Trusted Solaris 2.5 Man Pages: 1MTSOL Administrator Commands

Sun Microsystems Federal, Inc.
A Sun Microsystems, Inc. Business
901 San Antonio Road, MS USJC01-201
Palo Alto, CA 94303
U.S.A.

Preface

In the Trusted Solaris Reference Manual, each collection of information on a particluar topic is called a man page, even though a man *page* may actually consist of *many pages* of text.

A man page is intended to answer concisely the question “What does it do?”. The man pages are not intended to be a tutorial. Depending what you are trying to do, refer to the other Trusted Solaris user, developer, and administrator manuals for when and why to use a command or other features described in the man pages.

ACCESSING MAN PAGES

The man pages that make up the reference manual may be accessed in three ways.

Note: The following discussion of man page viewing options uses the term **package**, which is a unit of software that is typically delivered on Sun’s product CDs. Installing the documentation packages is optional, because they are not required for operations. Each customer’s administrators decides whether or not the documentation packages are installed and made available.

The first means of accessing the man pages is through the use of the **man**(1) command. When the contents of the man page package, SUNWman, are available on the local system, anyone with a login account, plus a terminal emulator (such as **cmdtool**(1), **shelltool**(1), or **dtterm**(1)) and the **man**(1) command in one of the account’s execution profiles can view a man page on-line. (For more about Trusted Solaris execution profiles and user accounts, see the Trusted Solaris user and administrator

documentation.) To view a man page, enter the **man** command followed by the name of the man page. For example, to view the **ls**(1) man page that describes the command used to print out a directory's contents, a user enters the command: **man**ls.

The second way to read man pages is in the printed Trusted Solaris Reference Manual. The reference manual is in the Trusted Solaris documentation set, and it may be ordered in hardcopy form from Sun by using part number: 805-8005-10.

The third means of reading the man pages is by viewing them in AnswerBook format. When the Trusted Solaris AnswerBook package, SUNWtab, is available on the local system, anyone with a login account and with the **answerbook**() command and a terminal emulator in an execution profile can display the Trusted Solaris reference manual and the other user documentation. For Trusted Solaris 2.5, the Trusted Solaris documentation AnswerBook is shipped on a separate documentation CD, but it may be bundled on the same CD with the Trusted Solaris software in future releases.

Trusted Solaris man pages are identified with a TSOL suffix in the section name. The TSOL suffix is used for man pages that are either new to Trusted Solaris or modified from the base man pages from the Solaris, CDE, or Solstice products that are bundled into Trusted Solaris. The man pages are organized alphabetically by section.

- Section 1TSOL describes new or modified user commands available with the Trusted Solaris operating system.

- Section 1BTSOL describes printer commands adapted for Trusted Solaris from the Berkeley Software Distribution (BSD) print subsystem, which are used chiefly for printing administration.

  Note: Use of the equivalent System V print commands is recommended (such as **lp**(1TSOL)instead of **lpr**(1BTSOL)) because although the BSD commands are included for compatability, they will be removed in future releases.

- Section 1MTSOL describes Trusted Solaris system maintenance and administration commands.

- Section 2TSOL describes Trusted Solaris system calls. Most of these calls have one or more error returns. An error condition is indicated by an otherwise impossible returned value.

- 3*TSOL subsections describe functions found in various Trusted Solaris libraries, other than those functions that directly invoke UNIX system primitives, which are described in Section 2TSOL.

Subsections include: 3CTSOL, 3NTSOL, 3RTSOL, 3TSOL, and 3X11TSOL.

- Section 4TSOL outlines the formats of various files. The C structure declarations for the file formats are given where applicable.
- Section 5TSOL contains miscellaneous documentation such as Trusted Solaris macros.
- 7*TSOL subsections describe various special files that refer to specific hardware peripherals and device drivers.

  Subsections include: 7DTSOL and 7TSOL.
- 9*TSOL subsections provide reference information for writing device drivers in the kernel operating system environment.

  Subsections include: 9FTSOL and 9TSOL.

Following is a generic list of headings on each man page. The man pages of each manual section include only the headings they need. 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. Literal characters (commands and options) are in **bold** font and variables (arguments, parameters and substitution characters) are in *italic* font. 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. The protocol specification pathname is always listed in **bold** font.

AVAILABILITY

This section briefly states any limitations on the availabilty of the command. These limitations could be hardware or software specific.

A specification of a class of hardware platform, such as **x86** or **SPARC**, denotes that the command or interface is applicable for the hardware platform specified.

In Section 1TSOL and Section 1MTSOL, **AVAILABILITY** indicates which package contains the command being described on the manual page. In order to use the command, the specified package must have been installed with the operating system. If the package was not installed, see **pkgadd**(1) for information on how to upgrade.

MT-LEVEL

This section lists the **MT-LEVEL** of the library functions described in the Section 3 manual pages. The **MT-LEVEL** defines the libraries' ability to support threads. See **Intro**(3TSOL) for more information.

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 7TSOL 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**(7).

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 as **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 in an administrative role,

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

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.

SEE ALSO

This section lists references to other man pages, in-house documentation, and outside publications.

DIAGNOSTICS

This section lists diagnostic messages with a brief explanation of the condition causing the error. Messages appear in **bold** font with the exception of variables, which are in *italic* font.

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.

SUMMARY OF TRUSTED SOLARIS CHANGES

On base man pages that have Trusted Solaris modifications, this section summarizes the changes in a single easy-to-find place on the man page.

NAME

Intro, intro – introduction to maintenance commands and application programs

AVAILABILITY

SUNWman

NOTE

In the *Trusted Solaris Reference Manual*, the **AVAILABILITY** section indicates which package contains the command being described on the current man page. Before the command can be used, the indicated package must be installed. See **pkginfo**(1) for how to check which packages are installed. See **pkgadd**(1) for how to add a package.

In the Trusted Solaris environment, even if a particular command is installed, the command may not be usable by anyone unless the site's *security administrator* has included that command in an *execution profile* that has been assigned to one or more users. The security administrator may restrict the use of any command and may change any of a command's *security attributes* using the *profile mechanism*. *Security administrator*, *security attributes*, *execution profiles* , and other new Trusted Solaris terms mentioned on the user commands man pages are defined in the **DEFINITIONS** section of **Intro**(1TSOL) and explained further in the *Trusted Solaris user's document set* and the *Trusted Solaris administrator's document set*. If any of the commands described in this section do not work at all or they do not work as expected, check with your security administrator.

DESCRIPTION

Section 1M of the *Trusted Solaris Reference Manual* describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes in the Trusted Solaris operating system. The Trusted Solaris system is based on the Solaris operating system, the Common Desktop Environment (CDE) window system, and the Solstice AdminSuite set of system administration tools. Man pages whose section IDs end with the 1MTSOL suffix describe administrative commands that are either new or modified to work within Trusted Solaris *security policy*. An example of a new Trusted Solaris administrative command added to the combined base Solaris, CDE, and Solstice functionality is **adminvi**, which is described on the **adminvi**(1MTSOL) man page. The **adminvi** command is a modified version of the **vi**(1) command that allows administrators and other users to edit files on the command line while preventing certain **vi** actions that present a security risk.

Modified commands are commands from any of the base products that have been modified to work within the Trusted Solaris security policy, such as: **mount**. Man pages for modified commands have been rewritten to remove information that is not accurate for how the command behaves within the Trusted Solaris system. Modified man pages, such as **mount**(1MTSOL), also add descriptions for any new features, options, and arguments added to the base.

