempts to spoof the mail system, such as using a non-standard queue directory.

The options can be selected from:

`authwarnings`	Put X-Authentication-Warning: headers in messages.
`goaway`	Disallow essentially all SMTP status queries.
`needexpnhelo`	Insist on `HELO` or `EHLO` command before `EXPN`.
`needmailhelo`	Insist on `HELO` or `EHLO` command before `MAIL`.
`needvrfyhelo`	Insist on `HELO` or `EHLO` command before `VRFY`.
`noetrn`	Disallow `ETRN` entirely.
`noexpn`	Disallow `EXPN` entirely.
`noreceipts`	Prevent return receipts.
`novrfy`	Disallow `VRFY` entirely.
`public`	Allow open access.
`restrictmailq`	Restrict `mailq` command.
`restrictqrun`	Restrict `–q` command line flag.

`QueueDirectory (Q`*dir*`)`
Use the named *dir* as the queue directory.

`QueueFactor (q`*factor*`)`

Use `factor` as the multiplier in the map function to decide when to just queue up jobs rather than run them. This value is divided by the difference between the current load average and the load average limit (xflag) to determine the maximum message priority that will be sent. Defaults to `600000`.

`QueueLA (x`*LA*`)`
When the system load average exceeds *LA*, just queue messages (that is, do not try to send them). Defaults to `8`.

`QueueSortOrder`
Select the queue sort algorithm. The default value is `Priority`. Other values are `Host` or `Time`.

`QueueTimeout (T`*rtime*`/`*wtime*`)`
Set the queue timeout to *rtime*. After this interval, messages that have not been successfully sent will be returned to the sender. Defaults to five days (`5d`). The optional *wtime* is the time after which a warning message is sent. If it is missing or `0`, then no warning messages are sent.

`RecipientFactor (y`*fact*`)`
The indicated factor *fact* is added to the priority (thus *lowering* the priority of the job) for each recipient, that is, this value penalizes jobs with large numbers of recipients. Defaults to `30000`.

`RefuseLA (X`*LA*`)`
When the system load average exceeds *LA*, refuse incoming SMTP connections. Defaults to `12`.

`RemoteMode (>[`*RemoteMboxHost*`])`
If *RemoteMboxHost* is specified, then *remote-mode* is enabled using this host. If *RemoteMboxHost* is not specified, and if `/var/mail` is remotely mounted, then remote-mode is enabled using the remote mount host. If *RemoteMboxHost* is not specified and `/var/mail` is locally mounted, then remote-mode is disabled.

When remote-mode is enabled, all outgoing messages are sent through that server.

`ResolverOptions (I)`
Tune DNS lookups.

`RetryFactor (Z`*fact*`)`
The indicated factor *fact* is added to the priority every time a job is processed. Thus, each time a job is processed, its priority will be decreased by the indicated value. In most environments this should be positive, since hosts that are down are all too often down for a long time. Defaults to `90000`.

`RunAsUser`
If set, become this user when reading and delivering mail. Intended for use of firewalls where users do not have accounts.

`SafeFileEnvironment`
If set, `sendmail` will do a `chroot` into this directory before writing files.

`SaveFromLine (f)`
Save Unix-style `From` lines at the front of headers. Normally they are assumed redundant and discarded.

`SendMimeErrors (j)`
If set, send error messages in MIME format (see RFC 1341 and RFC 1344 for details).

`ServiceSwitchFile`
Defines the path to the service-switch file. Since the service-switch file is defined in the Solaris operating environment this option is ignored.

`SevenBitInput (7)`
Strip input to seven bits for compatibility with old systems. This should not be necessary.

`SingleLineFromHeader`
If set, `From:` lines that have embedded newlines are unwrapped onto one line.

`SingleThreadDelivery`
If this option and the `HostStatusDirectory` option are both set, use single thread deliveries to other hosts.

`SmtpGreetingMessage or $e`
The initial SMTP greeting message.

`StatusFile (S`*file*`)`
Log statistics in the named `file`.

`SuperSafe (s)`
Be super-safe when running things, that is, always instantiate the queue file, even if you are going to attempt immediate delivery. `sendmail` always instantiates the queue file before returning control to the client under any circumstances.

`TempFileMode (F`*mode*`)`
The file mode for queue files.

`Timeout (r`*timeouts*`)`
Timeout reads after `time` interval. The *timeouts* argument is a list of *keyword=value* pairs. All but *command* apply to client SMTP. For backward

compatibility, a timeout with no *keyword*= part will set all of the longer values. The recognized timeouts and their default values, and their minimum values specified in RFC 1123 section 5.3.2 are:

`command`
command read [1h, 5m]

`connect`
initial connect [0, unspecified]

`datablock`
data block read [1h, 3m]

`datafinal`
reply to final "." in data [1h, 10m]

`datainit`
reply to `DATA` command [5m, 2m]

`fileopen`
file open [60sec, none]

`helo`
reply to `HELO` or `EHLO` command [5m, none]

`hoststatus`
host retry [30m, unspecified]

`iconnect`
first attempt to connect to a host [0, unspecified]

`ident`
`IDENT` protocol timeout [30s, none]

`initial`
wait for initial greeting message [5m, 5m]

`mail`
reply to `MAIL` command [10m, 5m]

`misc`
reply to `NOOP` and `VERB` commands [2m, none]

`queuereturn`
undeliverable message returned [5d]

`queuewarn`
deferred warning [4h]

`quit`
reply to `QUIT` command [2m, none]

`rcpt`
reply to `RCPT` command [1h, 5m]

`rset`
reply to `RSET` command [5m, none]

`TimeZoneSpec (t`*tzinfo*`)`
Set the local time zone info to *tzinfo*, for example, "PST8PDT". Actually, if this is not set, the `TZ` environment variable is cleared (so the system default is used); if set but null, the user's `TZ` variable is used, and if set and non-null, the `TZ` variable is set to this value.

`TryNullMXList (w)`
If you are the "best" (that is, lowest preference) MX for a given host, you should normally detect this situation and treat that condition specially, by forwarding the mail to a UUCP feed, treating it as local, or whatever. However, in some cases (such as Internet firewalls) you may want to try to connect directly to that host as though it had no MX records at all. Setting this option causes `sendmail` to try this. The downside is that errors in your configuration are likely to be diagnosed as "`host unknown`" or "`message timed out`" instead of something more meaningful. This option is deprecated.

`UnixFromLine or $l`
The "From " line used when sending to files or programs.

`UnsafeGroupWrites`
If set, group-writable :include: and .forward files are considered "unsafe", that is, programs and files cannot be directly referenced from such files.

`UseErrorsTo (l)`
If there is an `Errors-To`: header, send error messages to the addresses listed there. They normally go to the envelope sender. Use of this option causes `sendmail` to violate RFC 1123.

`UserDatabaseSpec (U)`
Defines the name and location of the file containing User Database information.

`Verbose (v)`
Run in verbose mode. If this is set, `sendmail` adjusts the `HoldExpensive` and `DeliveryMode` options so that all mail is delivered completely in a single job so that you can see the entire delivery process. The `Verbose` option should *never* be set in the configuration file; it is intended for command line use only.

All options can be specified on the command line using the –`o` flag, but most will cause `sendmail` to relinquish its setuid permissions. The options that will not cause this are `b`, `d`, `e`, `E`, `i`, `L`, `m`, `o`, `p`, `r`, `s`, `v`, `C`, and `7`. Also considered "safe" is `M` (define macro) when defining the `r` or `s` macros.

If the first character of the user name is a vertical bar, the rest of the user name is used as the name of a program to pipe the mail to. It may be necessary to quote the name of the user to keep `sendmail` from suppressing the blanks from between arguments.

If invoked as `newaliases`, `sendmail` rebuilds the alias database, so long as the `/etc/mail/aliases*` files are owned by root *and* root has exclusive write permission. If invoked as `mailq`, `sendmail` prints the contents of the mail queue.

OPERANDS

address address of an intended recipient of the message being sent.

USAGE

See `largefile`(5) for the description of the behavior of `sendmail` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

`sendmail` returns an exit status describing what it did. The codes are defined in `</usr/include/sysexits.h>`.

`EX_OK`	Successful completion on all addresses.
`EX_NOUSER`	User name not recognized.
`EX_UNAVAILABLE`	Catchall. Necessary resources were not available.
`EX_SYNTAX`	Syntax error in address.
`EX_SOFTWARE`	Internal software error, including bad arguments.
`EX_OSERR`	Temporary operating system error, such as “cannot fork”.
`EX_NOHOST`	Host name not recognized.
`EX_TEMPFAIL`	Message could not be sent immediately, but was queued.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –ba, –bd, –bi, –bs, –bt, –bv, –M, and –q options require that sendmail be invoked from the trusted path with UID of 0 and that needed privileges be inherited. The –d and –X options are ignored if sendmail is not invoked from the trusted path. The –bp option will list only queued messages that are dominated by the process. The –p processing option in the configuration file specifies actions to take for mail received at a sensitivity label that is below the recipient's minimum label. The –D option is not supported in the Trusted Solaris environment.

FILES

dead.letter	unmailable text
/etc/mail/aliases	mail aliases file (plain text)
/etc/mail/aliases.dir	database of mail aliases (binary)
/etc/mail/aliases.pag	database of mail aliases (binary)
/etc/mail/sendmail.cf	defines environment for sendmail
/etc/mail/sendmail.cf	defines environment for sendmail
/var/spool/mqueue/*	temp files and queued mail
~/.forward	list of recipients for forwarding messages

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

Trusted Solaris 7 Reference Manual

resolver(3N)

SunOS 5.7 Reference Manual

biff(1B), mail(1), mailx(1), newaliases(1), check-hostname(1M), check-permissions(1M), aliases(4), hosts(4), attributes(5), largefile(5)

Postel, Jon, *Simple Mail Transfer Protocol*, RFC 821, Network Information Center, SRI International, Menlo Park, Calif., August 1982.

Crocker, Dave, *Standard for the Format of ARPA-Internet Text Messages*, RFC 822, Network Information Center, SRI International, Menlo Park, Calif., August 1982.

Costales, Bryan with Eric Allman, *sendmail, Second Edition*, O'Reilly & Associates, Inc., 1997.

NOTES

The `sendmail` program requires a fully qualified host name when starting. A script has been included to help verify if the host name is defined properly (see `check-hostname`(1M)).

The permissions and the ownership of several directories have been changed in order to increase security. In particular, access to `/etc/mail` and `/var/spool/mqueue` has been restricted.

Security restrictions have been placed users using `.forward` files to pipe mail to a program or redirect mail to a file. The default shell (as listed in `/etc/passwd`) of these users must be listed in `/etc/shells`. This restriction does not affect mail that is being redirected to another alias.

Additional restrictions have been put in place on `.forward` and `:include:` files. These files and the directory structure that they are placed in cannot be group- or world-writable (see `check-permissions`(1M)).

NAME

setaudit – Run a command with the audit mask set

SYNOPSIS

setaudit [–u *username*] *command command_args*

DESCRIPTION

`setaudit` invokes a command using the audit characteristics of the specified user, rather than the audit characteristics of the effective uid of the process executing the `setaudit` command. The command can be used to selectively turn on auditing for daemons and commands that are run from the `/etc/rc` scripts. If the –u option is not used, `setaudit` sets the audit characteristics to the context of the user invoking the command; if the option is present, `setaudit` sets the audit characteristics to the context of the specified *username*. Within the set context, `setaudit` then executes the specified *command* with its arguments (*command_args*).

–u *username* — Use the audit characteristics of *username* rather than the audit characteristics of the effective uid of the process executing the `setaudit` command.

`command` *command_args* — The command to execute and its arguments.

To succeed, `setaudit` must have the `file_dac_read` and `sys_audit` privileges in its set of effective privileges.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

EXAMPLES

EXAMPLE 1 To setaudit with the characteristics of a user

To execute the `cat` command on the file `/etc/system` as the user `maverick`, use this:

```
setaudit -u maverick /usr/bin/cat /etc/system
```

EXAMPLE 2 To setaudit with the characteristics of the invoking shell

To execute the `ls` command on the current working directory from the system shell, use the following command:

```
setaudit /sbin/sysh -c ls
```

SEE ALSO

Trusted Solaris 7 Reference Manual

`audit_control`(4), `audit_user`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

setfsattr, newsecfs – Set security attributes on an existing or newly created file system

SYNOPSIS

/usr/sbin/setfsattr {[–a *access-acl*] | [–l *sensitivity-level-range*] | [–m *MLD-prefix*] | [–p *allowed-privilege-set*] | [–P *forced-privilege-set*] | [–s *CMW -Label*] }…{*special* | *filesystem* }

/usr/sbin/newsecfs {[–a *access-acl*] | [–l *sensitivity-level-range*] | [–M] | [–m *MLD-prefix*] | [–o *newfs options*] [–p *allowed-privilege-set*] | [–P *forced-privilege-set*] | [–s *CMW -Label*] }…{*special* | *filesystem* }

DESCRIPTION

`setfsattr` changes the security attributes of a file system. The file system may be specified either as a *filesystem* or as *special* , the device on which the file system resides. *filesystem* must be in `/etc/vfstab` , and it must be unmounted before `setfsattr` is invoked on it. `setfsattr` requires at least one option be specified; if not, an error is returned.

`newsecfs` works similarly to `setfsattr` except that it runs `newfs`(1M) on the file system prior to setting the security attributes, then sets the label on the `lost+found` directory to `[ADMIN_HIGH]` .

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–a *access-acl*
: Set the file system access ACL . The specified ACL must be a valid access ACL .

–l *sensitivity-level-range*
: Set the filesystem sensitivity level range, a semicolon-separated pair of sensitivity labels. The labels must be valid sensitivity labels for the system. The first in the pair is the minimum sensitivity label, and it must be dominated by the second label, the maximum sensitivity label. The default is `ADMIN_LOW;ADMIN_HIGH` .

–M
: Create the root directory of the file system as a multilevel directory (MLD). This option is available only with the `newsecfs` command.

–m *MLD-prefix*	Set the file system MLD prefix. The default is " .MLD. ". The MLD prefix is the string that disables multilevel directory translation in pathname lookup.
–o *newfs options*	Set the file system newfs options. The options must be exactly the same as those expected by the newfs(1M) command. This option is available only with newsecfs .
–p *allowed-privileges*	Set the file system allowed-privilege set, specified as a text-string of comma-separated privilege names. The privileges in the allowed set must include all privileges in the forced set, or the operation fails.
–P *forced-privileges*	Set the filesystem forced-privilege set, specified as a text string of comma-separated privilege names. All privileges in the forced set must also be in the allowed set, or the operation fails.
–s *CMW-Label*	Set the filesystem CMW label.

USAGE

To specify arguments that include semicolons or embedded spaces (such as for the –l and –o options), use quotes to enclose the arguments.

EXAMPLES

EXAMPLE 1 To set an access ACL

```
% setfsattr -a
\\"user:joni:rw-,user::rwx,group::r--,mask::rw-,other::---" filesystem
```

EXAMPLE 2 To create a new file system with a limited label range

To create a new file system with an allowable label range of Confidential to Secret, use this command:

```
% newsecfs -l 'confidential;secret' raw_device
```

RETURN VALUES

setfsattr exits with one of these values:

0 Success.

1 Failure.

SEE ALSO

Trusted Solaris 7 Reference Manual	`fork`(2)
SunOS 5.7 Reference Manual	`mkfs`(1M) , `newfs`(1M) , `terminfo`(4) , `attributes`(5)

NAME

setmnt – Establish mount table

SYNOPSIS

/usr/sbin/setmnt

DESCRIPTION

`setmnt` creates the `/etc/mnttab` table which is needed for both the `mount` and `umount` commands. `setmnt` reads standard input and creates a `mnttab` entry for each line. Input lines have the format:

filesys node

where *filesys* is the name of the file system's "special file" (such as `/dev/dsk/c?d?s?`) and *node* is the root name of that file system. Thus *filesys* and *node* become the first two strings in the mount table entry.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `/etc/mnttab` file must have a sensitivity label of `ADMIN_LOW` and an owner UID of `0`.