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**(1MTSOL), **mount_cachefs**(1MTSOL), **mount_hsfs**(1MTSOL), **mount_nfs**(1MTSOL), **mount_tmpfs**(1MTSOL), and, **mount_ufs**(1MTSOL). 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.

NOTE

The *printed* version of the *Trusted Solaris Reference Manual* includes only the Trusted Solaris man pages, while the *on-line man pages* that are viewable with the **man**(1) command include all the base man pages along with the Trusted Solaris man pages. The **man** command without any options always displays the Trusted Solaris version, so when both a base man page and a Trusted Solaris version exist, if you want to view the original man page you must use the **man** command with the **–s** option to specify the base section ID of the man page. For example, to display the **mount**(1M) man page instead of the modified **mount**(1MTSOL) man page, you would enter: **man –s1m mount.** To find out all the sections that contain man pages with the same name, enter: **man –l <man_page_name>.**

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 ENTERING AND DISPLAY 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 **Intro**(1TSOL). For rules for entering labels in configuration files, see **Intro**(4TSOL).

Enter a sensitivity label (SL), information label (IL), or clearance, in ASCII 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**(4TSOL). 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.

NOTE: By convention, words appear in sensitivity labels in reverse order to the way they appear in information labels. Order doesn't matter on input. TS A B in an SL is displayed as TS B A in an IL.

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 ASCII 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 (/).

EXAMPLES

- On the command line, enclose any label with more than one word in in double quotes because, without quotes, a second word or letter separated by a space is interpreted as a second argument.
  setlabel –i "C A B" somefile
  setlabel –s SECRET somefile
- Enclose labels containing **[** and **]** characters in quotes to suppress the shell's use of those characters in filename substitution.
  setlabel –s "[SECRET]" somefile
- Use any combination of upper and lowercase letters. You may separate items in a label with blanks, tabs, commas or slashes (/). Don't use any other punctuation.
  setlabel "CONFIDENTIAL[ts a b]" somefile
  setlabel "confidential[ts,a,b] somefile
  setlabel "confidential[ts/a b]" somefile
- 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
- To set somefile's IL to CONFIDENTIAL.

setlabel –i confidential somefile

- To set somefile's IL to ADMIN_LOW and SL to CONFIDENTIAL.
  setlabel "admin_low[confidential]" somefile
- 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
- To set somefile's IL to SECRET B A. (Remember that the words in an IL appear in reverse order to the words in an SL.)
  setlabel -i secret,b/a somefile
- When the IL is SECRET B A, reset the IL to CONFIDENTIAL.
  setlabel –i +confidential somefile
- To set somefile's IL to SECRET B.
  setlabel -i "secret a B –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 may perform privilege debugging [see **runpd**(1MTSOL)] to find out which privileges are needed and may 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 forced on the command or given to the command in the user's execution profile by the security administrator, the command or the option will not work at all. Required privileges are indicated on the man page with the words "must have," as shown in this sentence: "The **ifconfig**(1MTSOL) 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 then 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 doesn't 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 or alternate 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 document set*.

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

The printed reference manual contains only the Trusted Solaris new and modified man pages, while the on-line set of man pages viewed by the **man** command contains both the man pages from the base product and the Trusted Solaris man pages.

Commands may not work as expected in the Trusted Solaris system 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.

Besides the usual UNIX DAC checks performed when a executing command acting on behalf of a user attempts to access a file or directory, there are *mandatory access* checks that also must be passed. For each type of access failure that can occur there is an override *privilege* that may be assigned to the command at the security administrator's discretion.

NOTE

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.

SEE ALSO

getopt(1), **pkgadd**(1M), **runpd**(1MTSOL), **getopt**(3C), the *Trusted Solaris user's document set*, and the *Trusted Solaris administrator's document set*.

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(1MTSOL)	accept or reject print requests
add_install_client(1MTSOL)	See **install_scripts**(1MTSOL)
adminvi(1MTSOL)	edit text with restrictions
add_drv(1MTSOL)	add a new device driver to the system
allocate(1MTSOL)	device allocation
arp(1MTSOL)	address resolution display and control
atohexlabel(1MTSOL)	convert an ASCII coded label to its hexadecimal
audit(1MTSOL)	control the behavior of the audit daemon
auditconfig(1MTSOL)	configure auditing
auditd(1MTSOL)	audit daemon
auditreduce(1MTSOL)	merge and select audit records from audit trail files
auditstat(1MTSOL)	display kernel audit statistics
audit_startup(1MTSOL)	audit subsystem initialization script
audit_warn(1MTSOL)	audit daemon warning script
automount(1MTSOL)	install automatic mount points
automountd(1MTSOL)	autofs mount / unmount daemon
autopush(1MTSOL)	configures lists of automatically pushed STREAMS modules
autopush(1MTSOL)	configures lists of automatically pushed STREAMS modules
bootparamd(1MTSOL)	See **rpc.bootparamd**(1MTSOL)
bsmconv(1MTSOL)	enable / disable the auditing module
bsmunconv(1MTSOL)	See **bsmconv**(1MTSOL)
check(1MTSOL)	See **install_scripts**(1MTSOL)
chk_encodings(1MTSOL)	check label-encodings file syntax
chroot(1MTSOL)	change root directory for a command
cron(1MTSOL)	clock daemon
deallocate(1MTSOL)	device deallocation
device_clean(1MTSOL)	device clean programs
devpolicy(1MTSOL)	configure device policy
dispadmin(1MTSOL)	process scheduler administration
dl_booting(1MTSOL)	inform the kernel that a machine is in the state of

	disklessly booting or in the normal state
dl_restore(1MTSOL)	See **dl_booting**(1MTSOL)
dminfo(1MTSOL)	report information about a device entry in a device maps file
drvconfig(1MTSOL)	configure the /devices directory
du(1MTSOL)	summarize disk usage
eeprom(1MTSOL)	EEPROM display and load utility
fsdb_ufs(1MTSOL)	ufs file system debugger
fuser(1MTSOL)	identify processes using a file or file structure
getfsattr(1MTSOL)	display the file system security attributes
halt(1MTSOL)	stop the processor
hextoalabel(1MTSOL)	convert a hexadecimal label to its ASCII coded equivalent
ifconfig(1MTSOL)	configure network-interface parameters
in.ftpd(1MTSOL)	file-transfer protocol server
in.named(1MTSOL)	Internet domain name server
in.rarpd(1MTSOL)	DARPA Reverse Address Resolution Protocol server
in.rexecd(1MTSOL)	remote execution server
in.rlogind(1MTSOL)	remote login server
in.rshd(1MTSOL)	remote shell server
in.tftpd(1MTSOL)	Internet Trivial File Transfer Protocol server
inetd(1MTSOL)	Internet services daemon
init(1MTSOL)	process control initialization
install_scripts(1MTSOL)	scripts used to install the Solaris software
list_devices(1MTSOL)	list allocatable devices
lpadmin(1MTSOL)	configure the LP print service
lpfilter(1MTSOL)	administer filters used with the LP print service
lpforms(1MTSOL)	administer forms used with the LP print service
lpmove(1MTSOL)	See **lpsched**(1MTSOL)
lpsched(1MTSOL)	start/stop the LP print service and move requests
lpshut(1MTSOL)	See **lpsched**(1MTSOL)
lpsystem(1MTSOL)	register remove systems with the print service
lpusers(1MTSOL)	set printing queue priorities
modload(1MTSOL)	load a kernel module
modunload(1MTSOL)	unload a module

mount(1MTSOL)	mount or unmount file systems and remote resources
mount_nfs(1MTSOL)	mount remote NFS resources
mount_ufs(1MTSOL)	mount ufs file systems
mountd(1MTSOL)	NFS mount request server
named-xfer(1MTSOL)	See **in.named**(1MTSOL)
named(1MTSOL)	See **in.named**(1MTSOL)
ndd(1MTSOL)	get and set driver configuration parameters
netstat(1MTSOL)	show network status
nis_cachemgr(1MTSOL)	NIS+ utility to cache location information about NIS+ servers
nisd(1MTSOL)	See **rpc.nisd**(1MTSOL)
nisd_resolv(1MTSOL)	See **rpc.nisd_resolv**(1MTSOL)
nispasswdd(1MTSOL)	See **rpc.nispasswdd**(1MTSOL)
nispopulate(1MTSOL)	populate the NIS+ tables in a NIS+ domain
nissetup(1MTSOL)	initialize a NIS+ domain
newsecfs(1MTSOL)	See **setfsattr**(1MTSOL)
nfsd(1MTSOL)	NFS daemon
nfsstat(1MTSOL)	NFS statistics
nisd(1MTSOL)	See **rpc.nisd**(1MTSOL)
nscd(1MTSOL)	name service cache daemon
nslookup(1MTSOL)	query name servers interactively
nstest(1MTSOL)	DNS test shell
pbind(1MTSOL)	control and query bindings of processes to processors
praudit(1MTSOL)	print contents of an audit trail file
prtconf(1MTSOL)	print system configuration
pfsh(1MTSOL)	profile shell
poweroff(1MTSOL)	See **halt**(1MTSOL)
psradm(1MTSOL)	set processors on line or off line
rarpd(1MTSOL)	See **in.rarpd**(1MTSOL)
rdate(1MTSOL)	set system date from a remote host
reboot(1MTSOL)	restart the operating system
reject(1MTSOL)	See **accept**(1MTSOL)
rem_drv(1MTSOL)	remove a device driver from the system
rm_install_client(1MTSOL)	See **install_scripts**(1MTSOL)

route(1MTSOL)	manually manipulate the routing tables
rpc.bootparamd(1MTSOL)	boot parameter server
rpc.getpeerinfod(1MTSOL)	getpeerinfo service daemon
rpc.nisd(1MTSOL)	NIS+ service daemon
rpc.nisd_resolv(1MTSOL)	NIS+ service daemon
rpc.nispasswdd(1MTSOL)	NIS+ password update daemon
rpc.tbootparamd(1MTSOL)	Trusted Solaris boot parameter server
rpcbind(1MTSOL)	universal addresses to RPC program number mapper
rpcinfo(1MTSOL)	report RPC information
runpd(1MTSOL)	run a command for privilege debugging
rwall(1MTSOL)	write to all users over a network
sendmail(1MTSOL)	send mail over the internet
setaudit(1MTSOL)	run a command with the audit mask set
setfsattr(1MTSOL)	set security attributes on an existing or newly created file system
setuname(1MTSOL)	change machine information
setup_install_server(1MTSOL)	See **install_scripts**(1MTSOL)
share_nfs(1MTSOL)	make local NFS file systems available for mounting by remote systems
snoop(1MTSOL)	capture and inspect network packets
spray(1MTSOL)	spray packets
swap(1MTSOL)	swap administrative interface
sysh(1MTSOL)	system shell
tbootparam(1MTSOL)	send a request to rpc.tbootparamd to inform it that a host is in normal (labeled) state now
telinit(1MTSOL)	See **init**(1MTSOL)
tftpd(1MTSOL)	See **in.tftpd**(1MTSOL)
tnchkdb(1MTSOL)	check file syntax of trusted network databases
tnctl(1MTSOL)	configure Trusted Solaris network daemon control parameters
tnd(1MTSOL)	trusted network daemon
tninfo(1MTSOL)	print out kernel level network information and statistics
tokmapctl(1MTSOL)	configure token-mapping daemon
tokmapd(1MTSOL)	token-mapping daemon

umount(1MTSOL)	See **mount**(1MTSOL)
uadmin(1MTSOL)	administrative control
unshare_nfs(1MTSOL)	make local NFS file systems unavailable for mounting by remote systems
updatehome(1MTSOL)	update the home-directory copy and link files for the current label
writeaudit(1MTSOL)	write an audit record

NAME

accept, reject – accept or reject print requests

SYNOPSIS

accept *destinations*
reject [–**r** *reason*] *destinations*

AVAILABILITY

SUNWlpu

DESCRIPTION

accept allows the queueing of print requests for the named *destinations*. A *destination* can be either a printer or a class of printers. Run **lpstat** –**a** to find the status of *destinations*.

reject prevents queueing of print requests for the named *destinations*. A *destination* can be either a printer or a class of printers. Run **lpstat** –**a** to find the status of *destinations*.

OPTIONS

The following option is useful with **reject**.

–**r** *reason* Assign a *reason* for rejection of requests. This *reason* applies to all *destinations* specified. *reason* is reported by **lpstat** –**a**. It must be enclosed in quotes if it contains blanks. The default reason is **unknown reason** for existing destinations, and **new destination** for destinations just added to the system but not yet accepting requests.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **accept** and **reject** commands requires the **administer printing** authorization.

FILES

/var/spool/lp/*

SEE ALSO

enable(1TSOL), **lp**(1TSOL), **lpstat**(1TSOL), **lpadmin**(1MTSOL), **lpsched**(1MTSOL)

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*

AVAILABILITY

SUNWcsu

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 **NOTES**.

OPTIONS

–**b** *basedir*	Set the path to the root directory of the diskless client. Used on the server to execute **add_drv** for a diskless client. The client machine must be rebooted to install the driver.
–**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 **EXAMPLES**.

EXAMPLES

The following example adds the **SUNW,example** driver to the system, with an alias name of **SUNW,alias**. This example 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**.

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
```

The following example illustrates the case in which 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**. These are **name** and **compatible** properties for this device:

```
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

This example is currently relevant only for devices exporting a generic device name.

EXIT STATUS

Upon success, **add_drv** returns **0**. Upon failure, **add_drv** returns **1**.

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, the command needs the **file_dac_write** privilege.

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

SEE ALSO

boot(1M), **devlinks**(1M), **disks**(1M), **drvconfig**(1M), **kernel**(1M), **modinfo**(1M), **ports**(1M), **rem_drv**(1M), **tapes**(1M), **driver.conf**(4), **system**(4), **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 **–v** option). If a reconfiguration boot would be required to complete the driver installation, **add_drv** will fail unless the **–f** option is specified. See the last example in **EXAMPLES**.

BUGS

add_drv will accept a full path name for *device_driver*. However, the kernel does not use the full path name; the kernel uses only the final component and searches the internal driver search path for the driver. Thus the kernel may load a different driver from the

one expected.

For this reason, do **not** use **add_drv** with a full path name. See **kernel**(1M) for more information on the driver search path.

NAME

adminvi – Edit text with restrictions

SYNOPSIS

adminvi *filename* . . .

AVAILABILITY

SUNWtsolu

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 these 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 file names 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 file name other than the file names 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 file names that were given on the **adminvi** command line.

SEE ALSO

vi(1)

NOTES

These interfaces are uncommitted; although not expected to change between minor releases of Trusted Solaris systems, these interfaces may change.

NAME

allocate – device allocation

SYNOPSIS

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

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

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

AVAILABILITY

SUNWcsu

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.