FILES

`/etc/mnttab` Mount table file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mnttab`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

BUGS

Problems may occur if *filesys* or *node* are longer than 32 characters. `setmnt` silently enforces an upper limit on the maximum number of `mnttab` entries.

NAME

setuname – Change machine information

SYNOPSIS

setuname [–t] [–n *node*] [–s *name*]

DESCRIPTION

The setuname utility changes the parameter value for the system name and node name. Each parameter can be changed using setuname and the appropriate option.

Either or both the –s and –n options must be given when invoking setuname.

The system architecture may place requirements on the size of the system and network node name. The command will issue a fatal warning message and an error message if the name entered is incompatible with the system requirements.

OPTIONS

The following options are supported:

–t Temporary change. No attempt will be made to create a permanent change.

–n *node* Changes the node name. *node* specifies the new network node name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

–s *name* Changes the system name. *name* specifies new system name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit) SUNWcsxu (64-bit)

SUMMARY OF TRUSTED SOLARIS CHANGES

The setuname command must have the following privileges: file_dac_read, file_dac_write, file_mac_read, and file_mac_write.

SEE ALSO

SunOS 5.7 Reference Manual

attributes(5)

NOTES

setuname attempts to change the parameter values in two places: the running kernel and, as necessary per implementation, to cross system reboots. A temporary change changes only the running kernel.

NAME

share – Make local resource available for mounting by remote systems

SYNOPSIS

share [–F *FSType*] [–o *specific_options*] [–d *description*] [*pathname*]

DESCRIPTION

The `share` command exports, or makes a resource available for mounting, through a remote file system of type *FSType.* If the option –F *FSType* is omitted, the first file system type listed in `/etc/dfs/fstypes` is used as default. For a description of NFS specific options, see `share_nfs`(1M). *pathname* is the pathname of the directory to be shared. When invoked with no arguments, `share` displays all shared file systems.

OPTIONS

–F *FSType*
Specify the filesystem type.

–o *specific_options*
The *specific_options* are used to control access of the shared resource. (See `share_nfs`(1M) for the NFS specific options.) They may be any of the following:

`devices` | `nodevices`
Allow (disallow) opens on character and block devices. The default is `devices`.

Note: In the Trusted Solaris environment, device special files are typically located only in the `/dev` and `/devices` directories in the root file system. All other file systems should be mounted with the `nodevices` option to prevent recognition of devices that may reside in any other directories.

`priv` | `nopriv`
Forced privileges on executables are allowed or disallowed. The default is `priv`.

`rw`
pathname is shared read / write to all clients. This is also the default behavior.

`rw=`*client*[:*client*]...
pathname is shared read / write only to the listed clients. No other systems can access *pathname.*

`ro`
pathname is shared read-only to all clients.

`ro=`*client*[:*client*]...
pathname is shared read-only only to the listed clients. No other systems can access *pathname.*

–d *description*
The –d flag may be used to provide a description of the resource being shared.

EXAMPLES

EXAMPLE 1 A sample of using `share` command.

This line will share the `/disk` file system read-only at boot time.

```
share –F nfs –o ro /disk
```

SUMMARY OF TRUSTED SOLARIS CHANGES

When invoked with no option or with only the –F *FSType* option, the `share` command displays shared file systems. For all other cases, the command must be run with an effective UID of `0`. If the shared file is of the type NFS, then the `sys_nfs` privilege is also required. If the file `/etc/dfs/sharetab` does not exist, this command will create it; therefore this command must be run at the sensitivity label `ADMIN_LOW`. If the file `/etc/dfs/sharetab` exists, this command can be run at any other sensitivity label if it has the `file_mac_write` privilege. To succeed in all cases, this command needs the `file_mac_read` and `file_mac_search` privileges.

FILES

`/etc/dfs/dfstab`	list of `share` commands to be executed at boot time
`/etc/dfs/fstypes`	list of file system types, NFS by default
`/etc/dfs/sharetab`	system record of shared file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mountd`(1M), `nfsd`(1M), `share_nfs`(1M), `shareall`(1M), `unshare`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

Export (old terminology): file system sharing used to be called exporting on SunOS 4.x, so the `share` command used to be invoked as `exportfs`(1B) or `/usr/sbin/exportfs`.

If `share` commands are invoked multiple times on the same file system, the last `share` invocation supersedes the previous—the options set by the last `share` command replace the old options. For example, if read-write permission

was given to `usera` on `/somefs`, then to give read-write permission also to `userb` on `/somefs`:

```
example% share -F nfs -o rw=usera:userb /somefs
```

This behavior is not limited to sharing the root file system, but applies to all file systems.

NAME

shareall, unshareall – Share, unshare multiple resources

SYNOPSIS

shareall [–F *FSType* [, *FSType* …]] [– | *file*]

unshareall [–F *FSType* [, *FSType* …]]

DESCRIPTION

When used with no arguments, `shareall` shares all resources from *file* , which contains a list of `share` command lines. If the operand is a hyphen (–), then the `share` command lines are obtained from the standard input. Otherwise, if neither a *file* nor a hyphen is specified, then the file `/etc/dfs/dfstab` is used as the default.

Resources may be shared by specific file system types by specifying the file systems in a comma-separated list as an argument to –F .

`unshareall` unshares all currently shared resources. Without a –F flag, it unshares resources for all distributed file system types.

OPTIONS

–F *FSType*	Specify file system type. Defaults to the first entry in `/etc/dfs/fstypes` .

SUMMARY OF TRUSTED SOLARIS CHANGES

The `shareall` and `unshareall` commands must be run with an effective UID of `0` . If any file being shared or unshared is of the type NFS , then the command requires the `sys_nfs` privilege [see `share_nfs`(1M)]. If the file `/etc/dfs/sharetab` does not exist, the `shareall` command will create the file; thus, the `shareall` command must be run at the sensitivity level of `ADMIN_LOW` . If the file `/etc/dfs/sharetab` exists, then the `shareall` and `unshareall` commands can be run at any other sensitivity level if they have the `file_mac_write` privilege. To succeed in all cases, the commands need the `file_mac_read` and `file_mac_search` privileges.

FILES

`/etc/dfs/dfstab`	List of `share` commands to be executed at boot time.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`share`(1M) , `unshare`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

share_nfs – Make local NFS file systems available for mounting by remote systems

SYNOPSIS

share [–d *description*] [–F nfs] [–o *specific_options*] *pathname*

DESCRIPTION

The `share` utility makes local file systems available for mounting by remote systems.

If no argument is specified, then `share` displays all file systems currently shared, including NFS file systems and file systems shared through other distributed file system packages.

OPTIONS

The following options are supported:

–d *description*
: Provide a comment that describes the file system to be shared.

–F nfs
: Share NFS file system type.

–o *specific_options*
: Specify *specific_options* in a comma-separated list of keywords and attribute-value-assertions for interpretation by the file-system-type-specific command. If *specific_options* is not specified, then by default sharing will be read-write to all clients.

  specific_options can be any combination of the following:

  `aclok`
  : Allows the NFS server to do access control for NFS Version 2 clients (running SunOS 2.4 or earlier). When `aclok` is set on the server, maximal access is given to all clients. For example, with `aclok` set, if anyone has read permissions, then everyone does. If `aclok` is not set, minimal access is given to all clients.

  `anon=`*uid*
  : Set *uid* to be the effective user ID of unknown users. By default, unknown users are given the effective user ID `UID_NOBODY`. If *uid* is set to –1, access is denied.

  `index=file`
  : Load `file` rather than a listing of the directory containing this file when the directory is referenced by an NFS URL.

`kerberos`

This option has been deprecated in favor of the `sec=`*krb4* option.

`nosub`

Prevents clients from mounting subdirectories of shared directories. For example, if `/exportF` is shared with the `nosub` option on server *fooey* then a NFS client will not be able to do:

```
mount -F nfs fooey:/export/home/mnt
```

`nosuid`

By default, clients are allowed to create files on the shared file system with the setuid or setgid mode enabled. Specifying `nosuid` causes the server file system to silently ignore any attempt to enable the setuid or setgid mode bits.

`nodevices`

By default, clients are allowed to create block and character special devices on the shared file system. Specifying `nodevices` causes the server file system to prevent the creation of such devices.

`nopriv`

By default, clients are allowed to set forced privileges on files on the shared file system. Specifying `nopriv` causes the server file system to prevent the setting of forced privileges.

`public`

Enables NFS browsing of the file system by a Web NFS-enabled browser. Only one file system per server many use this option. The `-ro=`list and `-rw=`list options can be included with this option.

`ro`

Sharing will be read-only to all clients.

`ro=`*access_list*

Sharing will be read-only to the clients listed in *access_list*; overrides the `rw` suboption for the clients specified. See *access_list* below.

`root=`*access_list*

Only root users from the hosts specified in *access_list* will have root access. See *access_list* below. By default, no host has root access, so root users are mapped to an anonymous user ID (see the `anon=`*uid* option described above). Netgroups can be used if the file system shared is using UNIX authentication (`AUTH_SYS`).

`rw`

Sharing will be read-write to all clients.

`rw=`*access_list*

Sharing will be read-write to the clients listed in *access_list*; overrides the `ro` suboption for the clients specified. See *access_list* below.

`sec=`*mode*[`:`*mode*]...

Sharing will use one or more of the specified security modes. The *mode* in the `sec=`*mode* option must be a node name supported on the client. If the `sec=` option is not specified, the default security mode used is `AUTH_SYS`. Multiple `sec=` options can be specified on the command line, although each mode can appear only once. The security modes are defined in `nfssec`(5).

Each `sec=` option specifies modes that apply to any subsequent `window=`, `rw`, `ro`, `rw=`, `ro=` and `root=` options that are provided before another `sec=`option. Each additional `sec=` resets the security mode context, so that more `window=`, `rw`, `ro`, `rw=`, `ro=` and `root=` options can be supplied for additional modes.

`sec=`*none*

If the option `sec=`*none* is specified when the client uses `AUTH_NONE`, or if the client uses a security mode that is not one that the file system is shared with, then the credential of

each NFS request is treated as unauthenticated. See the `anon=`*uid* option for a description of how unauthenticated requests are handled.

`secure`

This option has been deprecated in favor of the `sec=`*dh* option.

`window=`*value*

When sharing with `sec=`*dh* or `sec=`*krb4* set the maximum life time (in seconds) of the RPC request's credential (in the authentication header) that the NFS server will allow. If a credential arrives with a life time larger than what is allowed, the NFS server will reject the request. The default value is 30000 seconds (8.3 hours).

access_list The *access_list* argument is a colon-separated list whose components may be any number of the following:

hostname — The name of a host. With a server configured for DNS naming in the `nsswitch` "hosts" entry, any hostname must be represented as a fully qualified DNS name.

netgroup — A netgroup contains a number of hostnames. With a server configured for DNS naming in the `nsswitch` "hosts" entry, any hostname in a netgroup must be represented as a fully qualified DNS name.

DNS suffix — To use domain membership the server must use DNS to resolve hostnames to IP addresses; that is, the "hosts" entry in the `/etc/nsswitch.conf` specify "dns" ahead of "nis" or "nisplus", since only DNS returns the full domain name of the host. Other name services like NIS or NIS+ cannot be used to resolve hostnames on the server because when mapping an IP address to a hostname they do not return domain information. For example,

NIS or NIS+	129.144.45.9 –> "myhost
DNS	129.144.45.9 –> "myhost.mydomain.mycompany.com"

The DNS suffix is distinguished from hostnames and netgroups by a prefixed dot. For example,

`rw=.mydomain.mycompany.com`

A single dot can be used to match a hostname with no suffix. For example,

`rw=.`

will match "mydomain" but not "mydomain.mycompany.com". This feature can be used to match hosts resolved through NIS and NIS+ rather than DNS.

network — The network or subnet component is preceded by an at-sign (`@`). It can be either a name or a dotted address. If a name, it will be converted to a dotted address by `getnetbyname`(3N). For example,

`=@mynet`

would be equivalent to:

`=@129.144` or `=@129.144.0.0`

The network prefix assumes an octet aligned netmask determined from the zero octets in the low-order part of the address. In the case where network prefixes are not byte-aligned, the syntax will allow a mask length to be specified explicitly following a slash (`/`) delimiter. For example,

`=@mynet/17` or `rw=@129.144.132/17`

where the mask is the number of leftmost contiguous significant bits in the corresponding IP address.

A prefixed minus sign (–) denies access to that component of *access_list*. The list is searched sequentially until a match is found that either grants or denies access, or until the end of the list is reached. For example, if host "terra" is in the "engineering" netgroup, then

```
rw=-terra:engineering
```

will deny access to terra but

```
rw=engineering:-terra
```

will grant access to terra.

OPERANDS

The following operands are supported:

pathname The pathname of the file system to be shared.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `nodevices` and `nopriv` options have been added.

When invoked with no option or with only the option –`F` *FSType*, the `share_nfs` command displays shared file systems. For all other cases, the command must be run with an effective UID of `0`. If the shared file is of the type NFS, then the `sys_nfs` privilege is also required. If the file `/etc/dfs/sharetab` does not exist, this command will create it; therefore this command must be run at the sensitivity label `ADMIN_LOW`. If the file `/etc/dfs/sharetab` exists, this command can be run at any other sensitivity label if it has the `file_mac_write` privilege. To succeed in all cases, this command needs the `file_mac_read` and `file_mac_search` privileges.

FILES

`/etc/dfs/dfstab`	List of share commands to be executed at boot time.
`/etc/dfs/fstypes`	List of system types, NFS by default.
`/etc/dfs/sharetab`	System record of shared file systems.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mount`(1M), `mountd`(1M), `nfsd`(1M), `share`(1M), `unshare`(1M)

SunOS 5.7 Reference Manual

`getnetbyname`(3N), `netgroup`(4), `attributes`(5), `nfssec`(5)

NOTES

If the `sec=` option is presented at least once, all uses of the `window=`, `rw`, `ro`, `rw=`, `ro=`, and `root=` options must come *after* the first `sec=` option. If the `sec=` option is not presented, then `sec=`*sys* is implied.

If one or more explicit sec= options are presented, *sys* must appear in one of the options mode lists for accessing using the AUTH_SYS security mode to be allowed. For example:

```
share –F nfs /var
share –F nfs –o sec=sys /var
```

will grant read-write access to any host using AUTH_SYS, but

```
share –F nfs –o sec=dh /var
```

will grant no access to clients that use AUTH_SYS.

Unlike previous implementations of share_nfs(1M), access checking for the window=, rw, ro, rw=, and ro= options is done per NFS request, instead of per mount request.

Combining multiple security modes can be a security hole is situations where the ro= and rw= options are used to control access to weaker security modes. In this example,

```
share –F nfs –o sec=dh,rw,sec=sys,rw=hosta /var
```

an intruder can forge the IP address for hosta (albeit on each NFS request) to side-step the stronger controls of AUTH_DES. Something like:

```
share –F nfs –o sec=dh,rw,sec=sys,ro /var
```

is safer, because any client (intruder or legitimate) that avoids AUTH_DES will only get read-only access. In general, multiple security modes per share command should only be used in situations where the clients using more secure modes get stronger access than clients using less secure modes.

If rw= and ro= options are specified in the same sec= clause, and a client is in both lists, the order of the two options determines the access the client gets. If client hosta is in two netgroups - group1 and group2 - in this example, the client would get read-only access:

```
share –F nfs –o ro=group1,rw=group2 /var
```

In this example hosta would get read-write access:

```
share –F nfs –o rw=group2,ro=group1 /var
```

If within a sec= clause, both the ro and rw= options are specified, for compatibility, the order of the options rule is not enforced. All hosts would get read-only access, with the exception to those in the read-write list. Likewise, if the ro= and rw options are specified, all hosts get read-write access with the exceptions of those in the read-only list.

The ro= and rw= options are guaranteed to work over UDP and TCP but may not work over other transport providers.

The root= option with AUTH_SYS is guaranteed to work over UDP and TCP but may not work over other transport providers.

The root= option with AUTH_DES and AUTH_KERB is guaranteed to work over any transport provider.