SUMMARY OF TRUSTED SOLARIS CHANGES

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

FILES

/etc/security/device_allocate
/etc/security/device_maps
/etc/security/dev/∗
/etc/security/lib/∗

SEE ALSO **device_allocate**(4TSOL), **device_maps**(4TSOL)

NAME

arp – Display and control address resolution

SYNOPSIS

/usr/sbin/arp *hostname*

/usr/sbin/arp –**a**

/usr/sbin/arp –**d** *hostname*

/usr/sbin/arp –**f** *filename*

/usr/sbin/arp –**s** *hostname ether_address* [**temp**] [**pub**] [**trail**]

AVAILABILITY

SUNWcsu

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. These are the definitions for the flags in the table:

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; used only for the multicast entry for **224.0.0.0**

This option must be run at a sensitivitly label of ADMIN_HIGH and effective uid of **0**. This restriction can be overridden by inheriting the **file_mac_read** and **file_dac_read** privileges.

–**d** Delete an entry for the host called *hostname*.

–**f** Read the file named *filename* and set multiple entries in the ARP tables. Entries in the file should take the form

hostname ether_address **[temp] [pub] [trail]**

(The description for the –**s** option shows the argument definitions.)

–**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

respomse is useful in certain SLIP or PPP configurations.

SUMMARY OF TRUSTED SOLARIS CHANGES

To run, options –**d**, –**f**, and –**s** need to inherit the **sys_net_config** privilege. Option –**a** needs to be run at a sensitivity label of ADMIN_HIGH and effective uid of **0**; this restriction can be overridden by inheriting the **file_mac_read** and **file_dac_read** privileges.

SEE ALSO

ifconfig(1MTSOL), **arp**(7P)

NAME

atohexlabel – Convert an ASCII coded label to its hexadecimal equivalent

SYNOPSIS

/usr/sbin/atohexlabel [*ASCII_coded_CMW_label*]

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

/usr/sbin/atohexlabel –**i** [*ASCII_coded_information_label*]

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

AVAILABILITY

SUNWtsolu

DESCRIPTION

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

OPTIONS

–**c** Identifies the ASCII coded label as a clearance.

–**i** Identifies the ASCII coded label as an information label.

–**s** Identifies the ASCII coded label as a sensitivity label.

RETURN VALUES

Upon success, this command exits with **0**. Upon failure, this command exits with **1** and writes diagnostics to the standard error file.

FILES

/etc/security/tsol/label_encodings
The label-encodings file containing the **CLASSIFICATIONS**, **WORDS**, constraints, and values for the defined labels of this system

SEE ALSO

label_encodings(4TSOL)

Trusted Solaris administrator's document set

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**

AVAILABILITY

SUNWcsu

DESCRIPTION

The **audit**(1MTSOL) command is the general administrator's interface to maintaining the audit trail. The administrator can request the audit daemon to read the contents of the **audit_control**(4TSOL) file and re-initialize the current audit directory to the first directory listed in the **audit_control** file; to open a new audit file in the current audit directory specified in the **audit_control** file as last read by the audit daemon; or to close the audit trail and disable auditing.

OPTIONS

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

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

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

RETURN VALUES

Upon success, the **audit** command returns **0**. Upon failure, the **audit** command returns a positive integer.

SUMMARY OF TRUSTED SOLARIS CHANGES

This command should run at ADMIN_HIGH.

FILES

/etc/security/audit_user

/etc/security/audit_control

SEE ALSO

bsmconv(1MTSOL), **praudit**(1MTSOL), **audit**(2TSOL), **audit_control**(4TSOL), **audit_user**(4TSOL)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module has been enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

Trusted Solaris 2.x will soon extend the number of audit classes and introduce new but similar structures and programming interfaces.

This command does not modify a preselection mask of a process. The command affects only the selection of audit directories for audit-data storage and the specification of the minimum size free.

NAME

audit_startup – Script to initialize audit subsystem

SYNOPSIS

/etc/security/audit_startup

DESCRIPTION

The **audit_startup** script is used to initialize the audit subsystem before the audit daemon is started. This script, configurable by the system administrator, currently consists of a series of **auditconfig**(1MTSOL) 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 has been enabled on Trusted Solaris systems.

SEE ALSO

auditconfig(1MTSOL), **auditd**(1MTSOL), **bsmconv**(1MTSOL)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module has been enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

Trusted Solaris 2.x will soon extend the number of audit classes and introduce new but similar structures and programming interfaces.

NAME

audit_warn – Script activating warning messages from the audit daemon

SYNOPSIS

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

AVAILABILITY

SUNWcsu

DESCRIPTION

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

By defining a mail alias called *audit_warn* in **aliases**(4), the system administrator can specify a list of mail recipients to be notified when an audit-warning situation arises. The users that make up the **audit_warn** alias are typically the **audit** and **root** users.

OPTIONS

allhard *count* Indicates that the hard limit for all file systems 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 to avoid saturation of the file system that contains the mail spool directory.

allsoft Indicates that the soft limit for all file systems 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**(4TSOL). The audit daemon will sleep 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 to avoid 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

By default, the audit module has been enabled on Trusted Solaris systems.

SEE ALSO

audit(1MTSOL), **auditd**(1MTSOL), **bsmconv**(1MTSOL), **aliases**(4), **audit.log**(4TSOL), **audit_control**(4TSOL)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module has been enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

Trusted Solaris 2.x will soon extend the number of audit classes and introduce new but similar structures and programming interfaces.

NAME

auditconfig – Configure auditing

SYNOPSIS

auditconfig [*args*]

AVAILABILITY

SUNWcsu

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.

-getcond Display the kernel audit condition. The condition displayed is a literal string: **auditing** means that auditing is enabled and turned on (the kernel audit module is constructing and queuing audit records); **noaudit** means that auditing is enabled but turned off (the kernel audit module is not constructing and queuing audit records); **disabled** means that the audit module has not been enabled. See **auditon**(2TSOL) and **auditd**(1MTSOL) for further information.

-setcond[**auditing** | **noaudit**]
Set the kernel audit condition to the *condition* specified by a literal string: **auditing** means that auditing should be enabled; **noaudit** means that 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**(4TSOL) for further information.

-getkmask Display the kernel preselection mask for nonattributable events.

-setkmask [+-] *audit_flag* **[** ,*audit_flag* ... **]**
Set the kernel preselection mask for nonattributable audit events to the classes specified by *audit_flags*. An *audit_flag* is a two-character string representing an audit class. The minus (-) modifier indicates that failure events in the represented class are audited. The plus (+) modifier indicates that success events in the represented class are audited. No modifier indicates that both success and failure events in the represented class are audited. See **audit_control**(4TSOL) for further information.

-setkmaskac Set the kernel preselection mask for nonattributable audit events to the classes defined by the **naflags** field of the **audit_control**(4TSOL) file.

-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 ASCII representation of the flags similar to that in **audit_control**(4TSOL).

-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 a literal string that denotes an audit policy. A prefix of plus (**+**) adds the policies specified to the current audit policies. A prefix of minus (**-**) removes the policies specified from the current audit policies. The next section lists and describes the valid policy-flag strings (listed by **auditconfig**-lspolicy.

Policy Flags

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, processes are suspended until audit resources become available.

group Include the supplementary group token in audit records. By default, the group token is not included.

ilabel	Include ilabels in audit records. However, if ilabels are not enabled on this system, ilabels will not be generated regardless of this flag. This information is not included by default.
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 path names 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

```
# 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
```

RETURN VALUES

Upon success, **auditconfig** returns **0**. Upon failure, **auditconfig** returns **1**.