There are no interactions between the root= option and the rw, ro, rw=, and ro= options. Putting a host in the root list does not override the semantics of the other options. The access the host gets is the same as when the root= options is absent. For example, the following share command will deny access to hostb:

```
share –F nfs –o ro=hosta,root=hostb /var
```

The following will give read-only permissions to hostb:

```
share –F nfs –o ro=hostb,root=hostb /var
```

The following will give read-write permissions to hostb:

```
share –F nfs –o ro=hosta,rw=hostb,root=hostb /var
```

If the file system being shared is a symbolic link to a valid pathname, the canonical path (the path which the symbolic link follows) will be shared. For example, if /export/foo is a symbolic link to /export/bar (/export/foo -> /export/bar), the following share command will result in /export/bar as the shared pathname (and not /export/foo).

```
example# share –F nfs /export/foo
```

Note that an NFS mount of server:/export/foo will result in server:/export/bar really being mounted.

This line in the /etc/dfs/dfstab file will share the /disk file system read-only at boot time:

```
share –F nfs –o ro /disk
```

Note that the same command entered from the command line will not share the `/disk` file system unless there is at least one file system entry in the `/etc/dfs/dfstab` file. The `mountd`(1M) and `nfsd`(1M) daemons only run if there is a file system entry in `/etc/dfs/dfstab` when starting or rebooting the system.

NAME

showmount – Show all remote mounts

SYNOPSIS

/usr/sbin/showmount [–ade] [*hostname*]

DESCRIPTION

`showmount` lists all the clients that have remotely mounted a filesystem from *hostname*. This information is maintained by the `mountd`(1M) server on *hostname*, and is saved across crashes in the file `/etc/rmtab`. The default value for *hostname* is the value returned by `hostname`(1).

OPTIONS

–a Print all remote mounts in the format:

hostname : *directory*

where *hostname* is the name of the client, and *directory* is the root of the file system that has been mounted.

–d List directories that have been remotely mounted by clients.

–e Print the list of shared file systems.

SUMMARY OF TRUSTED SOLARIS CHANGES

The list of clients that have remotely mounted a filesystem is maintained by the `mountd`(1M) server.

FILES

`/etc/rmtab` List of clients that have remotely mounted a filesystem from this machine.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`mountd`(1M)

SunOS 5.7 Reference Manual

`hostname`(1), `attributes`(5)

BUGS

If a client crashes, its entry will not be removed from the list of remote mounts on the server.

NAME

snoop – Capture and inspect network packets

SYNOPSIS

snoop [–aCDNPSvV] [–t [r | a | d]] [–c *maxcount*] [–d *device*] [–i *filename*] [–n *filename*] [–o *filename*] [–p *first* [, *last*]] [–s *snaplen*] [–x *offset* [, *length*]] [*expression*]

DESCRIPTION

snoop captures packets from the network and displays their contents. snoop uses both the network packet filter and streams buffer modules to provide efficient capture of packets from the network. Captured packets can be displayed as they are received, or saved to a file for later inspection.

snoop can display packets in a single-line summary form or in verbose multi-line forms. In summary form, only the data pertaining to the highest level protocol is displayed. For example, an NFS packet will have only NFS information displayed. The underlying RPC, UDP, IP, and ethernet frame information is suppressed but can be displayed if either of the verbose options are chosen.

snoop requires an interactive interface.

OPTIONS

–a Listen to packets on /dev/audio (warning: can be noisy).

–C List the code generated from the filter expression for either the kernel packet filter, or snoop's own filter.

–D Display number of packets dropped during capture on the summary line.

–N Create an IP address-to-name file from a capture file. This must be set together with the –i option that names a capture file. The address-to-name file has the same name as the capture file with .names appended. This file records the IP address to hostname mapping at the capture site and increases the portability of the capture file. Generate a .names file if the capture file is to be analyzed elsewhere. Packets are not displayed when this flag is used.

–P Capture packets in non-promiscuous mode. Only broadcast, multicast, or packets addressed to the host machine will be seen.

–S Display size of the entire ethernet frame in bytes on the summary line.

–v Verbose mode. Print packet headers in lots of detail. This display consumes many lines per packet and should be used only on selected packets.

–V Verbose summary mode. This is halfway between summary mode and verbose mode in degree of verbosity. Instead of displaying just the summary line for the highest level protocol in a packet, it displays a summary line for each protocol layer in the packet. For instance, for

an NFS packet it will display a line each for the ETHER, IP, UDP, RPC and NFS layers. Verbose summary mode output may be easily piped through `grep` to extract packets of interest. For example to view only RPC summary lines:

```
example# snoop -i rpc.cap -V | grep RPC
```

`-t [ r | a | d ]` Time-stamp presentation. Time-stamps are accurate to within 4 microseconds. The default is for times to be presented in `d` (delta) format (the time since receiving the previous packet). Option `a` (absolute) gives wall-clock time. Option `r` (relative) gives time relative to the first packet displayed. This can be used with the `-p` option to display time relative to any selected packet.

`-c` *maxcount* Quit after capturing *maxcount* packets. Otherwise keep capturing until there is no disk left or until interrupted with CTRL-C.

`-d` *device* Receive packets from the network using the interface specified by *device*. Usually `le0` or `ie0`. The program `netstat`(1M), when invoked with the `-i` flag, lists all the interfaces that a machine has. Normally, `snoop` will automatically choose the first non-loopback interface it finds.

`-i` *filename* Display packets previously captured in *filename*. Without this option, `snoop` reads packets from the network interface. If a *filename*`.names` file is present, it is automatically loaded into `snoop`'s IP address-to-name mapping table (See `-N` flag).

`-n` *filename* Use *filename* as an IP address-to-name mapping table. This file must have the same format as the `/etc/hosts` file (IP address followed by the hostname).

`-o` *filename* Save captured packets in *filename* as they are captured. During packet capture, a count of the number of packets saved in the file is displayed. If you wish just to count packets without saving to a file, name the file `/dev/null`.

`-p` *first* [, `last`] Select one or more packets to be displayed from a capture file. The *first* packet in the file is packet #1.

`-s` *snaplen* Truncate each packet after *snaplen* bytes. Usually the whole packet is captured. This option is useful if only certain packet header information is required. The packet truncation is done within the kernel giving better utilization of the streams packet buffer. This means less chance of dropped packets due to buffer overflow during periods of high traffic. It also saves disk space when capturing large traces to a

capture file. To capture only IP headers (no options) use a *snaplen* of `34`. For UDP use `42`, and for TCP use `54`. You can capture PC headers with a *snaplen* of `80` bytes. NFS headers can be captured in `120` bytes.

–x *offset* [, *length*] — Display packet data in hexadecimal and text format. The *offset* and *length* values select a portion of the packet to be displayed. To display the whole packet, use an *offset* of `0`. If a *length* value is not provided, the rest of the packet is displayed.

OPERANDS

expression — Select packets either from the network or from a capture file. Only packets for which the expression is true will be selected. If no expression is provided it is assumed to be true.

Given a filter expression, `snoop` generates code for either the kernel packet filter or for its own internal filter. If capturing packets with the network interface, code for the kernel packet filter is generated. This filter is implemented as a streams module, upstream of the buffer module. The buffer module accumulates packets until it becomes full and passes the packets on to `snoop`. The kernel packet filter is very efficient, since it rejects unwanted packets in the kernel before they reach the packet buffer or `snoop`. The kernel packet filter has some limitations in its implementation—it is possible to construct filter expressions that it cannot handle. In this event, `snoop` generates code for its own filter. The `–C` flag can be used to view generated code for either the kernel's or `snoop`'s own packet filter. If packets are read from a capture file using the `–i` option, only `snoop`'s packet filter is used.

A filter *expression* consists of a series of one or more boolean primitives that may be combined with boolean operators (`AND`, `OR`, and `NOT`). Normal precedence rules for boolean operators apply. Order of evaluation of these operators may be controlled with parentheses. Since parentheses and other filter expression characters are known to the shell, it is often necessary to enclose the the filter expression in quotes. The primitives are:

`host` *hostname*

True if the source or destination address is that of *hostname*. The keyword `host` may be omitted if the name does not

conflict with the name of another expression primitive e.g. "`pinky`" selects packets transmitted to or received from the host `pinky` whereas "`pinky and dinky`" selects packets exchanged between hosts `pinky AND dinky`. Normally the IP address is used. With the `ether` qualifier the ethernet address is used, for instance, "`ether pinky`".

ipaddr or *etheraddr*

Literal addresses, both IP dotted and ethernet colon are recognized. For example, "`129.144.40.13`" matches all packets with that IP address as source or destination, and similarly, "`8:0:20:f:b1:51`" matches all packets with the ethernet address as source or destination. An ethernet address beginning with a letter is interpreted as a hostname. To avoid this, prepend a zero when specifying the address. For example, if the ethernet address is "`aa:0:45:23:52:44`", then specify it by add a leading zero to make it "`0aa:0:45:23:52:44`".

`from` or `src`

A qualifier that modifies the following `host`, `net`, *ipaddr*, *etheraddr*, `port` or `rpc` primitive to match just the source address, port, or RPC reply.

`to` or `dst`

A qualifier that modifies the following `host`, `net`, *ipaddr*, *etheraddr*, `port` or `rpc` primitive to match just the destination address, port, or RPC call.

`ether`

A qualifier that modifies the following `host` primitive to resolve a name to an ethernet address. Normally, IP address matching is performed.

`ethertype` *number*

True if the ethernet type field has value *number*. Equivalent to "`ether[12:2] =` *number*".

`ip`, `arp`, `rarp`

True if the packet is of the appropriate ethertype.

`broadcast`

True if the packet is a broadcast packet. Equivalent to "`ether[2:4] = 0xffffffff`".

`multicast`

True if the packet is a multicast packet. Equivalent to "`ether[0] & 1 = 1`".

`apple`

True if the packet is an Apple Ethertalk packet. Equivalent to "`ethertype 0x809b or ethertype 0x803f`".

`decnet`

True if the packet is a DECNET packet.

`greater` *length*

True if the packet is longer than *length*.

`less` *length*

True if the packet is shorter than *length*.

`udp`, `tcp`, `icmp`

True if the IP protocol is of the appropriate type.

`net` *net*

True if either the IP source or destination address has a network number of *net*. The `from` or `to` qualifier may be used to select packets for which the network number occurs only in the source or destination address.

`port` *port*

True if either the source or destination port is *port*. The *port* may be either a port number or name from `/etc/services`. The `tcp` or `udp` primitives may be used to select TCP or UDP ports only. The `from` or `to` qualifier may be used to select packets for which the *port* occurs only as the source or destination.

`rpc` *prog*

[, *vers* [, *proc*]] True if the packet is an RPC call or reply packet for the protocol identified by *prog*. The *prog* may be either the name of an RPC protocol from `/etc/rpc` or a program number. The *vers* and *proc* may be used to further qualify the program *version* and *proc* number, for example, "`rpc nfs,2,0`" selects all calls and replies for the NFS null procedure. The `to` or `from` qualifier may be used to select either call or reply packets only.

`gateway` *host*

True if the packet used *host* as a gateway, that is, the ethernet source or destination address was for *host* but not the IP address. Equivalent to "`ether host` *host* and not host *host*".

`nofrag`

True if the packet is unfragmented or is the first in a series of IP fragments. Equivalent to "`ip[6:2] & 0x1fff = 0`".

`sectype` *type*

True if the packet security type is *type*. The valid values for *type* are `unlabeled`, `tsix`, and `tsol`.

expr relop expr

True if the relation holds, where *relop* is one of `>`, `<`, `>=`, `<=`, `=`, `!=`, and `expr` is an arithmetic expression composed of numbers, packet field selectors, the `length` primitive, and arithmetic operators `+`, `–`, `*`, `&`, `|`, `^`, and `%`. The arithmetic operators within `expr` are evaluated before the relational operator and normal precedence rules apply between the arithmetic operators, such as multiplication before addition. Parentheses may be used to control the order of evaluation. To use the value of a field in the packet use the following syntax:

base`[expr [: size ] ]`
where `expr` evaluates the value of an offset into the packet from a *base* offset which may be `ether`, `ip`, `udp`, `tcp`, or `icmp`. The `size` value specifies the size of the field. If not given, 1 is assumed. Other legal values are 2 and 4. Examples:

"`ether[0] & 1 = 1`" is equivalent to `multicast`.

"`ether[2:4] = 0xffffffff`" is equivalent to `broadcast`.

"`ip[ip[0] & 0xf * 4 : 2] = 2049`" is equivalent to "`udp[0:2] = 2049`".

"`ip[0] & 0xf > 5`" selects IP packets with options.

"`ip[6:2] & 0x1fff = 0`" eliminates IP fragments.

"`udp and ip[6:2]&0x1fff = 0 and udp[6:2] != 0`" finds all packets with UDP checksums.

The `length` primitive may be used to obtain the length of the packet. For instance "`length > 60`" is equivalent to "`greater 60`", and "`ether[length – 1]`" obtains the value of the last byte in a packet.

`and`

Perform a logical `AND` operation between two boolean values. The `AND` operation is implied by the juxtaposition of two boolean expressions, for example "`dinky pinky`" is the same as "`dinky AND pinky`".

`or` or `,`

Perform a logical `OR` operation between two boolean values. A comma may be used instead, for example, "`dinky,pinky`" is the same as "`dinky OR pinky`".

`not` or `!`

Perform a logical `NOT` operation on the following boolean value. This operator is evaluated before `AND` or `OR`.

EXAMPLES

EXAMPLE 1 Sample output from the snoop command.

Capture all packets and display them as they are received:

```
example# snoop
```

Capture packets with host `funky` as either the source or destination and display them as they are received:

```
example# snoop funky
```

Capture packets between `funky` and `pinky` and save them to a file. Then inspect the packets using times (in seconds) relative to the first captured packet:

```
example# snoop –o cap  funky pinky
example$ snoop –i cap  –t r | more
```

Look at selected packets in another capture file:

```
example$ snoop –i pkts  –p99,108
 99   0.0027   boutique -> sunroof     NFS C GETATTR FH=8E6C
100   0.0046   sunroof -> boutique      NFS R GETATTR OK
101   0.0080   boutique -> sunroof      NFS C RENAME FH=8E6C MTra00192 to .nfs08
102   0.0102   marmot -> viper          NFS C LOOKUP FH=561E screen.r.13.i386
103   0.0072   viper -> marmot          NFS R LOOKUP No such file or directory
104   0.0085   bugbomb -> sunroof     RLOGIN C PORT=1023 h
105   0.0005   kandinsky -> sparky    RSTAT C Get Statistics
106   0.0004   beeblebrox -> sunroof    NFS C GETATTR FH=0307
```

```
107   0.0021   sparky -> kandinsky    RSTAT R
108   0.0073   office -> jeremiah       NFS C READ FH=2584
at 40960 for 8192
```

CODE EXAMPLE 1 Sample TSOL packets.

Now select only those TSOL packets in the above capture file:

```
example$ snoop –i pkts  –p99,108  sectype tsol
 99   0.0027   boutique -> sunroof     NFS C GETATTR FH=8E6C
100   0.0046   sunroof -> boutique      NFS R GETATTR OK
101   0.0080   boutique -> sunroof      NFS C RENAME FH=8E6C MTra00192 to .nfs08
```

Packet 101 looks interesting. Take a look in more detail:

```
example$ snoop –i pkts  –v –p101


ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 101 arrived at 16:09:53.59
ETHER:  Packet size = 210 bytes
ETHER:  Destination = 8:0:20:1:3d:94, Sun
ETHER:  Source      = 8:0:69:1:5f:e,  Silicon Graphics
ETHER:  Ethertype = 0800 (IP)
ETHER:
IP:   ----- IP Header -----
IP:
IP:   Version = 4, header length = 20 bytes
IP:   Type of service = 00
IP:         ..0. .... = routine
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length = 196 bytes
IP:   Identification 19846
IP:   Flags = 0X
IP:   .0.. .... = may fragment
IP:   ..0. .... = more fragments
IP:   Fragment offset = 0 bytes
IP:   Time to live = 255 seconds/hops
IP:   Protocol = 17 (UDP)
IP:   Header checksum = 18DC
IP:   Source address = 129.144.40.222, boutique
IP:   Destination address = 129.144.40.200, sunroof
IP:
UDP:  ----- UDP Header -----
UDP:
UDP:  Source port = 1023
UDP:  Destination port = 2049 (Sun RPC)
UDP:  Length = 176
UDP:  Checksum = 0
UDP:
```

```
TSOL: ----- TSOL SECURITY ATTRIBUTES -----
TSOL:
TSOL:  SM Type = 0x0002, Version = 0x3032
TSOL:  Total Length = 200
TSOL:  Attribute Type = 4 (Raw)
TSOL:  Attributes Length = 192
TSOL:  Domain = 0x00000000
TSOL:  Generation = 0x00000000
TSOL:  Attribute Mask = 0x0000856f
TSOL:  Attribute List:
TSOL:   Sensitivity Label = SECRET
TSOL:   Session ID = 35
TSOL:   Clearance = TOP SECRET
TSOL:   Information Label = SECRET ALL EYES
TSOL:   Effective Privilege Mask = 0x2080004108402000020010802000000
TSOL:           89
TSOL:           99
TSOL:           file_dac_read
TSOL:           file_mac_read
TSOL:           ipc_owner
TSOL:           net_downgrade_sl
TSOL:           net_rawaccess
TSOL:           net_upgrade_sl
TSOL:           proc_owner
TSOL:           sys_trans_label
TSOL:           win_upgrade_sl
TSOL:   Process ID = 22540
TSOL:   Effective User ID = 27042
TSOL:   Effective Group ID = 100
TSOL:   Process Attributes Flags = 0x00000001
TSOL:           Trusted Path Flag = 1
TSOL:           Privilege Debug Flag = 0
TSOL:           Trusted Net Process Flag = 0
TSOL:           Label Translation Flags = 0x0
TSOL:           Label View Flags = 0x0
TSOL:
RPC:  ----- SUN RPC Header -----
RPC:
RPC:  Transaction id = 665905
RPC:  Type = 0 (Call)
RPC:  RPC version = 2
RPC:  Program = 100003 (NFS), version = 2, procedure = 1
RPC:  Credentials: Flavor = 1 (Unix), len = 32 bytes
RPC:     Time = 06-Mar-90 07:26:58
RPC:     Hostname = boutique
RPC:     Uid = 0, Gid = 1
RPC:     Groups = 1
RPC:  Verifier   : Flavor = 0 (None), len = 0 bytes
RPC:
NFS:  ----- SUN NFS -----
NFS:
NFS:  Proc = 11 (Rename)
NFS:  File handle = 0000164300000001000800003 05A1C47
NFS:                597A0000000800002046314AFC450000
NFS:  File name = MTra00192
```

```
  NFS:  File handle = 0000164300000001000800000305A1C47
  NFS:                597A0000000800002046314AFC450000
  NFS:  File name = .nfs08
  NFS:
```

CODE EXAMPLE 2 Sample NFS packets.

View just the NFS packets between `sunroof` and `boutique`:

```
example$ snoop –i pkts  rpc nfs and sunroof and boutique

   1   0.0000   boutique -> sunroof    NFS C GETATTR FH=8E6C
   2   0.0046    sunroof -> boutique   NFS R GETATTR OK
   3   0.0080   boutique -> sunroof    NFS C RENAME FH=8E6C MTra00192 to .nfs08
```

Save these packets to a new capture file:

```
$ snoop –i pkts –o pkts.nfs rpc nfs sunroof boutique
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

`/dev/audio` Symbolic link to the system's primary audio device.

`/dev/null` The null file.

`/etc/hosts` Host name database.

`/etc/rpc` RPC program number database.

`/etc/services` Internet services and aliases.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

Except when using the `–i` option alone, this program should be run at the `ADMIN_HIGH` sensitivity label with effective user ID `0` to open the network device. The `file_mac_read`, `file_mac_write`, `file_dac_read`, and `file_dac_write` privileges can override this restriction. This program also must inherit the `sys_net_config` privilege to put the device in promiscuous mode. (In promiscuous mode, you can see all the packets transmitted on the physical network attached to your interface.)

The `sectype` primitive described under `OPTIONS` is new in the Trusted Solaris environment.

SEE ALSO

Trusted Solaris 7 Reference Manual

`netstat`(1M)

SunOS 5.7 Reference Manual

`hosts`(4), `rpc`(4), `services`(4), `attributes`(5), `audio`(7I), `bufmod`(7M), `dlpi`(7P), `iee`(7D), `le`(7D), `pfmod`(7M)

WARNINGS

The processing overhead is much higher for realtime packet interpretation. Consequently, the packet drop count may be higher. For more reliable capture, output raw packets to a file using the –`o` option and analyze the packets off-line.

Unfiltered packet capture imposes a heavy processing load on the host computer—particularly if the captured packets are interpreted realtime. This processing load further increases if verbose options are used. Since heavy use of `snoop` may deny computing resources to other processes, it should not be used on production servers. Heavy use of `snoop` should be restricted to a dedicated computer.

`snoop` does not reassemble IP fragments. Interpretation of higher level protocol halts at the end of the first IP fragment.

`snoop` may generate extra packets as a side-effect of its use. For example it may use a network name service (NIS or NIS+) to convert IP addresses to host names for display. Capturing into a file for later display can be used to postpone the address-to-name mapping until after the capture session is complete. Capturing into an NFS-mounted file may also generate extra packets.

Setting the *snaplen* (–`s` option) to small values may remove header information that is needed to interpret higher level protocols. The exact cutoff value depends on the network and protocols being used. For NFS Version 2 traffic using UDP on 10 Mb/s ethernet, do not set `snaplen` less than `150` bytes. For NFS Version 3 traffic using TCP on 100 Mb/s ethernet, `snaplen` should be `250` bytes or more.

`snoop` requires information from an RPC request to fully interpret an RPC reply. If an RPC reply in a capture file or packet range does not have a request preceding it, then only the RPC reply header will be displayed.

NAME

spray – Spray packets

SYNOPSIS

/usr/sbin/spray [–c *count*] [–d *delay*] [–l *length*] [–t *nettype*] *host*

DESCRIPTION

`spray` sends a one-way stream of packets to *host* using RPC, and reports how many were received, as well as the transfer rate. The *host* argument can be either a name or an Internet address.

`spray` is not useful as a networking benchmark as it uses unreliable connectionless transports, (upd for example). `spray` can report a large number of packets dropped when the drops were caused by `spray` sending packets faster than they can be buffered locally (before the packets get to the network medium).

OPTIONS

–c*count* Specify how many packets to send. The default value of *count* is the number of packets required to make the total stream size 100000 bytes.

–d*delay* Specify how many microseconds to pause between sending each packet. The default is `0`.

–l*length* The *length* parameter is the numbers of bytes in the Ethernet packet that holds the RPC call message. Since the data is encoded using XDR, and XDR only deals with 32 bit quantities, not all values of *length* are possible, and `spray` rounds up to the nearest possible value. When *length* is greater than 1514, then the RPC call can no longer be encapsulated in one Ethernet packet, so the *length* field no longer has a simple correspondence to Ethernet packet size. The default value of *length* is `86 bytes` (the size of the RPC and UDP headers).

–t*nettype* Specify class of transports. Defaults to `netpath`. See `rpc`(3N) for a description of supported classes.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

If the *host* is a broadcast address, this program needs to inherit the `net_broadcast` privilege to run properly.

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc`(3N)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

statd – Network status monitor

SYNOPSIS

/usr/lib/nfs/statd

DESCRIPTION

`statd` is an intermediate version of the status monitor. It interacts with `lockd`(1M) to provide the crash and recovery functions for the locking services on NFS. `statd` keeps track of the clients with processes which hold locks on a server. When the server reboots after a crash, `statd` sends a message to the `statd` on each client indicating that the server has rebooted. The client `statd` processes then inform the `lockd` on the client that the server has rebooted. The client `lockd` then attempts to reclaim the lock(s) from the server.

`statd` on the client host also informs the `statd` on the server(s) holding locks for the client when the client has rebooted. In this case, the `statd` on the server informs its `lockd` that all locks held by the rebooting client should be released, allowing other processes to lock those files.

FILES

`/var/statmon/sm`	Lists hosts and network addresses to be contacted after a reboot.
`/var/statmon/sm.bak`	Lists hosts and network addresses that could not be contacted after last reboot.
`/var/statmon/state`	Includes a number which changes during a reboot.
`/usr/include/rpcsvc/sm_inter.x`	Contains the rpcgen source code for the interface services provided by the statd daemon.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`statd` must be started with a UID of `0`, a sensitivity label of `ADMIN_LOW`, and a clearance of `ADMIN_HIGH`. It must be started from the Trusted Path and must have these privileges: `net_mac_read`, `net_privaddr`, and `net_upgrade_sl`.

`statd` creates the directory `/var/statmon/sm` and all its files, the directory `/var/statmon/sm.bak` and all its files, and the file `/var/statmon/state` at the sensitivity label `ADMIN_LOW`. The directories `/var/statmon/sm` and `/var/statmon/sm.bak` are created with the mode `700`.

SEE ALSO

Trusted Solaris 7 Reference Manual

`lockd`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

The crash of a server is only detected upon its recovery.

NAME

swap – Swap administrative interface

SYNOPSIS

/usr/sbin/swap –a *swapname* [*swaplow*] [*swaplen*]

/usr/sbin/swap –d *swapname* [*swaplow*]

/usr/sbin/swap –l

/usr/sbin/swap –s

DESCRIPTION

`swap` provides a method of adding, deleting, and monitoring the system swap areas used by the memory manager.

OPTIONS

The following options are supported:

–a *swapname*

Add the specified swap area. The –a *swapname* option requires appropriate privilege. *swapname* is the name of the swap file: for example, `/dev/dsk/c0t0d0s1` or a regular file. *swaplow* is the offset in 512-byte blocks into the file where the swap area should begin. *swaplen* is the desired length of the swap area in 512-byte blocks. The value of *swaplen* can not be less than `16`. For example, if *n* blocks are specified, then (*n*–1) blocks would be the actual swap length. *swaplen* must be at least one page in length. One page of memory is equivalent to eight 512-byte blocks. The size of a page of memory can be determined by using the `pagesize` command. See `pagesize`(1). Since the first page of a swap file is automatically skipped, and a swap file needs to be at least one page in length, the minimum size should be a factor of 2 pagesize bytes. The size of a page of memory is machine dependent.

swaplow + *swaplen* must be less than or equal to the size of the swap file. If *swaplen* is not specified, an area will be added starting at *swaplow* and extending to the end of the designated file. If neither *swaplow* nor *swaplen* are specified, the whole file will be used except for the first page. Swap areas are normally added automatically during system startup by the `/sbin/swapadd` script. This script adds all swap areas which have been specified in the `/etc/vfstab` file; for the syntax of these specifications, see `vfstab`(4).

To use an NFS or local file-system *swapname*, you should first create a file using `mkfile`(1M). A local file-system swap file can now be added to the running system by just running the `swap` –a command. For NFS mounted swap files, the server needs to export the file. Do this by performing the following steps:

1. Add the following line to `/etc/dfs/dfstab`:

   `share` –F `nfs` –o `rw=`*clientname*`,root=`*clientname path-to-swap-file*

2. Run `shareall`(1M).
3. Have the client add the following lines to `/etc/vfstab`:

server`:`*path-to-swap-file* `-` *local-path-to-swap-file* `nfs - - -`
local-path-to-swap-file `- -` `swap - - -`

4. Have the client run `mount`:

```
# mount local-path-to-swap-file
```

5. The client can then run `swap` –`a` to add the swap space:

```
# swap –a local-path-to-swap-file
```

–`d` *swapname*
Delete the specified swap area. The –`d` *swapname* option requires appropriate privilege. *swapname* is the name of the swap file: for example, `/dev/dsk/c0t0d0s1` or a regular file. *swaplow* is the offset in 512-byte blocks into the swap area to be deleted. If *swaplow* is not specified, the area will be deleted starting at the second page. When the command completes, swap blocks can no longer be allocated from this area and all swap blocks previously in use in this swap area have been moved to other swap areas.

–`l` List the status of all the swap areas. The output has five columns:

`path` The path name for the swap area.

`dev` The major / minor device number in decimal if it is a block special device; zeroes otherwise.

`swaplo` The *swaplow* value for the area in 512-byte blocks.

`blocks` The *swaplen* value for the area in 512-byte blocks.

`free` The number of 512-byte blocks in this area that are not currently allocated.

The list does not include swap space in the form of physical memory because this space is not associated with a particular swap area.

If `swap` –`l` is run while *swapname* is in the process of being deleted (by `swap` –`d`), the string INDEL will appear in a sixth column of the swap stats.

–`s` Print summary information about total swap space usage and availability:

`allocated` The total amount of swap space in bytes currently allocated for use as backing store.

`reserved` The total amount of swap space in bytes not currently allocated, but claimed by memory mappings for possible future use.

`used` The total amount of swap space in bytes that is either allocated or reserved.

`available` The total swap space in bytes that is currently available for future reservation and allocation.

These numbers include swap space from all configured swap areas as listed by the `–l` option, as well swap space in the form of physical memory.

USAGE

See `largefile`(5) for the description of the behavior of `swap` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

SUMMARY OF TRUSTED SOLARIS CHANGES

When used with the `–a` or `–d` option, this command needs the `sys_mount` privilege to succeed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`shareall`(1M), `vfstab`(4)

SunOS 5.7 Reference Manual

`pagesize`(1), `mkfile`(1M), `getpagesize`(3C), `attributes`(5), `largefile`(5)

WARNINGS

No check is done to see if a swap area being added overlaps with an existing file system.

NAME

sysdef – Output system definition

SYNOPSIS

/usr/sbin/sysdef [–n *namelist*]

/usr/sbin/sysdef [–h] [–d] [–D]

DESCRIPTION

The `sysdef` utility outputs the current system definition in tabular form. It lists all hardware devices, as well as pseudo devices, system devices, loadable modules, and the values of selected kernel tunable parameters.

It generates the output by analyzing the named bootable operating system file (*namelist*) and extracting the configuration information from it.

The default system *namelist* is `/dev/kmem`.

This command needs the `file_mac_read` privilege to succeed.

OPTIONS

–n *namelist*	Specifies a *namelist* other than the default (`/dev/kmem`). The *namelist* specified must be a valid bootable operating system.
–h	Prints the identifier of the current host in hexadecimal. This numeric value is unique across all Sun hosts.
–d	The output includes the configuration of system peripherals formatted as a device tree.
–D	For each system peripheral in the device tree, display the name of the device driver used to manage the peripheral.

EXAMPLES

EXAMPLE 1 Sample sysdef output format

The following example displays the format of the `sysdef –d` output:

```
example%  sysdef –d
Node 'Sun 4/60', unit #0 (no driver)
      Node 'options', unit #0 (no driver)
      Node 'zs', unit #0
      Node 'zs', unit #1
      Node 'fd', unit #0
      Node 'audio', unit #0
      Node 'sbus', unit #0
             Node 'dma', unit #0
             Node 'esp', unit #0
                  Node 'st', unit #1 (no driver)
                  Node 'st', unit #0
                  Node 'sd', unit #2
                  Node 'sd', unit #1
                  Node 'sd', unit #0
             Node 'le', unit #0
             Node 'bwtwo', unit #0
      Node 'auxiliary-io', unit #0
      Node 'interrupt-enable', unit #0
      Node 'memory-error', unit #0
      Node 'counter-timer', unit #0
```