SUMMARY OF TRUSTED SOLARIS CHANGES

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

FILES

/etc/security/audit_event
/etc/security/audit_class

SEE ALSO

auditd(1MTSOL), **bsmconv**(1MTSOL), **praudit**(1MTSOL), **auditon**(2TSOL), **execv**(2), **audit_class**(4TSOL), **audit_control**(4TSOL), **audit_event**(4TSOL)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module has been enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

NAME

auditd – Audit daemon

SYNOPSIS

/usr/sbin/auditd

AVAILABILITY

SUNWcsu

DESCRIPTION

The audit daemon controls the generation and location of audit-trail files. If auditing is desired, **auditd** reads the **audit_control**(4TSOL) 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 file system 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 is reread, and a new trail is opened. If **SIGTERM** is received, the audit trail is closed and auditing is terminated. The program **audit**(1MTSOL) sends these signals and is recommended for this purpose.

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

Auditing Conditions

The audit daemon invokes the **audit_warn**(1MTSOL) program 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**(4TSOL). 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 become unavailable for some reason. A new audit trail has been opened on another file system.
audit_warn allhard *count*	All available file systems have been filled or become unavailable for some reason. 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	Someone other than the audit daemon has changed the internal system audit condition from **AUC_AUDITING**. This change 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

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

FILES

/etc/security/audit/audit_control
/etc/security/audit/audit_data

SEE ALSO

audit(1MTSOL), **audit_warn**(1MTSOL), **bsmconv**(1MTSOL), **praudit**(1MTSOL), **auditon**(2TSOL), **auditsvc**(2TSOL), **audit.log**(4TSOL), **audit_control**(4TSOL), **audit_data**(4TSOL)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module is enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

NAME

auditreduce – Merge and select audit records from audit-trail files

SYNOPSIS

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

AVAILABILITY

SUNWcsu

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 into a single output file the audit records from one or more input audit-trail files. Assuming that the records in an audit-trail file are sorted in chronological order (oldest first), **auditreduce** maintains this order 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**(4TSOL) for details of the structure of the audit root.] Without the **-R** or the **-S** option, *audit_root_dir* defaults to **/etc/security/audit**. Using the file-selection options enables selection of some subset of these files or files from another directory or files named explicitly on the command line.

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

Audit-Trail File-Name Format

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

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

where *start-time* is the 14-character time stamp showing when the file was opened, *end-time* is the 14-character time stamp showing when the file was closed, and *suffix* is either the name of the machine that generated the audit-trail file or some other meaningful suffix , such as **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 audit system is still writing to the file. Time stamps take the form *yyyymmddhhmmss* (year, month, day, hour, minute, second). The time stamps 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 time stamp. This option effectively disables the –**a**, –**b**, and –**d** record-selection options. This option is useful in preventing the loss of records if the –**D** option is used to delete the input files after they are processed. However, if another option forbids a record's selection, –**A** will not override that option.

–C	Process only complete files. Files whose file-name *end-time* time stamp is **not_terminated** are not processed. (The audit system is currently writing to such a file.) This option is useful in preventing the loss of records if **–D** is used to delete the input files after they are processed. This option does not apply to files specified on the command line.
–D *suffix*	Delete input files after they are processed. The files are deleted only if the entire run is successful. If **auditreduce** detects an error while reading a file, then that file is not deleted. Specifying **–D** implies **–A**, **–C**, and **–O** also. *suffix* is given to the **–O** option to help 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*	Select records from files with *machine* as the file-name suffix. If **–M** is not specified, all files are processed regardless of suffix. **–M** can be used also to allow selection of records from files that contain combined records from many machines and have a common suffix (such as **all**). Contrast this option with **–h**, which uses the content of an audit record to select where a particular audit record was collected.
–O *suffix*	Direct output stream to a file in the current *audit_root_dir* with the indicated suffix. *suffix* may alternatively contain a full path name. If so, the last component is taken as the suffix, ahead of which the time stamps (*yyyymmddhhmmss*) will be placed, ahead of which the remainder of the pathname will be placed for the name of the output file. If the **-O** option is not specified, the output is sent to the standard output. When it places time stamps in the file name, **auditreduce** 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*	Use *pathname* in place of the audit root directory *audit_root_dir*. Examine *pathname*/*/**files** rather than using **/etc/security/audit/***/**files** by default.
–S *server*	This option causes **auditreduce** to read audit-trail files from a specific location (server directory). Because *server* is normally interpreted as the name of a subdirectory of the audit root, **auditreduce** will look in *audit_root_dir*/*server*/**files** for the audit-trail files. However, if *server* contains any slash (/) 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 state the total number of records that were written to the output stream.

Record-Selection Options

The record-selection options listed are used to indicate which records **auditreduce** writes to the output file.

NOTE: 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 subsequently under **Option Arguments**. *date-time* is 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; select only records with events that are mapped to the audit classes specified by *audit-classes*. Audit-class names are defined in **audit_class**(4TSOL). The *audit-classes* can be a comma-separated list of *audit flags* like those described in **audit_control**(4TSOL). Using the *audit flags*, one can use success and failure as selection criteria.

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

−h *hostmachine* Select records generated on *hostmachine* using the content of the audit record, not the audit-file name. Contrast this option with **−M**, which bases selection on file name.

−i *information-label* Select records with the specified *information-label*, which may be a range as explained under **Option Arguments**, *information-label*.

−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 either 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.* These are allowable object types and values:

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 preceded by a tilde (˜), 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. Surround the *pathname* with quotes to prevent the shell from expanding any tildes.

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:** Processes 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 file-name format. However, –**M**, –**S**, and –**R** may not be used when processing named files. If the *filename* is hyphen (-), then the input is taken from the standard input.

Option Arguments

audit-trail-file An audit-trail file as defined in **audit.log**(4TSOL). 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:

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

start-time is the 14-character time stamp denoting when the file was opened. *end-time* is either the 14-character time stamp denoting when the file was closed or the literal string **not_terminated**, indicating either that the audit daemon is still writing to the file or that the file was not closed properly (a system crash or abrupt halt occurred). *suffix* is either the name of the machine that generated the audit-trail file or some other meaningful suffix; for example, **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 absolute or offset. An absolute *date-time* takes the form:

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

where *yyyy* is a year (1970 at the earliest), *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 for *hh*, *mm*, and *ss* is **00**.

An offset can be specified as **+***nd*|*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. Because an offset is relative to the starting time, this form can be used only with the –**b** option.

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

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

information-label The literal string representation of either an exact, valid information label or a range of two valid information labels

To specify a range, use **[***x***]:[***y***]** where *x* and *y* are valid information 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.

pathname A regular expression describing a path name

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

RETURN VALUES

Upon success, **auditreduce** returns **0**. Upon failure, **auditreduce** returns **1**.

EXAMPLES

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

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, delete the original files to prevent the records from appearing twice:

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

Print what user **wetmore** did on April 13, 1988; and display the output in a human-readable form to the standard output:

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

Because the previous example may produce a large volume of data if **wetmore** has been busy, look at only login and logout times:

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

The **–c** option selects records from a specified class.

Record **wetmore**'s login / logout activity for April 13, 14, and 15 in a file in the current working directory:

```
% auditreduce –a 19880413 –b +3d –u wetmore  –c lo –O wetmorelo
```

The output file has **wetmorelo** as the *suffix* and the appropriate time-stamp prefixes. Note that the short form (**lo**) of the audit-event name is used for the **–c** option.