```
        Node 'eeprom', unit #0
```

FILES

`/dev/kmem` Default operating system image.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit) SUNWcsxu (64-bit)

SUMMARY OF TRUSTED SOLARIS CHANGES

`sysdef` needs the `file_mac_read` privilege to succeed.

SEE ALSO

Trusted Solaris 7 Reference Manual

`prtconf`(1M)

SunOS 5.7 Reference Manual

`hostid`(1), `nlist`(3E), `attributes`(5)

NAME

sysh – System shell

SYNOPSIS

sysh [–acefhiknpPrstuvx] [*argument*...]

DESCRIPTION

sysh, the system shell, is a modified version of the Bourne shell, sh(1). sysh is used to control the use of privileges in commands run from the rc scripts. sysh allows any command to be executed but consults profiles for the privileges, user ID (UID), group ID (GID), and sensitivity label (SL) with which the command is to be run.

Usage

The system shell can be run only from a process with the Trusted Path attribute. Refer to the sh(1) man page for a complete usage description. sysh adds the setprof and clist commands.

Commands

setprof [*profilename*]	sysh uses the specified profile to determine security attributes and privileges for executing subsequent commands. This switch is useful when the same command needs to be run with different privileges at different times. The default profile is the "boot" profile, used when sysh starts up and when setprof is called with no arguments.
clist [–hpnilu]	Displays list of the commands that are permitted for the user.
–h	Includes a hexadecimal list of the privileges assigned to each command in the command list.
–p	Includes a list of the privileges assigned to each command in the command list. The list is in text form.
–n	Includes a comma-separated decimal list of the privileges assigned to each command in the command list.
–i	Includes the UID and GID assigned to each command in the command list.
–l	Includes the SL assigned to each command in the command list.

–u	Lists only those commands for which the profile assigned privileges that `sysh` does not have. (See `WARNINGS`.)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsrSUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`tsolprof`(4)

SunOS 5.7 Reference Manual

`sh`(1), `attributes`(5)

WARNINGS

`sysh` normally has all privileges forced so it can run commands with privileges. If `sysh` finds that a command needs privileges that `sysh` is not permitted, a warning message is printed and the command is run with no privileges.

NOTES

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

tbootparam – Send a request to rpc.tbootparamd to inform it that a host is in normal (labeled) state now

SYNOPSIS

/usr/sbin/tbootparam *server_host client_host*

/usr/sbin/tbootparam *client_host*

DESCRIPTION

The first form informs the server *server_host* that the host *client_host* is now in the normal state.

The second form broadcasts a message to all `rpc.tbootparamd` processes listening that the host *client_host* is now in the normal state.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`rpc.tbootparamd`(1M), `chstate`(2).

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

init, telinit – Process control initialization

SYNOPSIS

/sbin/init [0123456abcQqSs]

/etc/telinit [0123456abcQqSs]

DESCRIPTION

`init` is a general process spawner. Its primary role is to create processes from information stored in the file `/etc/inittab` .

Run Level Defined

At any given time, the system is in one of eight possible run levels. A run level is a software configuration under which only a selected group of processes exists. Processes spawned by `init` for each of these run levels are defined in `/etc/inittab` . `init` can be in one of eight run levels, `0`–`6` and `S` or `s` (`S` and `s` are identical). The run level changes when a privileged user runs `/sbin/init` . This sends appropriate signals to the original `init` spawned by the operating system at boot time, saying which run level to invoke.

init and System Booting

When the system is booted, `init` is invoked and the following occurs. First, it reads `/etc/default/init` to set environment variables. This is typically where `TZ` (time zone) and locale-related environments such as `LANG` or `LC_CTYPE` get set.

`init` then looks in `/etc/inittab` for the `initdefault` entry [see `inittab`(4)]. If the `initdefault` entry:

exists — `init` usually uses the run level specified in that entry as the initial run level to enter.

does not exist — `/etc/inittab` , `init` asks the user to enter a run level from the system console.

`S` or `s` — `init` goes to the single-user state. In this state, the system console device (`/dev/console`) is opened for reading and writing and the command `/sbin/su` , [see `su`(1M)], is invoked. Use either `init` or `telinit` to change the run level of the system. Note that if the shell is terminated (using an end-of-file), `init` only re-initializes to the single-user state if `/etc/inittab` does not exist.

`0-6` — `init` enters the corresponding run level. Run levels `0` , `5` , and `6` are reserved states for shutting the system down. Run levels `2` , `3` , and `4` are available as multi-user operating states.

If this is the first time since power up that `init` has entered a run level other than single-user state, `init` first scans `/etc/inittab` for `boot` and `bootwait` entries [see `inittab`(4)]. These entries are performed before any other processing of `/etc/inittab` takes place, providing that the run level entered

matches that of the entry. In this way any special initialization of the operating system, such as mounting file systems, can take place before users are allowed onto the system. `init` then scans `/etc/inittab` and executes all other entries that are to be processed for that run level.

To spawn each process in `/etc/inittab`, `init` reads each entry and for each entry that should be respawned, it forks a child process. After it has spawned all of the processes specified by `/etc/inittab`, `init` waits for one of its descendant processes to die, a `powerfail` signal, or a signal from another `init` or `telinit` process to change the system's run level. When one of these conditions occurs, `init` re-examines `/etc/inittab`.

inittab Additions

New entries can be added to `/etc/inittab` at any time; however, `init` still waits for one of the above three conditions to occur before re-examining `/etc/inittab`. To get around this, `init Q` or `init q` command wakes `init` to re-examine `/etc/inittab` immediately.

When `init` comes up at boot time and whenever the system changes from the single-user state to another run state, `init` sets the `ioctl`(2) states of the console to those modes saved in the file `/etc/ioctl.syscon`. `init` writes this file whenever the single-user state is entered.

Run Level Changes

When a run level change request is made, `init` sends the warning signal (`SIGTERM`) to all processes that are undefined in the target run level. `init` waits five seconds before forcibly terminating these processes by sending a kill signal (`SIGKILL`).

When `init` receives a signal telling it that a process it spawned has died, it records the fact and the reason it died in `/var/adm/utmp` and `/var/adm/wtmp` if it exists [see `who`(1)]. A history of the processes spawned is kept in `/var/adm/wtmp`.

If `init` receives a `powerfail` signal (`SIGPWR`) it scans `/etc/inittab` for special entries of the type `powerfail` and `powerwait`. These entries are invoked (if the run levels permit) before any further processing takes place. In this way `init` can perform various cleanup and recording functions during the powerdown of the operating system.

/etc/defaults/init File

Default values can be set for the following flags in `/etc/default/init`. For example: `TZ` =US/Pacific

`TZ`	Either specifies the timezone information [see `ctime`(3C)] or the name of a timezone information file `/usr/share/lib/zoneinfo`.
`LC_CTYPE`	Character characterization information.
`LC_MESSAGES`	Message translation.

LC_MONETARY	Monetary formatting information.
LC_NUMERIC	Numeric formatting information.
LC_TIME	Time formatting information.
LC_ALL	If set, all other LC_* environmental variables take-on this value.
LANG	If LC_ALL is not set, and any particular LC_* is also not set, the value of LANG is used for that particular environmental variable.

telinit

telinit , which is linked to /sbin/init , is used to direct the actions of init . It takes a one-character argument and signals init to take the appropriate action.

SECURITY

init uses pam(3) for session management. The PAM configuration policy, listed through /etc/pam.conf , specifies the session management module to be used for init . Here is a partial pam.conf file with entries for init using the UNIX session management module.

```
init    session    required    /usr/lib/security/pam_unix.so.1
```

If there are no entries for the init service, then the entries for the "other" service will be used.

OPTIONS

0	Go into firmware.
1	Put the system in system administrator mode. All local file systems are mounted. Only a small set of essential kernel processes are left running. This mode is for administrative tasks such as installing optional utility packages. All files are accessible and no users are logged in on the system.
2	Put the system in multi-user mode. All multi-user environment terminal processes and daemons are spawned. This state is commonly referred to as the multi-user state.
3	Extend multi-user mode by making local resources available over the network.
4	Is available to be defined as an alternative multi-user environment configuration. It is not necessary for system operation and is usually not used.
5	Shut the machine down so that it is safe to remove the power. Have the machine remove power, if possible.

6	Stop the operating system and reboot to the state defined by the `initdefault` entry in `/etc/inittab` .
a, b, c	Process only those `/etc/inittab` entries having the `a` , `b` , or `c` run level set. These are pseudo-states, which may be defined to run certain commands, but which do not cause the current run level to change.
Q, q	Re-examine `/etc/inittab` .
S, s	Enter single-user mode. This is the only run level that doesn't require the existence of a properly formatted `/etc/inittab` file. If this file does not exist, then by default, the only legal run level that `init` can enter is the single-user mode. When in single-user mode, the filesystems required for basic system operation will be mounted. When the system comes down to single-user mode, these file systems will remain mounted (even if provided by a remote file server), and any other local filesystems will also be left mounted. During the transition down to single-user mode, all processes started by `init` or `init.d` scripts that should only be running in multi-user mode are killed. In addition, any process that has a `utmp` entry will be killed. This last condition insures that all port monitors started by the SAC are killed and all services started by these port monitors, including `ttymon` login services, are killed.

FILES

`/etc/inittab`	Controls process dispatching by `init` .
`/var/adm/utmp`	Accounting information.
`/var/adm/wtmp`	History of all logins since file was last created.
`/etc/ioctl.syscon`	System console states.
`/dev/console`	System console device.
`/etc/default/init`	Environment variables.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

`init` requires privilege to run in the Trusted Solaris environment.

SEE ALSO

Trusted Solaris 7 Reference Manual

`login`(1) , `kill`(2) , `inittab`(4)

SunOS 5.7 Reference Manual

`sh`(1) , `stty`(1) , `who`(1) , `shutdown`(1M) , `su`(1M) , `ttymon`(1M) , `ioctl`(2) , `ctime`(3C) , `pam`(3) , `pam.conf`(4) , `utmp`(4) , `utmpx`(4) , `attributes`(5) , `pam_unix`(5) , `termio`(7I)

DIAGNOSTICS

If `init` finds that it is respawning an entry from `/etc/inittab` more than ten times in two minutes, assumes that there is an error in the command string in the entry, and generates an error message on the system console. It will then refuse to respawn this entry until either five minutes has elapsed or it receives a signal from a user-spawned `init` or `telinit` . This prevents `init` from eating up system resources when someone makes a typographical error in the `inittab` file, or a program is removed that is referenced in `/etc/inittab` .

NOTES

`init` and `telinit` can be run only by a privileged user.

The `S` or `s` state must not be used indiscriminately in `/etc/inittab` . When modifying this file, it is best to avoid adding this state to any line other than `initdefault` .

If a default state is not specified in the `initdefault` entry in `/etc/inittab` , state `6` is entered. Consequently, the system will loop by going to firmware and rebooting continuously.

If the `utmp` file cannot be created when booting the system, the system will boot to state " `s` " regardless of the state specified in the `initdefault` entry in `/etc/inittab` . This can occur if the `/var` file system is not accessible.

NAME

in.tftpd, tftpd – Internet Trivial File Transfer Protocol server

SYNOPSIS

in.tftpd [–s] [*homedir*]

DESCRIPTION

`tftpd` is a server that supports the Internet Trivial File Transfer Protocol (TFTP). This server is normally started by `inetd`(1M) and operates at the port indicated in the `tftp` Internet service description in the `/etc/inetd.conf` file. By default, the entry for `in.tftpd` in `etc/inetd.conf` is commented out. To make `in.tftpd` operational, the comment character(s) must be deleted from the file. See `inetd.conf`(4) .

Before responding to a request, the server attempts to change its current directory to *homedir* ; the default directory is `/tftpboot` .

The use of `tftp` does not require an account or password on the remote system. Due to the lack of authentication information, `in.tftpd` will allow only publicly readable files to be accessed. Files may be written only if they already exist and are publicly writable. Note that this extends the concept of “public” to include all users on all hosts that can be reached through the network; this may not be appropriate on all systems, and its implications should be considered before enabling this service.

`in.tftpd` runs with the user ID and group ID set to `[GU]ID_NOBODY` under the assumption that no files exist with that owner or group. However, nothing checks this assumption or enforces this restriction.

OPTIONS

–s Secure. When specified, the directory change to *homedir* must succeed. The daemon also changes its root directory to *homedir* .

SUMMARY OF TRUSTED SOLARIS CHANGES

`in.tftpd` should be started from the trusted path with a UID of `0` ; it must inherit the `proc_chroot` , `proc_owner` , and `proc_setid` privileges.

FILES

`/etc/inetd.conf` Configuration file for inetd.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`inetd`(1M)

SunOS 5.7 Reference Manual

`tftp`(1) , `netconfig`(4) , `attributes`(5)

Sollins, K.R., *The TFTP Protocol (Revision 2)* , RFC 783, Network Information Center, SRI International, Menlo Park, California, June 1981.

NAME

tnchkdb – Check file syntax of trusted network databases

SYNOPSIS

/usr/sbin/tnchkdb

/usr/sbin/tnchkdb –t [*pathname*]

/usr/sbin/tnchkdb –h [*pathname*]

/usr/sbin/tnchkdb –t [*t_pathname*] –h [*h_pathname*]

/usr/sbin/tnchkdb –i [*pathname*]

DESCRIPTION

tnchkdb checks the syntax of the tnrhtp(4), tnrhdb(4), or tnidb(4) databases at *pathname*. (*pathname* is the full pathname and filename of the file.) If no database is specified, all three databases in /etc/security/tsol are checked. tnchkdb returns an exit status of 0 (true) and no output if the file is syntactically and semantically correct. Otherwise, tnchkdb returns a nonzero (false) exit status and writes an error diagnostic to the standard output file. tnchkdb also examines the label and DAC information on the specified database files and reports mismatches as WARNINGS rather than ERRORS.

tnchkdb can be run at any sensitivity label that dominates the sensitivity label of the database file. This restriction can be overridden by the file_mac_read privilege.

OPTIONS

–t [*pathname*]
Check *pathname* for proper tnrhtp syntax. If the *pathname* is not specified, then check /etc/security/tsol/tnrhtp.

–h [*pathname*]
Check *pathname* for proper tnrhdb syntax. If the *pathname* is not specified, then check /etc/security/tsol/tnrhdb.

–t [*t_pathname*] –h [*h_pathname*]
Check *t_pathname* for proper tnrhtp syntax and check *h_pathname* for proper tnrhdb syntax. This option complains about template names assigned in tnrhdb but not defined in tnrhtp. If the pathname is not specified, then check /etc/security/tsol/tnrhtp for the –t option and /etc/security/tsol/tnrhdb for the –h option.

–i [*pathname*]
Check *pathname* for proper tnidb syntax. If the pathname is not specified, then check /etc/security/tsol/tnidb.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

FILES

`/etc/security/tsol/tnidb`
Trusted network interface-control database

`/etc/security/tsol/tnrhdb`
Trusted network remote-host database

`/etc/security/tsol/tnrhtp`
Trusted network remote-host templates

SEE ALSO

Trusted Solaris 7 Reference Manual

`tnd`(1M), `tnctl`(1M), `tnidb`(4), `tnrhdb`(4), `tnrhtp`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

It is possible to have inconsistent but valid configurations of `tnrhtp` and `tnrhdb`, since NIS+ may be used to supply missing templates.

NAME

tnctl – Configure Trusted Solaris network-daemon control parameters

SYNOPSIS

/usr/sbin/tnctl [–v] [–d *debug_level*] [–f *logfile*] [–p *poll-interval*] [–i *interface_name*] [–h *host_name*] [–t *template_name*] [–b *ip_address*] [–B *ip_address*]

/usr/sbin/tnctl –I *tnidb_path*

/usr/sbin/tnctl –T *tnrhtp_path*

/usr/sbin/tnctl –H *tnrhdb_path*

DESCRIPTION

`tnctl` provides an interface to send control and configuration messages either to the kernel directly or to `tnd`(1M).

If a local trusted-networking database file is modified, the administrator should issue `tnchkdb`(1M) to check the syntax, and must also issue `tnctl` to reload the kernel caches.

`tnctl` must be started from the trusted path; and for the –i, –t, –h, –b, –B, –I, –T, and –H options, it must have the `sys_net_config` privilege. `tnctl` can be run at any sensitivity label.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–v	Turn on verbose mode.
–d *debug_level*	Turn on debugging for `tnd` to the level specified by *debug_level*. *debug_level* may be `1` or `2`; however, currently no distinction is made between the two values. If `tnd` is not already using a logfile, use `/var/tsol/tndlog`.
–f *logfile*	Set the logfile path, *logfile*, to which `tnd` writes debugging information. If *logfile* already exists, debugging information is appended to *logfile*.
–p *poll-interval*	Set poll interval to *poll-interval* seconds. The valid range is 0 to 2147483647; a zero value turns polling off.
–i *interface_name*	Update the kernel-interface cache on the specified *interface_name*. If the entry does not exist in the database, return an error message.
–h *hostname*	Update the kernel remote-host cache on the specified *hostname*. If the entry does not exist in the database, delete the entry from the cache.

–t *template_name*	Update the kernel template cache on the specified *template_name*. If the entry does not exist in the database, return an error message. See WARNINGS about the risks of changing a template when the network is up.
–I *tnidb_path*	Load all entries in the *tnidb_path* file into the kernel cache. *tnidb_path* is the full pathname plus filename of the file.
–T *tnrhtp_path*	Load all entries in the file *tnrhtp_path* into the kernel cache. *tnrhtp_path* is the full pathname plus filename of the file.
–H *tnrhdb_path*	Load all entries in the *tnrhdb_path* file into the kernel cache. *tnrhdb_path* is the full pathname plus filename of the file.
–b *ip_address*	Add a remote broadcast address.
–B *ip_address*	Delete a remote broadcast address.

FILES

`/etc/security/tsol/tnidb`
Trusted network interface-control database

`/etc/security/tsol/tnrhdb`
Trusted network remote-host database

`/etc/security/tsol/tnrhtp`
Trusted network remote-host templates

`/etc/nsswitch.conf`
Configuration file for the name service switch

SEE ALSO

Trusted Solaris 7 Reference Manual

`tninfo`(1M), `tnd`(1M), `tnchkdb`(1M), `nsswitch.conf`(4), `tnidb`(4), `tnrhdb`(4), `tnrhtp`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

Currently, only level-1 debugging is supported.

WARNINGS

Changing a template while the network is up can change the security view of an undetermined number of hosts.

NAME

tnd – Trusted network daemon

SYNOPSIS

/usr/sbin/tnd [–d *debug_level*] [–f *logfile*] [–p *poll-interval*]

DESCRIPTION

The tnd (trusted network daemon) initializes the kernel with trusted network databases and also reloads the databases on demand. tnd is started at the beginning of the boot process if needed.

tnd loads these databases into the kernel: the remote host database, tnrhdb(4); the remote-host template database, tnrhtp(4); and the interface database, tnidb(4). These databases and their effect on the trusted network are described in their respective man pages. When tnrhdb(4) and tnrhtp(4) and the associated NIS+ tables are changed, tnd also updates the local kernel and file-system caches at the predetermined interval.

tnd logs its debugging information in a log file (by default, /var/tsol/tndlog) which is either set by using the –f option or changeable by using tnctl(1M).

If a local trusted networking database file is modified, the administrator should issue a tnchkdb(1M) to check the syntax, and must issue a tnctl to reload the kernel caches.

tnd must be started from the Trusted Path and inherit these privileges to run: net_privaddr, net_mac_read, net_downgrade_sl, sys_net_config, proc_setclr, proc_setsl. tnd is intended to be started from an rc script and to run at the ADMIN_LOW sensitivity label.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–d *debug_level* Turn on debugging to the level specified by *debug_level*. *debug_level* may be 1 or 2; however, currently no distinction is made between the two values. If log file is not specified with the –f option, use /var/tsol/tndlog.

–f *logfile* Set logfile path to *logfile* for writing debugging information. If *logfile* already exists, append debugging information to it.

–p *poll-interval* Set poll interval to *poll-interval* seconds. By default, *poll-interval* is 1800 seconds (30 minutes).

FILES

/etc/security/tsol/tnidb
Trusted network interface-control database

/etc/security/tsol/tnrhdb

Trusted network remote-host database

`/etc/security/tsol/tnrhtp`
Trusted network remote-host templates

`/var/tsol/tndlog`
Log of `tnd` debugging information

`/var/tsol/tnrhtp_c`
Cache of the trusted network remote-host database

`/var/tsol/tnrhdb_c`
Cache of trusted network remote-host templates

`/etc/nsswitch.conf`
Configuration file for the name service switch

SEE ALSO

Trusted Solaris 7 Reference Manual

`tnchkdb`(1M) `tninfo`(1M) `tnctl`(1M), `tnidb`(4) `tnrhdb`(4) `tnrhtp`(4) `tndlog`(4) `nsswitch.conf`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

tninfo – Print information and statistics about kernel-level network

SYNOPSIS

/usr/sbin/tninfo [–skc] [–i [*if_name*]] [–h [*hostname*]] [–t [*template_name*]]

DESCRIPTION

`tninfo` provides an interface to retrieve and display kernel-level network information and statistics.

`tninfo` is intended to be run at `ADMIN_HIGH` and effective user ID 0. These restrictions can be overridden by these privileges: `file_mac_read`, `sys_trans_label`, `file_dac_read`. The `tninfo` executable should be maintained with a sensitivity label of `ADMIN_LOW`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–s	Print the default security structures associated with each socket or stream.
–k	Print the network statistics. This is the default option.
–c	Print the cache statistics.
–i [*if_name*]	Display the security structure for the specified interface in the kernel cache. The output should reflect what is specified in the `tnidb` database. If *if_name* is not specified, display the entire interface cache.
–h [*hostname*]	Display the security structure for the specified host in the kernel remote-host cache. The output should reflect what is specified in the `tnrhdb` and `tnrhtp` databases. If *hostname* is not specified, display the entire remote-host cache.
–t [*template_name*]	Display the structure associated with the specified *template_name.* The output should reflect what is specified in the `tnrhtp` database. If *template_name* is not specified, display the entire remote-host template cache. If a field within an entry is not specified (for example, `def_uid=empty;`), then that field will not be displayed.

FILES

`/etc/security/tsol/tnidb`	Trusted network interface-control database
`/etc/security/tsol/tnrhdb`	Trusted network remote-host database
`/etc/security/tsol/tnrhtp`	Trusted network remote-host templates

SEE ALSO

Trusted Solaris 7 Reference Manual

`tnd`(1M), `tnctl`(1M), `tnidb`(4), `tnrhdb`(4), `tnrhtp`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

The kernel's tables can change while `tninfo` is examining them; the result is incorrect or partial displays.

NAME

tokmapctl – Configure token-mapping daemon

SYNOPSIS

tokmapctl [–H *hostname*] [–P *satmp_port*] [–s *timeout*] [–r *retries*] [–R *retry_interval*] [–I [*cachesize*]] [–F [*hostname*]] [–m *meter_type*] [–d *level*] [–D *level*] [–l *logfile*] [–M *hostname*] [–x]

DESCRIPTION

`tokmapctl` provides an interface to send control and configuration requests to a `tokmapd` process.

`tokmapctl` must be started from the trusted path and must inherit the `net_privaddr` and `net_mac_read` privileges. `tokmapctl` should be run at sensitivity label `ADMIN_HIGH`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–H *hostname* — Send the control and configuration requests to the `tokmapd` process on host *hostname*. If this option is not specified, the request is sent to the `tokmapd` process on the local host.

–P *port* — Send the requests to `tokmapd` on port number *port*. This option is intended for debugging only. If this option is not specified, requests are sent to port 90.

–s *timeout* — Tell `tokmapd` to use *timeout* seconds as its timeout period before retrying a request that has been sent to another token-mapping server but has received no reply. The default is 5 seconds.

–r *retries* — Tell `tokmapd` to use *retries* as the maximum number of times to retry requests to other token-mapping servers. The default is 5 retries.

–R *retry_interval* — Tell `tokmapd` to use *retry_interval* milliseconds as its interval between checks for the need to retry requests to other token-mapping servers. The default interval is 100 milliseconds.

–I [*cachesize*] — Tell `tokmapd` to reinitialize its token store. If it is specified, *cachesize* is used to set the size of the token store in-memory cache. *cachesize* specifies how many entries of each attribute type to keep in the cache. The default is 10.

–F [*hostname*]	Tell `tokmapd` to flush all tokens for *hostname* from its token store. If *hostname* is omitted, `tokmapd` flushes all tokens for remote hosts.
–m *meter_type*	Fetch and display metering data from `tokmapd`. The allowable values for *meter_type* are `hostlist`, `general`, `store`, and `all`. Multiple –m options may be specified to request multiple types of metering data; specify type `all` to fetch and display all the meter types.
–d *level*	Set `tokmapd` debugging level to *level*. Debugging level 1 produces minimal output showing when messages are sent and received. Level 3 shows the contents of the headers of messages. Level 5 shows detailed information including buffer addresses and contents. Levels above 5 show additional internal information.
–D *level*	Set `tokmapctl` debugging level to *level*. Debugging level 1 produces minimal output showing when messages are sent and received. Level 3 shows the contents of the headers of messages. Level 5 shows detailed information including buffer addresses and contents. Levels above 5 show additional internal information.
–l *logfile*	Tell `tokmapd` to write its debugging output to *logfile*.
–M *hostname*	Fetch and display metering data from `tokmapd` for its token-mapping exchanges with host *hostname*.
–x	Send a request for an orderly shutdown and exit to `tokmapd`.

SEE ALSO

Trusted Solaris 7 Reference Manual

`tokmapd`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

If the token store becomes too large, use the –`I` option of `tokmapctl` to make `tokmapd` delete the current token store and reinitialize.

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

tokmapd – Token-mapping daemon

SYNOPSIS

/usr/sbin/tokmapd [–d *level*] [–l *logfile*] [–c *cachesize*] [–P *satmp_port*] [–p *kernel_port*] [–s *timeout*] [–r *retries*] [–R *retry_interval*] [–f *path*]

DESCRIPTION

`tokmapd` implements the SATMP token-mapping protocol to support the labeling of information transferred over the trusted network. The information is labeled using tokens that represent attribute values. `tokmapd` is responsible for mapping tokens to attribute values and vice versa. `tokmapd` accepts token-mapping requests from the kernel and from token-mapping servers on other hosts.

`tokmapd` must be started from the trusted path and must inherit the `net_privaddr`, `proc_setclr`, and `proc_setsl` privileges. `tokmapd` should be run at sensitivity label `ADMIN_HIGH`.

If `tokmapd` is stopped and its on-disk cache reinitialized or removed, the machine should be rebooted.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

–d *level* Set `tokmapd` debugging level to *level*. Debugging level 1 produces minimal output showing when messages are sent and received. Level 3 shows the contents of the headers of messages. Level 5 shows detailed information including buffer addresses and contents. Levels above 5 show additional internal information.

–l *logfile* Write any debugging output to *logfile*. If *logfile* already exists, the debugging output is appended to it. If this option is not specified, the default logfile `/var/tsol/tokmapdlog` is used.

–c *cachesize* Set the size of the token store in-memory cache to *cachesize*. *cachesize* specifies how many entries of each attribute type to keep in the cache. The default is 10.

–P *satmp_port* Listen on *satmp_port* for SATMP and `tokmapctl` requests. This option is intended for debugging only. If this option is not specified, port 90 is used.

–p *kernel_port* Listen on *kernel_port* for token-mapping requests from the kernel. This option is intended for debugging only. If this option is not specified, port 10800 is used.

–s *timeout*	Use *timeout* seconds as the timeout period before retrying a request that has been sent to another token-mapping server but has received no reply. If this option is not specified, a timeout interval of 5 seconds is used.
–r *retries*	Resend requests to other token-mapping servers a maximum of *retries* times. If this option is not specified, a retry limit of 5 is used.
–R *retry_interval*	Use *retry_interval* milliseconds as the interval between checks for the need to do retries. The default interval is 100 milliseconds.
–f *path*	Place the token store and host-list files in the *path* directory. If this option is not specified, the files are stored in `/etc/security/tsol`.

FILES

`/etc/security/tsol/tokendb.pag`	Token store file
`/etc/security/tsol/tokendb.dir`	Token store file
`/etc/security/tsol/tokendb.ir`	Token store file
`/etc/security/tsol/tokendb.hosts`	Token store file
`/var/tsol/tokmapdlog`	Logfile of debugging output

SEE ALSO

Trusted Solaris 7 Reference Manual

`tokmapctl`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

The token store is checked for consistency each time `tokmapd` is started. If the token store was not properly flushed to disk at the last shutdown, or if other inconsistencies are found, the token-store contents are deleted and the token store is reinitialized.

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

NAME

traceroute – Print the route packets take to network host

SYNOPSIS

traceroute [–dFInvx] [–f *first_ttl*] [–g *gateway* [–g *gateway*…] –r][–i *iface*] [–m *max_ttl*] [–p *port*] [–q *nqueries*] [–s *src_addr*] [–t *tos*] [–w *waittime*] host [*packetlen*]

DESCRIPTION

The Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route a packet follows can be difficult. The utility `traceroute` traces the route that an IP packet follows to another internet host. In the Trusted Solaris environment, `traceroute` must be run with the `net_rawaccess` privilege.

`traceroute` utilizes the IP protocol `ttl` (time to live) field and attempts to elicit an `ICMP TIME_EXCEEDED` response from each *gateway* along the path, and a `PORT_UNREACHABLE`(or `ECHO_REPLY` if –`I` is used) response from the destination host. It starts by sending probes with a `ttl` of `1` and increases by `1` until it either gets to the host, or it hits the maximum `ttl`. The default maximum `ttl` is 30 hops, but this can be set by the –`m` option.

Three probes are sent at each `ttl` setting, and a line is printed showing the `ttl`, the hostname and the address of the gateway, and the `rtt` (round trip time) of each probe. The number of probes may be specifically set using the –`q` option. If the probe answers come from different gateways, the hostname and the address of each responding system will be printed. If there is no response within a 5 second timeout interval, a "`*`" is printed for that probe. The –`w` option may be used to set the timeout interval. Other possible annotations that may appear after the time are:

`!` the *ttl* value in the received packet is `<= 1`.

`!H` host unreachable.

`!N` network unreachable.

`!P` protocol unreachable.

`!S` source route failed. This should never occur. If this is seen, the associated gateway is broken.

`!F` fragmentation needed. This should never occur. If this is seen, the associated gateway is broken.

`!X` communication administratively prohibited.

`<!N>` ICMP unreachable code N.

If almost all the probes result in some kind of unreachable code, then `traceroute` gives up and exits.

The destination *host* is not supposed to process the UDP probe packets, so the destination *port* default is set to an unlikely value. However, if some application on the destination is using that value, the value of *port* can be changed with the –p option.

The only mandatory parameter is the destination *host* name or IP number. The default probe datagram length is 40 bytes, but this may be increased by specifying a packet length (in bytes) after the destination *host* name.

All numeric arguments to `traceroute` can be specified in either decimal or hexadecimal notation. For example, *packetlen* can be specified either as `256` or `0x100`.

OPTIONS

–d	Set the `SO_DEBUG` socket option.
–F	Set the "don't fragment" bit.
–f *first_ttl*	Set the starting `ttl` value to *first_ttl*, to override the default value `1`. `traceroute` skips processing for those intermediate gateways which are less than *first_ttl* hops away.
–g *gateway*	Specify a loose source route *gateway*. The user can specify more than one *gateway* by using –g for each gateway. The maximum that can be set is `8`.
–I	Use ICMP `ECHO` instead of UDP datagrams.
–i *iface*	Specify a network interface to obtain the source IP address for outgoing probe packets. This is normally only useful on a multi-homed host. The –s option is also another way to do this. Note that this option does not provide a way to specify the interface on which the probe packets are sent.
–m *max_ttl*	Set the maximum `ttl` used in outgoing probe packets. The default is 30 hops, which is the same default used for TCP connections.
–n	Print hop addresses numerically rather than symbolically and numerically. This saves a nameserver address-to-name lookup for each gateway found on the path.
–p *port*	Set the base UDP *port* number used in probes.The default is 33434. `traceroute` hopes that nothing is listening on UDP *ports* `(base+(nhops-1)*nqueries)` to `(base+(nhops*nqueries)-1)`at the destination host, so that an `ICMP PORT_UNREACHABLE` message will be returned to terminate the route tracing. If something is listening on a *port* in the default range, this option can be used to select an

```
istanbul% traceroute -g cairo -g paris -g
  london sanfrancisco
```

unused *port* range.`nhops` is defined as the number of hops between the source and the destination.

`–q` *nqueries* Set the desired number of probe queries. The default is 3.

`–r` Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly-attached network, an error is returned. This option can be used to send probes to a local host through an interface that has been dropped by the router deamon. See `in.routed`(1M).

`–s` *src_addr* Use the following address, which usually is given as an IP address, not a hostname, as the source address in outgoing probe packets. On multi-homed hosts, those with more than one IP address, this option can be used to force the source address to be something other than the IP address traceroute picks by default. If the IP address is not one of this machine's interface addresses, an error is returned and nothing is sent. When used together with the `–i` option, the given IP address should be configured on the specified interface. Otherwise, an error will be returned.

`–t` *tos* Set the *tos*(type-of-service) in probe packets to the specified value. The default is zero. The value must be an integer in the range from 0 to 255. Gateways along the path may route the probe packet differently depending upon the *tos* value set in the probe packet.

`–v` Verbose output. For each hop, the size and the destination of the response packets is displayed. Also ICMP packets received other than `TIME_EXCEEDED` and `UNREACHABLE` are listed as well.

`–w` *waittime* Set the time, in seconds, to wait for a response to a probe. The default is five (5) seconds.

`–x` Prevent `traceroute` from calculating checksums. Note that checksums are usually required for the last hop when using ICMP `ECHO` probes. See the `–I` option.

OPERANDS

The following operands are supported:

host The network host.

EXAMPLES

EXAMPLE 1 Using the `traceroute` Utility

The following is a sample `traceroute` run and its output. It shows the 7-hop path that a packet would follow from the host istanbul to the host sanfrancisco.