Viewing his directory changes (**chdir**) tracks **wetmore**'s movement about the file system on April 13, 14, and 15. To get the same time range as the previous example, you need to specify the **–b** time as the day after the range 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 wetmore –m AUE_CHDIR | praudit

Determine whether **wetmore** accessed any highly classified information at **SECRET A B**, a valid label on the system:

% auditreduce –a 19880413 –b +3d –u wetmore –s "SECRET A B:ADMIN_HIGH" | praudit

Collect the audit records in summary form (the login/logout records only). The records are being written to a summary file in a different directory from 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

Activity for user ID 9944 has been observed, but that user is not known to the system administrator. 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

SUMMARY OF TRUSTED SOLARIS CHANGES

The Trusted Solaris system has added these record-selection options to this command: **-h** *hostmachine*, **-i** *information-label*, and **-s** *sensitivity-label*. These option arguments have been added: *information-label* and *sensitivity-label*. The **EXAMPLES** section shows how to audit a user for access to data at a specific sensitivity label.

FILES

/etc/security/audit/*server***/files/***** Storage location of audit trails

SEE ALSO

bsmconv(1MTSOL), **praudit**(1MTSOL), **audit.log**(4TSOL), **audit_class**(4TSOL), **audit_control**(4TSOL), **group**(4), **hosts**(4), **passwd**(4)

DIAGNOSTICS

If there are command-line errors, **auditreduce** prints error messages and then exits. If there are fatal errors during the run, **auditreduce** prints an explanatory message and exits. In this case, the output file may be in an inconsistent state (missing a trailer or containing a partially written record) and **auditreduce** prints a warning message before exiting.

Because **auditreduce** may be processing a large number of input files, it is possible that the machinewide limit on open files may be exceeded. If it is, **auditreduce** prints a message to that effect, gives information on how many file there are, and exits.

If **auditreduce** prints a record's time stamp in a diagnostic message, that time is local time. However, when file names are displayed, their time stamps are in GMT.

NOTES

This functionality is active only if the audit module has been enabled. By default, this module is enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

Trusted Solaris 2.x will soon extend the number of audit classes and introduce new but similar structures and programming interfaces.

BUGS

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

NAME

auditstat – Display kernel audit statistics

SYNOPSIS

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

AVAILABILITY

SUNWcsu

DESCRIPTION

auditstat displays kernel audit statistics. These fields display total number:

aud Audit records processed by the **audit**(2TSOL) system call

ctl Obsolete

drop Audit records that have been dropped according to the kernel audit policy. See **auditon**(2TSOL), AUDIT_CNT policy for details.

enq Audit records put on the kernel audit queue

gen Audit records that have been constructed (not the number written)

kern Audit records produced by user processes (as a result of system calls)

mem Kbytes of memory currently in use by the kernel audit module

nona Nonattributable (not attributable to any particular user) audit records that have been constructed

rblk Times that **auditsvc**(2TSOL) has blocked waiting-to-process audit data

tot Kbytes of audit data written to the audit trail

wblk Times that user processes blocked on the audit queue at the high-water mark

wrtn 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 displays 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 collections.

-n Display the number of kernel audit events currently configured.

-v Display the version number of the kernel audit-module software.

ERRORS

Upon success, **auditstat** returns **0**. Upon failure, **auditstat** returns **1**.

SUMMARY OF TRUSTED SOLARIS CHANGES

By default, the audit module is enabled on Trusted Solaris systems.

SEE ALSO **auditconfig**(1MTSOL), **praudit**(1MTSOL), **bsmconv**(1MTSOL), **audit**(2TSOL), **auditon**(2TSOL), **auditsvc**(2TSOL)

NOTES This functionality is active only if the audit module has been enabled. By default, this module is enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

NAME

automount – install automatic mount points

SYNOPSIS

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

AVAILABILITY

SUNWcsu

DESCRIPTION

automount is a command that installs **autofs** mount points and associates an **automount** map with each mount point. The **autofs** filesystem monitors attempts to access directories within it and notifies the **automountd**(1M) daemon. The daemon uses the map to locate a filesystem, which it then mounts at the point of reference within the **autofs** filesystem. You can assign a map 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, that follow the automounter map name, may be supplied for 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**(1MTSOL) 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 an direct map, or to perform unmounts for entries that have been removed.

OPTIONS

–t *duration* Specify a *duration*, in seconds, that a file system is to remain mounted when not in use. The default is 5 minutes.

-v Verbose mode. Notify 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, *attribute-list* is a list of security attributes, and *location* specifies a file system from which the directory may be mounted. In the case of a simple NFS mount, *location* takes the form:

host:*pathname*

host is the name of the host from which to mount the file system (it may be omitted if the pathname refers to a local device on which the filesystem resides) and *pathname* is the pathname of the directory to mount.

Replicated Filesystems

Multiple *location* fields can be specified for replicated NFS filesystems, in which case **automount** chooses a server with preference given to a server on the local subnet or net.

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**, the lowest.

Note: 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, you can still apply the weighting. 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.

Multiple servers should be of the same host type with compatible label ranges.

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 output of uname -m.	The architecture name. For example "sun4"
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 "5.3"
OSVERS	The output of uname -v.	The OS version. For example "beta1.0"

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*] [–*Sattribute-list*] [[*mountpoint*] [–*mount-options*] [–*Sattribute-list*] *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   \
        /1.0            svr1,svr2:/export/src/beta/1.0   \
        /1.0/man        svr1,svr2:/export/src/beta/1.0/man
```

automount would 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 Filesystem Types

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

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

or to perform an **autofs** mount:

```
src     -fstype=autofs      auto_src
```

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
```

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 filesystems 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 then 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 filesystems. 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.

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**.

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**, **proc_nofloat**, **sys_mount**.

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.
/etc/security/tsol/vfstab_adjunct	mount-time attributes for file systems.

SEE ALSO

automountd(1M), **mount**(1MTSOL), **vfstab_adjunct**(4TSOL), **fns**(5), **fns_initial_context**(5), **fns_policies**(5),

NFS Administration Guide

NOTES

The –**hosts** map must mount all of the exported NFS filesystems from a server. If frequent access to just a single filesystem is required, it is more efficient to access the filesystem with a map entry that is tailored to mount just the filesystem of interest.

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.

If a directory contains direct map mount points then an **ls –l** in the directory will force all the direct map mounts to occur.

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.

An **autofs** directory associated with an indirect map shows only currently-mounted entries. This is a deliberate policy to avoid inadvertent mounting of every entry in a map via an **ls –l** of the directory.

The multiple location feature for NFS mounts allows the **automountd** daemon to choose the most appropriate server at mount time. While such a mount is in effect, the daemon does not monitor the status of the server. If the server crashes, **automountd** will not select an alternative server from the list.

Default mount options and security attributes can be assigned to an entire map when they follow the automounter map name. These options apply only to map entries that have no mount options or security attributes.

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 [–**Tv**] [–**D** *name=value*]

AVAILABILITY

SUNWcsu

DESCRIPTION

automountd is an RPC server that answers file system 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.

–**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**.

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_upgrate_il**, **net_mac_read**, **net_privaddr**, **net_upgrade_sl**, **net_upgrate_il**, **proc_nofloat**, **proc_audit_tcb**, **proc_setsl**, **proc_setil**, **sys_mount**, **sys_trans_label**.

FILES

/etc/auto_master master map for automounter

SEE ALSO

automount(1MTSOL)

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*

AVAILABILITY

SUNWcsu

DESCRIPTION

This command is used to configure 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

–**f** *filename* Set 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** Get 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** Remove 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. The **sys_devices** privilege is required for this command to succeed.

EXAMPLES

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

SEE ALSO

bdconfig(1M), **ttymon**(1M), **ldterm**(7M), **sad**(7D), **streamio**(7I) **ttcompat**(7M)

STREAMS Programming Guide

NAME

bsmconv, bsmunconv – Enable and disable the auditing module

SYNOPSIS

/etc/security/bsmconv [*rootdir* **...]**
/etc/security/bsmunconv [*rootdir* **...]**

DESCRIPTION

The **bsmconv** and **bsmunconv** scripts are used to enable and disable auditing. The optional argument *rootdir* is a list of one or more root directories of diskless clients.

To enable or disable auditing on a diskless client, a server, or a stand-alone system, assume the secadmin role, start a profile shell, and run the **bsmconv** or **bsmunconv** command. (The profile shell runs these commands as the root user.)

To enable or disable auditing on a diskless client from that client's server, assume the secadmin role, start a profile shell, and run **bsmconv**, specifying the root directory of each diskless client you wish to affect. For example, the command

myhost# bsmconv /export/root/client1 /export/root/client2

enables auditing on the two machines named **client1** and **client2**.

The command

myhost# bsmconv

enables auditing only on the machine called **myhost.** It is no longer necessary to enable auditing on both the server and its diskless clients.

After running **bsmconv**, you can configure the system by editing the files in **/etc/security**. Each diskless client has its own copy of configuration files in its root directory. You may wish to edit these files before rebooting each client.

Following the completion of either script, the affected system(s) must be rebooted to allow the auditing subsystem to come up properly initialized.

SUMMARY OF TRUSTED SOLARIS CHANGES

Auditing and device allocation are both enabled by default.

SEE ALSO

auditd(1MTSOL), **audit_startup**(1MTSOL), **audit.log**(4TSOL), **audit_control**(4TSOL)

NAME

chk_encodings – Check label-encodings file syntax

SYNOPSIS

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

AVAILABILITY

SUNWtsolu

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.

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 containing the **CLASSIFICATIONS**, **WORDS**, constraints, and values for the defined labels of this system

SEE ALSO

label_encodings(4TSOL)

Trusted Solaris administrator's document set

NAME

chroot – Change root directory for a command

SYNOPSIS

/usr/sbin/chroot *newroot command*

AVAILABILITY

SUNWcsu

DESCRIPTION

chroot causes *command* to be executed relative to *newroot*. The meaning of any initial slashes (/) in the path names 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:

chroot *newroot command* >**x**

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

The new root path name 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.

Appropriate privilege is required to run this command.

RETURN VALUES

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

EXAMPLE

chroot provides an easy way to extract **tar** files written with absolute file names 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 you do not have to copy any shared libraries to the *newroot* file system.

SUMMARY OF TRUSTED SOLARIS CHANGES

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

SEE ALSO

cd(1), **chroot**(2TSOL)

NOTES

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

NAME

cron – clock daemon

SYNOPSIS

/usr/sbin/cron

AVAILABILITY

SUNWcsu

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/CRON** 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 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: **proc_nofloat**, **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 system, 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/CRON	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**.

SEE ALSO

at(1TSOL), **crontab**(1TSOL), **sh**(1), **queuedefs**(4)

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*]

AVAILABILITY

SUNWcsu

DESCRIPTION

deallocate deallocates a *device* allocated to the evoking user. *device* can be a device defined in **device_allocate**(4TSOL) or one of the device special files associated with the device. It resets the ownership and the permission on all device special files listed in the **device_maps** file, 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**(4TSOL). 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*. This option requires the trusted path attribute to be successful.
–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.

DIAGNOSTICS

deallocate returns an nonzero exit status in the event of an error.

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.

FILES

/etc/security/device_allocate
/etc/security/device_maps
/etc/security/dev/∗
/etc/security/lib/∗

SEE ALSO

allocate(1MTSOL), **device_allocate**(4TSOL), **device_maps**(4TSOL)

NAME

device_clean – device clean programs

SYNOPSIS

/etc/security/lib/ <*device-clean-program*> *ascii-media-label-string* **-[***A* **|** *D***]**

AVAILABILITY

SUNWcsu

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**(4TSOL) file. Device clean programs are invoked by **allocate**(1MTSOL) and **deallocate**(1MTSOL) 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.

OPTIONS

ascii-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**(1MTSOL) command before the device is allocated to a user.

-D The device clean program is invoked from **deallocate**(1MTSOL) command after the device is deallocated from a user.

FILES

/etc/security/device_allocate

SEE ALSO

allocate(1MTSOL), **deallocate**(1MTSOL), **device_allocate**(4TSOL)

NAME

devpolicy – Configure device policy

SYNOPSIS

devpolicy [**-s**] [**-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.

OPTIONS

–**s**	Silent mode; suppresses warning messages.
–**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.

SEE ALSO

drvconfig(1MTSOL), **device_policy**(4TSOL)

FILES

/etc/security/tsol/device_policy

NAME

dispadmin – Process scheduler administration

SYNOPSIS

dispadmin –l
dispadmin –c *class* **–g [–r** *res***]**
dispadmin –c *class* **–s** *file*

AVAILABILITY

SUNWcsu

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 *UNKNOWN TITLE ABBREVIATION: SYSADMIN2*.)

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 interactive 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 interactive 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 it cannot be expressed as an integer in the specified resolution, the time quantum 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.

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

EXAMPLES

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

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

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

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.

SEE ALSO

priocntl(1), **priocntl**(2), **rt_dptbl**(4), **ts_dptbl**(4),
System Interface Guide

DIAGNOSTICS

dispadmin prints an appropriate diagnostic message if the command fails to overwrite the current scheduler parameters because of 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*]

AVAILABILITY

SUNWtsolu

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.

SEE ALSO

chstate(2TSOL)

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**(4TSOL) 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**(4TSOL) for a *device_name* field matching *dev–name*.
–d *dev–path*	Search by *dev–path*. Search **device_maps**(4TSOL) for a device special pathname in the *device_list* field matching the *dev–path* argument.
–t *dev–type*	Search by *dev–type*. Search **device_maps**(4TSOL) for a *device_type* field matching the given *dev–type*.
–u *dm–entry*	Update the **device_maps**(4TSOL) file. This option is provided to add entries to the **device_maps**(4TSOL) file. The *dm–entry* must be a complete **device_maps**(4TSOL) 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 *proc_setid* privilege to work.

DIAGNOSTICS

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

FILES

/etc/security/device_maps

SEE ALSO

device_maps(4TSOL)

NAME

drvconfig – configure the / devices directory

SYNOPSIS

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

AVAILABILITY

SUNWcsu

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**(1MTSOL)) 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.adjunc** 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 IL[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**(1MTSOL) converts when an entry is added to the file. For readability in the example shown below the first three lines would be entered as a single line, as would the last three lines.

**SD:* 0x0x0000000000000000000000000000000000000000000000000000000000000000 **
**0000000000000000000000000000000000000000000000000000000000000000000000000000 **
[0x7fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff]
**mm:kmem 0x7fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff **
**fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff[0x7fffffffffffff **

fffffffffffffffffffffffffffffffffffffffffffffffffffff]

The above example sets all devices exported by the **sd** node to have an information label of ADMIN_LOW and a sensitivity label of ADMIN_HIGH. The **kmem** device exported by the **mm** driver is set to have an information label of ADMIN_HIGH and 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**(1MTSOL) and **rem_drv**(1MTSOL), and may not be supported in future versions of Solaris and Trusted Solaris.

–**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**(1MTSOL). 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 and information label of devices.