```
istanbul% traceroute sanfrancisco
traceroute: Warning: Multiple interfaces found; using 172.31.86.247 @ le0
traceroute to sanfrancisco (172.29.64.39), 30 hops max, 40 byte packets
1  frbldg7c-86 (172.31.86.1)  1.516 ms  1.283 ms  1.362 ms
2  bldg1a-001 (172.31.1.211)  2.277 ms  1.773 ms  2.186 ms
3  bldg4-bldg1 (172.30.4.42)  1.978 ms  1.986 ms  13.996 ms
4  bldg6-bldg4 (172.30.4.49)  2.655 ms  3.042 ms  2.344 ms
5  ferbldg11a-001 (172.29.1.236)  2.636 ms  3.432 ms  3.830 ms
6  frbldg12b-153 (172.29.153.72)  3.452 ms  3.146 ms  2.962 ms
7  sanfrancisco (172.29.64.39)  3.430 ms  3.312 ms  3.451 ms
```

EXAMPLE 2 Using the `traceroute` Utility With Source Routing

The following example shows the path of a packet that goes from istanbul to sanfrancisco through the hosts cairo and paris, as specified by the –`g` option. The –`I` option makes `traceroute` send ICMP `ECHO` probes to the host sanfrancisco. The –`i` option sets the source address to the IP address configured on the interface `qe0`.

```
istanbul% traceroute -g cairo -g paris -i qe0 -q 1 -I sanfrancisco

traceroute to sanfrancisco (172.29.64.39), 30 hops max, 56 byte packets
1  frbldg7c-86 (172.31.86.1)  2.012 ms
2  flrbldg7u (172.31.17.131)  4.960 ms
3  cairo (192.168.163.175) 4.894 ms
4  flrbldg7u (172.31.17.131)  3.475 ms
5  frbldg7c-017 (172.31.17.83)  4.126 ms
6  paris (172.31.86.31) 4.086 ms
7  frbldg7b-82 (172.31.82.1)  6.454 ms
8  bldg1a-001 (172.31.1.211)  6.541 ms
9  bldg6-bldg4 (172.30.4.49)  6.518 ms
10  ferbldg11a-001 (172.29.1.236)  9.108 ms
11  frbldg12b-153 (172.29.153.72)  9.634 ms
12  sanfrancisco (172.29.64.39)  14.631 ms
```

EXIT STATUS

The following exit values are returned:

`0` Successful operation.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The `traceroute` utility must be run with the `net_rawaccess` privilege.

An administrative role must run this utility. By default, `traceroute` is in the Network Management profile.

SEE ALSO

Trusted Solaris 7 Reference Manual

`netstat`(1M)

SunOS 5.7 Reference Manual

`ping`(1M), `attributes`(5)

`attributes`(5)

WARNINGS

This utility is intended for use in network testing, measurement and management. It should be used primarily for manual fault isolation. Because of the load it could impose on the network, it is unwise to use `traceroute` during normal operations or from automated scripts.

NAME

uadmin – Administrative control

SYNOPSIS

/sbin/uadmin *cmd fcn*

DESCRIPTION

The `uadmin` command provides control for basic administrative functions. This command is tightly coupled to the System Administration procedures and is not intended for general use.

Both *cmd* (command) and *fcn* (function) are converted to integers and passed to the `uadmin` system call.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SUMMARY OF TRUSTED SOLARIS CHANGES

The privileges needed for this command to succeed depend on *cmd* and *fcn*. For `A_SHUTDOWN`, `A_REBOOT`, and `A_FREEZE`, the necessary privilege is `sys_boot`. For `A_REMOUNT`, `A_SWAPCTL ADD`, and `A_SWAPCTL REMOVE`, the necessary privilege is `sys_mount`.

SEE ALSO

Trusted Solaris 7 Reference Manual

`uadmin`(2)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

mount, umount – Mount or unmount file systems and remote resources

SYNOPSIS

mount [–p | –v]

mount [–F *FSType*] [*generic_options*] [–o *specific_options*] [–0] *special* | *mount_point*

mount [–F *FSType*] [*generic_options*] [–o *specific_options*] [–0] [–S *attribute_list*] *special mount_point*

mount –a [–F *FSType*] [–V] [*current_options*] [–o *specific_options*] [–S *attribute_list*] [*mount_point*…]

umount [–V] [–o *specific_options*] *special* | *mount_point*

umount –a [–V] [–o *specific_options*] [*mount_point*…]

DESCRIPTION

`mount` attaches a file system to the file system hierarchy at the *mount_point* , which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

`umount` unmounts a currently mounted file system, which may be specified either as a *mount_point* or as *special* , the device on which the file system resides.

`mount` and `umount` maintain a table of mounted file systems in `/etc/mnttab` , which is described in `mnttab`(4) . `mount` adds an entry to the mount table; `umount` removes an entry from the table.

When invoked with both the *special* and *mount_point* arguments and the –F option, `mount` validates all arguments except for *special* and invokes the appropriate *FSType* -specific `mount` module. If invoked with no arguments, `mount` lists all the mounted file systems recorded in the mount table, `/etc/mnttab` . If invoked with a partial argument list (with only one of *special* or *mount_point* , or with both *special* or *mount_point* specified but not *FSType*), `mount` will search `/etc/vfstab` for an entry that will supply the missing arguments. If no entry is found, and the special argument starts with "/", the default local file system type specified in `/etc/default/fs` will be used. Otherwise the default remote file system type will be used. The default remote file system type is determined by the first entry in the `/etc/dfs/fstypes` file. After filling in missing arguments, `mount` will invoke the *FSType* -specific `mount` module.

The –S option can be used to assign any or all of the following mount-time security attributes to the named file system when appropriate: an ACL , a mode, a user ID , a group ID , a sensitivity label, forced privilege(s), allowed privilege(s), a file attribute flag, a filesystem label range, or an MLD prefix. If the –S option is not used, `mount` also searches `/etc/security/tsol/vfstab_adjunct` for any security attributes that may be specified there for the file system being

mounted. Specifying mount-time attributes is useful only when mounting file systems that do not support the attributes.

Mount-time security attributes should be specified for file systems whose objects do not have any attributes, such as user and group ID s, and for file systems whose objects do not support the Trusted Solaris extended security attributes, such as sensitivity labels. When a required attribute is not specified at mount-time, a default value is applied. The defaults are described in the `OPTIONS` section, where the keywords are defined for the `–S` option.

File system types `UFS` , `TMPFS` , and `NFS` (from a Trusted Solaris server) have a full set of Trusted Solaris extended security attributes already defined. (See the `getfsattr`(1M) man page for how to get attributes on mounted file systems). Because the attributes can be changed on these file systems *after* they are mounted, they are called *variable* file systems. For example, the sensitivity label on a file in a variable file system can be changed by an authorized user. The security attributes on a variable file system can be overridden at mount time, but individual objects in the file system retain any attributes that were originally set on the objects.

File systems that do not support the Trusted Solaris extended security attributes are called *fixed* because any attributes assigned to them (either at mount time or by default) cannot be changed. For example, the sensitivity label specified at mount time for a fixed-attribute file system cannot be changed on any of the objects in that file system. An object that is moved or copied from the fixed file system to a variable file system can be changed after the move.

Mount-time security attributes override existing security attributes on a file system. However, mount-time attributes never override security attributes on the files and directories within the file system.

Without privilege, `mount` can be used to list mounted file systems and resources. To be able to mount and unmount, the `mount` command must have the `sys_mount` privilege and must run with an effective UID of `0` . The `umount` command must have the `sys_mount` privilege. Mandatory and discretionary read access is required both to the mount point and to the device being mounted; otherwise, MAC or DAC override privileges are required as described in `Intro`(2) . To succeed in all cases, `mount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` . To succeed in all cases, `umount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , and `file_dac_search` .

OPTIONS

`–F` *FSType*

Used to specify the *FSType* on which to operate. The *FSType* must be specified or must be determinable from `/etc/vfstab` , or by consulting `/etc/default/fs` or `/etc/dfs/fstypes` .

–a [*mount_points* . . .]
Perform `mount` or `umount` operations in parallel, when possible.

If mount points are not specified, `mount` will mount all file systems whose `/etc/vfstab` "mount at boot" field is "yes". If mount points are specified, then `/etc/vfstab` "mount at boot" field will be ignored.

If mount points are specified, `umount` will only unmount those mount points. If none is specified, then `umount` will attempt to unmount all filesystems in `/etc/mnttab` , with the exception of certain system required file systems: `/` , `/usr` , `/var` , `/proc` , `/dev/fd` , and `/tmp` .

–p
Print the list of mounted file systems in the `/etc/vfstab` format. Must be the only option specified.

–v
Print the list of mounted file systems in verbose format. Must be the only option specified.

–V
Echo the complete command line, but do not execute the command. `umount` generates a command line by using the options and arguments provided by the user and adding to them information derived from `/etc/mnttab` . This option should be used to verify and validate the command line.

generic_options
Options that are commonly supported by most *FSType* -specific command modules. The following options are available:

–m
Mount the file system without making an entry in `/etc/mnttab` .

–g
Globally mount the file system. On a clustered system, this globally mounts the file system on all nodes of the cluster. On a non-clustered system this has no effect.

–o
Specify *FSType* -specific options in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the *FSType* -specific module of the command. (See `mount_ufs`(1M))

–O

Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error “device busy”.

–r

Mount the file system read-only.

–S *attribute_list*

Specify in *attribute_list* a quoted semicolon-separated list of security attributes to associate with the file-system mount. Each attribute is specified with a value assigned to a keyword in semicolon-separated fields. All keywords are optional and follow the format:

keyword=value

where *keyword* is one of the following:

`acc_acl`	Sets the same ACL on all files or directories in the file system. See `aclfromtext`(3) for the format.
`mode`	Sets a DAC permission mode for each object in the file system. The only supported mode is the absolute mode, which is specified using octal numbers. See the description for the absolute-mode parameter on the `chmod`(1) man page. (Because the mode is an object-level attribute that has precedence over any mount-time attributes, setting a mode is only useful in the rare case when the type of file system being mounted does not support permission bits. In such cases, it is recommended that an explicit value be specified for the mode.)
`attr_flg`	Sets an attribute flag on all files in the file system. The only supported `attr_flag` value is `public` , whose effect is that when certain read operations are performed on any object in the file system on which this flag is set, audit records are not generated even when the operations are part of a preselected audit class, with the following exception. If the audit pseudo event for use of privilege (`AUE_UPRIV`) is included in a preselected audit class and if the operation involves the use of privilege), then an audit record is always generated. With the previous exception, the read operations for which audit records are

not generated when the public flag is set are: access(2) , fgetcmwlabel(2) , fgetsldname(2) , fstatvfs(2) , getcmwfsrange(2) , getcmwlabel(2) , getfpriv(2) , getmldadorn(2) , getsldname(2) , lgetcmwlabel(2) , lstat(2) , open(2) —read only, pathconf(2) , preadl(2) , readl(2) , readlink(2) , stat(2) , statvfs(2) , mldlstat(3) , and mldstat(3) . See *Trusted Solaris Audit Administration* and *Trusted Solaris Administrator's Procedures* for more details.

`gid` Sets the group ID for all objects in the file system. (Because the GID is an object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have GID s on its files or directories. In such cases, it is recommended that an explicit value be specified for the GID .)

`uid` Sets the user ID for all objects in the file system. (Because the UID is an object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have UID s on its files or directories. In such cases, it is recommended that an explicit value be specified for the UID .)

`slabel` Sets the sensitivity label for all objects in the file system. Specify the sensitivity label in hexadecimal or text format.

`forced` Specify one or more forced privileges for all executable files in the file system. Specify symbolic privilege name(s) in a comma-separated list (such as: `forced=file_audit, file_chown;`) or use `all` to indicate all privileges. Using `none` or omitting the keyword results in no forced privileges being applied. See `priv_desc`(4) . Any forced privileges must be a subset of the allowed privileges.

`allowed` Specify one or more allowed privilege(s) for all executable files in the file system. Specify symbolic privilege names in a comma-separated list (such as: `allowed=file_audit, file_chown;`) or use `all` to indicate all privileges. Using `none` or omitting the keyword results in no allowed privileges being applied.

See priv_desc(4) for names of privileges. Any allowed privilege(s) must be a superset of the forced privileges.

`low_range` Specify the lower bound of the file system label range as a sensitivity label in text format.

`hi_range` Specify the upper bound of the file system label range as a sensitivity label in text format.

`mld_prefix` Set a prefix to be used in the adorned names of multilevel directories. (See `multilevel directories` in the `DEFINITIONS` in `Intro`(2) for more about the MLD prefix.) Specify the value in text format (such as: `.MLD.` or `.hidden.`). On unlabeled (fixed attribute) file systems, the prefix generally has no useful effect—with the exception that an `mld_prefix` should be supplied if a variable filesystem is being mounted on the unlabeled filesystem and the root of the variable filesystem is an MLD .

Any of the above keywords may be omitted.

Note - Note: The semicolon separators between keyword / value pairs and any brackets used to specify sensitivity labels must be commented out so that the separators and brackets can be interpreted properly by the shell.

When a keyword appears without an attribute value or when a keyword is missing, a default value is assigned to that attribute. The default values for fixed attribute file systems are:

`acc_acl` None

`mode` The mode should always be explictly set for file systems that do not support file access modes, such as MS-DOS (`pcfs` type) file systems.

`attr_flag` None

`gid` The GID should always be explictly set for file systems that do not support group ID s, such as MS-DOS (`pcfs` type) file systems.

`uid` The UID should always be explictly set for file systems that do not support user ID s, such as MS-DOS (`pcfs` type) file systems.

`slabel`	The default sensitivity label of a fixed file system being mounted from a local device (such as a hard disk, floppy, or CD-ROM) is the sensitivity label of the device. For an allocated device, the file system is assigned the sensitivity label at which the device was allocated.
`forced`	None
`allowed`	None
`low_range`	`ADMIN_LOW`
`hi_range`	`ADMIN_HIGH`
`mld_prefix`	None

For example, the assignment of `forced=;` results in the default of "none" being applied.

USAGE

See `largefile`(5) for the description of the behavior of `mount` and `umount` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

SUMMARY OF TRUSTED SOLARIS CHANGES

Trusted Solaris security policy applies when mounting and unmounting file systems.

Mount-time security attributes may be specified either by using `mount` with the –`S` option on the command line or by specifying the attributes in the `vfstab_adjunct` file. Mount-time security attributes override existing security attributes on a file system. However, they never override security attributes on the files and directories within the file system. When access-control decisions are made, security attributes on a file or directory take precedence over security attributes specified either at the filesystem level or at mount time.

Except when merely listing mounted file systems and resources, `mount` must run with an effective UID of `0` and with the `sys_mount` privilege. `umount` also must run with an effective UID of `0` and with the `sys_mount` privilege. To succeed in all cases, `mount` needs: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` .

Note - Information labels (IL s) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any IL s on communications and files from systems running earlier releases as `ADMIN_LOW` .

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]` ; however, the IL component is fixed at `ADMIN_LOW` .

As a result, Trusted Solaris 7 has the following characteristics:

- IL s do not display in window labels; SL s (Sensitivity Labels) display alone within brackets.
- IL s do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW` .
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting IL s are always `ADMIN_LOW` , and cannot be set on any objects.
- Options related to information labels in the `label_encodings`(4) file can be ignored:

```
Markings Name= Marks;
Float Process Information Label;
```

FILES

`/etc/mnttab`	Mount table
`/etc/default/fs`	Default local file system type. Default values can be set for the following flags in `/etc/default/fs` . For example: `LOCAL=ufs` Specifies that `LOCAL:` is the default partition for a command if no *FSType* is specified.
`/etc/vfstab`	List of default parameters for each file system.
`/etc/security/tsol/vfstab.adjunct`	Mount-time attributes for file systems.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`getfsattr`(1M) , `getmldadorn`(1) , `mount_hsfs`(1M) , `mount_nfs`(1M) , `mount_pcfs`(1M) , `mount_tmpfs`(1M) , `mount_ufs`(1M) , `mountall`(1M) , `setfsattr`(1M) , `setmnt`(1M) , `mnttab`(4) , `priv_desc`(4) , `vfstab`(4) , `vfstab_adjunct`(4)

Trusted Solaris Administrator's Procedures

SunOS 5.7 Reference Manual

`setfacl`(1) , `mount_cachefs`(1M) , `default_fs`(4) , `attributes`(5)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mountall, umountall – Mount, unmount multiple file systems

SYNOPSIS

mountall [–F *FSType*] [–l | –r][*file_system_table*]

umountall [–k] [–s] [–F *FSType*] [–l | –r]

umountall [–k] [–s] [–h *host*]

DESCRIPTION

`mountall` is used to mount file systems specified in a file system table. The file system table must be in `vfstab`(4) format. If no *file_system_table* is specified, `/etc/vfstab` will be used. If '–' is specified as *file_system_table* , `mountall` will read the file system table from the standard input. `mountall` only mounts those file systems with the `mount at boot` field set to `yes` in the *file_system_table* .

Each file system which has an `fsckdev` entry specified in the file system table will be checked using `fsck`(1M) in order to determine if it may be safely mounted. If the file system does not appear mountable, it is fixed using `fsck` before the mount is attempted. File systems with a '–' entry in the `fsckdev` field will be mounted without first being checked.

`umountall` causes all mounted file systems except `root` , `/proc` , `/var` , and `/usr` to be unmounted. If the *FSType* is specified, `mountall` and `umountall` limit their actions to the *FSType* specified. There is no guarantee that `umountall` will unmount *busy* filesystems, even if the –k option is specified.

`mountall` and `umountall` must run with an effective UID of `0` and with the `sys_mount` privilege.

Mandatory and discretionary read access are required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in `Intro`(2) . To succeed in all cases, the `mountall` and `umountall` commands need the privileges: `file_mac_read` , `file_dac_read` , `file_mac_write` , `file_dac_write` , `file_mac_search` , `file_dac_search` , `net_privaddr` , `proc_setsl` , `proc_setil` , `sys_mount` , and `sys_trans_label` .

OPTIONS

–F Specify the *FSType* of the file system to be mounted or unmounted.

–h *host* Unmount all file systems listed in `/etc/mnttab` that are remote-mounted from host.

–k Use the `fuser` –k *mount-point* command. See the `fuser`(1M) for details. The –k option sends the `SIGKILL` signal to each process using the file. As this option spawns kills for each process, the kill messages may not show up immediately. There is no guarantee that `umountall` will unmount *busy* filesystems, even if the –k option is specified.

–l Limit the action to local file systems.

–r Limit the action to remote file system types.

–s Do not perform the umount operation in parallel.

SUMMARY OF TRUSTED SOLARIS CHANGES

Trusted Solaris security policy applies when mounting and unmounting file systems.

mountall and umountall must run with an effective UID of 0 and with the sys_mount privilege.

Mandatory and discretionary read access are required to both the mount point and the device being mounted; to override MAC and DAC restrictions requires privilege as described in Intro(2) . To succeed in all cases, the mountall and umountall commands need the privileges: file_mac_read , file_dac_read , file_mac_write , file_dac_write , file_mac_search , file_dac_search , net_privaddr , proc_setsl , proc_setil , sys_mount , and sys_trans_label .

Mount-time security attributes may be specified in the vfstab_adjunct file.

FILES

/etc/mnttab mounted file system table

/etc/vfstab table of file system defaults

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

mount(1M) , mnttab(4) , vfstab(4) , vfstab_adjunct(4)

SunOS 5.7 Reference Manual

fsck(1M) , attributes(5)

DIAGNOSTICS

No messages are printed if the file systems are mountable and clean.

Error and warning messages come from fsck(1M) and mount(1M) .

NAME

unshare – Make local resource unavailable for mounting by remote systems

SYNOPSIS

unshare [–F *FSType*] [–o *specific_options*] [*pathname* | *resourcename*]

DESCRIPTION

The `unshare` command makes a shared local resource unavailable as file system type *FSType.* If the option –F *FSType* is omitted, then the first file system type listed in file `/etc/dfs/fstypes` will be used as the default. *Specific_options,* as well as the semantics of *resourcename,* are specific to particular distributed file systems.

OPTIONS

–F *FSType*	Specify the file system type.
–o *specific_options*	Specify options specific to the file system provided by the –F option.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `unshare` command must be run with an effective UID of `0`. If the file being unshared is of the type NFS, this command must have the `sys_nfs` privilege to succeed. If this command has the `file_mac_write` privilege, it can be run any sensitivity label other than `ADMIN_LOW`. To succeed in all cases, this command needs the `file_mac_read` and `file_mac_search` privileges.

FILES

`/etc/dfs/fstypes`	List of system types, NFS by default.
`/etc/dfs/sharetab`	System record of shared file systems.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`share`(1M), `shareall`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

If *pathname* or *resourcename* is not found in the shared information, an error message will be sent to standard error.

NAME

shareall, unshareall – Share, unshare multiple resources

SYNOPSIS

shareall [–F *FSType* [, *FSType* …]] [–| *file*]

unshareall [–F *FSType* [, *FSType* …]]

DESCRIPTION

When used with no arguments, `shareall` shares all resources from *file* , which contains a list of `share` command lines. If the operand is a hyphen (–), then the `share` command lines are obtained from the standard input. Otherwise, if neither a *file* nor a hyphen is specified, then the file `/etc/dfs/dfstab` is used as the default.

Resources may be shared by specific file system types by specifying the file systems in a comma-separated list as an argument to –F .

`unshareall` unshares all currently shared resources. Without a –F flag, it unshares resources for all distributed file system types.

OPTIONS

–F *FSType*	Specify file system type. Defaults to the first entry in `/etc/dfs/fstypes` .

SUMMARY OF TRUSTED SOLARIS CHANGES

The `shareall` and `unshareall` commands must be run with an effective UID of `0` . If any file being shared or unshared is of the type NFS , then the command requires the `sys_nfs` privilege [see `share_nfs`(1M)]. If the file `/etc/dfs/sharetab` does not exist, the `shareall` command will create the file; thus, the `shareall` command must be run at the sensitivity level of `ADMIN_LOW` . If the file `/etc/dfs/sharetab` exists, then the `shareall` and `unshareall` commands can be run at any other sensitivity level if they have the `file_mac_write` privilege. To succeed in all cases, the commands need the `file_mac_read` and `file_mac_search` privileges.

FILES

`/etc/dfs/dfstab`	List of `share` commands to be executed at boot time.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`share`(1M) , `unshare`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

unshare_nfs – Make local NFS file systems unavailable for mounting by remote systems

SYNOPSIS

unshare [–F nfs] *pathname*

DESCRIPTION

The `unshare` command makes local file systems unavailable for mounting by remote systems. The shared file system must correspond to a line with NFS as the *FSType* in the file `/etc/dfs/sharetab`.

OPTIONS

The following options are supported:

–F This option may be omitted if NFS is the first file system type listed in the file `/etc/dfs/fstypes`.

SUMMARY OF TRUSTED SOLARIS CHANGES

The `sys_nfs` privilege is required to run this command, which must be run as UID `0` at label `ADMIN_LOW`[`ADMIN_LOW`].

FILES

`/etc/dfs/fstypes` List of system types, NFS by default.

`/etc/dfs/sharetab` System record of shared file systems.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

Trusted Solaris 7 Reference Manual

`share`(1M)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

If the file system being unshared is a symbolic link to a valid pathname, the canonical path (the path which the symbolic link follows) will be unshared.

For example, if `/export/foo` is a symbolic link to `/export/bar` (`/export/foo -> /export/bar`), the following `unshare` command will result in `/export/bar` as the unshared pathname (and not `/export/foo`).

```
example# unshare –F nfs /export/foo
```

NAME

updatehome – Update the home directory copy and link files for the current label

SYNOPSIS

updatehome [–cirs]

DESCRIPTION

`updatehome` reads the user's minimum-label copy and link-control files (`.copy_files` and `.link_files`), which contain a list of files to be copied and symbolically linked from the user's minimum-label home directory to the user's home directory at the current label.

The Trusted Solaris `dtsession` performs an `updatehome` whenever a newly labeled workspace is created so that the user's favorite files are available for use. For example, the user probably wants a symlink to such files as `.profile`, `.login`, `.cshrc`, `.exrc`, `.mailrc`, and `~/bin`. `updatehome` provides a convenient mechanism for accomplishing this symlink. The user may add files to those to be copied (`.copy_files`) and to those to be symbolically linked (`.link_files`).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

OPTIONS

`–c` Replace existing home-directory copies at the current label. (The default is to skip over existing copies.)

`–i` Ignore errors encountered. (The default aborts on error.)

`–r` Replace existing home-directory copies or symbolic links at the current label. This option implies options `–c` and `–s`. (The default is to skip over existing copies or symbolic links.)

`–s` Replace existing home-directory symbolic links at the current label. (The default is to skip over existing symbolic links.)

RETURN VALUES

Upon success, `updatehome` returns `0`. Upon failure, `updatehome` returns `1` and writes diagnostic messages to standard error.

EXAMPLES

EXAMPLE 1 A Sample copy_files

The files listed in `.copy_files` can be modified at every user's label.

```
.cshrc
.mailrc
.netscape/bookmarks.html
```

EXAMPLE 2 A Sample link_files

The files listed in `.link_files` can be modified at the lowest label, and the changes will propagate to the other labels available to the user.

```
~/bin
.netscape/preferences
```

```
.xrc
.rhosts
```

FILES

`$HOME/.copy_files`	List of files to be copied
`$HOME/.link_files`	List of files to be symbolically linked

SEE ALSO

SunOS 5.7 Reference Manual

`attributes`(5)

NAME

writeaudit – Write an audit record

SYNOPSIS

writeaudit *event* [–a *type:value* …] [–f *type:filename*…]

DESCRIPTION

For a specified event, this command writes an audit record containing zero or more attributes. If no AW_RETURN attribute is specified, a successful return attribute (0,0) will be included in the audit record. Multiple –a or –f options can be specified on a single writeaudit call.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtsu

FIELDS

event The name of the event to record in the audit record. This option must always be present. The name must be defined in audit_event file. See audit_event(4).

OPTIONS

–a *type:value* Add an attribute to the audit record. The type must be AW_DATA, AW_ILABEL, AW_INADDR, AW_OPAQUE, AW_PATH, AW_RETURN, AW_SLABEL, or AW_TEXT. Valid formats for *value* are described below.

–f *type:filename* Add an attribute to the audit record. The *type* must be AW_DATA, AW_ILABEL, AW_INADDR, AW_OPAQUE, AW_PATH, AW_RETURN, AW_SLABEL, or AW_TEXT. The *value* is read from the file *filename.* Valid formats for *value* are described below.

AW_DATA Format

AW_DATA: *printformat* :*itemsize* :*numberitems*:*item1*: . . . *itemN*

The *printformat* field must be one of these:

AWD_BINARY Print data in binary

AWD_OCTAL Print data in octal

AWD_DECIMAL Print data in decimal

AWD_HEX Print data in hex

AWD_STRING Print data as a string

The *itemsize* field must be one of these:

AWD_BYTE Data is in units of bytes

AWD_CHAR Data is in units of chars (1 byte)

AWD_SHORT Data is in units of shorts (2 bytes)

	AWD_INT Data is in units of ints (4 bytes)
	AWD_LONG Data is in units of longs (4 bytes)
	numberitems specifies the number of items to be printed and must be an integer in the range 1-255.
	item1 through *itemN* specify the data fields to be printed and must be entered in hex (for example, 0xfff), octal (for example, 0777), or decimal.
AW_INADDR Format	AW_INADDR:*hostname*
	hostname must be a valid hostname (for example, hamlet), or a standard IP address (for example, 129.150.117.44).
AW_OPAQUE Format	AW_OPAQUE:*numberitems*:*item1: ... itemN*
	numberitems specifies the number of items to be printed and must be an integer in the range 1-255.
	item1 through *itemN* specify the fields to be printed and must be input in hex (for example, 0xfff), octal (for example, 0777), or decimal. Each field must not exceed 1 byte in length.
AW_PATH Format	AW_PATH:*path*
	path is a text string (for example, /usr/bin/).
AW_RETURN Format	AW_RETURN:*status_value : return_value*
	status_value identifies the error status of the call and must be an integer in the range 0-255.
	return_value identifies the call return value and must be an integer in the range 0-255.
AW_SLABEL Format	AW_SLABEL:*sensitivity_label*
	sensitivity_label must be a valid character-coded sensitivity label; for example, S AB or
	0x7fffffffffffffffffffffffffffffff\ ffffffffffffffffffffffffffffffff
AW_TEXT Format	AW_TEXT:*string*
	string must be a text string; for example, successful change.
EXAMPLES	**EXAMPLE 1** Write Event Records
	For the event, write an AUE_event record containing the string successful change:

```
writeaudit AUE_event
-a AW_TEXT:
"successful change"
```

For the event, read the text string from the file `eventfile` and write an `AUE_event` record (the file `eventfile` might, for example, contain the string `successful change`):

```
writeaudit AUE_event -f \ AW_TEXT:eventfile -a AW_RETURN:-1:4
```

For the event, write an `AUE_event` record containing the specified arbitrary data:

```
writeaudit AUE_event -a\ AW_DATA:AWD_DECIMAL:AWD_BYTE:5:1:2:3:4:5
```

SEE ALSO

Trusted Solaris 7 Reference Manual

`audit`(2), `auditwrite`(3), `audit_event`(4)

SunOS 5.7 Reference Manual

`attributes`(5)

NOTES

This command must have the `proc_audit_appl` privilege in its set of effective privileges. To translate labels (for example, *type* `AW_ILABEL` or `AW_SLABEL`) that dominate the process's sensitivity label, this command must have the `priv_sys_trans_label` privilege in its set of effective privileges.

These interfaces are uncommitted. Although they are not expected to change between minor releases of the Trusted Solaris environment, they may.

Information labels (ILs) are not supported in Trusted Solaris 7 and later releases. Trusted Solaris software interprets any ILs on communications and files from systems running earlier releases as `ADMIN_LOW`.

Objects still have CMW labels, and CMW labels still include the IL component: `IL[SL]`; however, the IL component is fixed at `ADMIN_LOW`.

As a result, Trusted Solaris 7 has the following characteristics:

- ILs do not display in window labels; SLs (Sensitivity Labels) display alone within brackets.
- ILs do not float.
- Setting an IL on an object has no effect.
- Getting an object's IL will always return `ADMIN_LOW`.
- Although certain utilities, library functions, and system calls can manipulate IL strings, the resulting ILs are always `ADMIN_LOW`, and cannot be set on any objects.

- In auditing, the `ilabel` token is recorded as `ADMIN_LOW`, when it is recorded. The audit event numbers 519 (AUE_OFLOAT), 520 (AUE_SFLOAT), and 9036 (AUE_iil_change) continue to be reserved, but those events are no longer recorded.

Index

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

W