FILES

/devices device nodes directory
/etc/minor_perm minor mode permissions
/etc/security/tsol/minor_perm.adjunct
default sensitivity label and information label
/etc/name_to_major major number binding
/etc/driver_classes driver class binding file

SEE ALSO

sh(1), **add_drv**(1MTSOL), **devlinks**(1M), **disks**(1M), **modinfo**(1M), **modload**(1MTSOL), **modunload**(1MTSOL), **ports**(1M), **rem_drv**(1MTSOL), **tapes**(1M), **path_to_inst**(4)

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* . . .]

AVAILABILITY

/usr/bin/du SUNWcsu

/usr/xpg4/bin/du SUNWxcu4

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 by **/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 by **/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 specified file.

–**d** Do not cross filesystem boundaries. " **du -d /** " would report usage only on the root partition, for example.

–**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 by **/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 no *file* is 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.

ENVIRONMENT

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.

SEE ALSO

ls(1), **stat**(2), **environ**(5)

UNKNOWN TITLE ABBREVIATION: SYSADMIN2

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.

SUMMARY OF TRUSTED SOLARIS CHANGES

The **–M** option processes SLDs when descending multilevel directories.

NAME

eeprom – EEPROM display and load utility

SYNOPSIS

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

AVAILABILITY

SPARC

SUNWcsu
eeprom is available only on OpenBoot-compatible SPARC systems

DESCRIPTION

eeprom displays or changes the values of parameters in the EEPROM. **eeprom** processes parameters in the order given. When processing a *parameter* accompanied by a *value*, **eeprom** makes the indicated alteration to the EEPROM; otherwise **eeprom** 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).

OPTIONS

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

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. Defaults to **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. Defaults to **true**.

boot-command Command executed if **auto-boot?** is **true**. Default value is **boot**.

boot-device Device from which to boot. Defaults to **disk net**.

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

boot-from Boot device and file (OpenBoot PROM version 1.*x* only). Defaults to **vmunix**.

boot-from-diag Diagnostic boot device and file (OpenBoot PROM version 1.*x* only). Defaults to **le( )unix**.

diag-device Diagnostic boot source device. Defaults to **net**.

diag-file File from which to boot in diagnostic mode. Defaults to 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 interpreted by POST. The default value is **platform-dependent**.
diag-switch?	If **true**, run in diagnostic mode. Defaults to **true**.
fcode-debug?	If **true**, include name parameter for plug-in device FCodes. Defaults to **false**.
hardware-revision	System version information.
input-device	Power-on input device (usually **keyboard**, **ttya**, or **ttyb**). Defaults to **keyboard**.
keyboard-click?	If **true** enable keyboard click. Defaults to **false**.
keymap	Keymap for custom keyboard.
last-hardware-update	System update information.
load-base	Default load address for client programs. Default value is **16384**.
local-mac-address?	If true, network drivers use their own MAC address, not system's. Defaults to **false**.
mfg-mode	Manufacturing mode argument for POST. Possible values include **off** or **chamber**. The value is passed as an argument to POST. Default value: **off**.
mfg-switch?	If true, repeat system self-tests until interrupted with STOP-A. Defaults to **false**.
nvramrc	Contents of NVRAMRC. Defaults to empty.
oem-banner	Custom OEM banner (enabled by setting **oem-banner?** to **true**). Defaults to empty string.
oem-banner?	If **true**, use custom OEM banner. Defaults to **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). Defaults to **false**.
output-device	Power-on input device (usually **screen**, **ttya**, or **ttyb**). Defaults to **screen**.
sbus-probe-list	Which SBus slots are probed and in what order. Defaults to **0123**.
screen-#columns	Number of on-screen columns (characters/line). Defaults to **80**.
screen-#rows	Number of on-screen rows (lines). Defaults to **34**.
scsi-initiator-id	SCSI bus address of host adapter, range 0-7. Defaults to **7**.

sd-targets Map SCSI disk units (OpenBoot PROM version 1.*x* only). Defaults to **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.

security-mode Firmware security level (options: **none**, **command**, or **full**). If set to **command** or **full**, system will prompt for PROM security password. Defaults to **none**.

security-password Firmware security password (never displayed). Can be set only when **security-mode** is set to **command** or **full**.

```
example# eeprom security-password=
Changing PROM password:
New password:
Retype new password:
```

selftest-#megs Metabytes of RAM to test. Ignored if **diag-switch?** is **true**. Defaults to **1**.

skip-vme-loopback? If **true**, POST does not do VMEbus loopback tests. Defaults to **false**.

st-targets Map SCSI tape units (OpenBoot PROM version 1.*x* only). Defaults to **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 (>). Defaults to **false**.

testarea One-byte scratch field, available for read / write test. Defaults to **0**.

tpe-link-test? Enable 10baseT link test for built-in twisted pair Ethernet. Defaults to **true**.

ttya-mode TTYA (baud rate, #bits, parity, #stop, handshake). Defaults to **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). Defaults to **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. Defaults to **true**.
ttyb-ignore-cd	If **true**, operating system ignores carrier-detect on TTYA. Defaults to **true**.
ttya-rts-dtr-off	If **true**, operating system does not assert DTR and RTS on TTYA. Defaults to **false**.
ttyb-rts-dtr-off	If **true**, operating system does not assert DTR and RTS on TTYB. Defaults to **false**.
use-nvramrc?	If **true**, execute commands in NVRAMRC during system start-up. Defaults to **false**.
version2?	If **true**, hybrid (1.*x* / 2.*x*) PROM comes up in version 2.*x*. Defaults to **true**.
watchdog-reboot?	If **true**, reboot after watchdog reset. Defaults to **false**.

EXAMPLES

This 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
```

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

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

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**. To obtain *platform-name*, use **uname –i**.

SEE ALSO **passwd**(1), **sh**(1), **uname**(1TSOL)

NAME

fsdb_ufs – ufs file system debugger

SYNOPSIS

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

AVAILABILITY

SUNWcsu

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 overriden 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 etc. to a byte address. Once the address has been calculated, **fsdb** will record the result in **dot** (see next paragraph).

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 carriage-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 carriage-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, subsequent carriage-returns (or +, -) will advance to subsequent entries. Note that

> 2:inode; :ls

or

> :ls /

is again synonymous.

Expressions

The symbols recognized by **fsdb** are:

CARRIAGE-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 **Formatted Output** section).
/ *f*	display in unstructured style with format specifier *f* (see **Formatted Output** section).
.	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 (see **Formatted Output** section), then the value of **dot** is displayed using format specifier *f*. Otherwise, assignment is assumed (see next item).
= [*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 command 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 left off, 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.

shadow If the current **inode** is a shadow inode, then the value of **dot** is converted to a shadow data item index in that shadow and **dot** now points to this entry.

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.

! 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** sub-section).

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 **7**th 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.

sz file size.

si shadow inode number field in the current inode.

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 just displays raw data. The following table 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 data items

/

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**.

> 7:dir:nm="name" changes the name field in the directory slot to *name*.

> 2:db:block,*?d displays the third block of the current **inode** as directory entries.

> 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.

SEE ALSO

clri(1M), **fsck_ufs**(1M), **dir_ufs**(4), **fs_ufs**(4)

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

fuser – Identify processes using a file or file structure

SYNOPSIS

/usr/sbin/fuser [– [**c** | **f**] **ku**] *files* [[– [**c** | **f**] **ku**] *files*] . . .

AVAILABILITY

SUNWcsu

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, interpreted as follows: if the process is using the file as 1) its current directory, the code is **c**; 2) its root directory, the code is **r**; 3) an open file, the code is **o**; or 4) its text file, the code is **t**. 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 on), only the processes using that file are reported.

If more than one group of files is 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 on the standard output as a single line, separated by spaces and terminated with a single new line. All other output is written on standard error.

Appropriate privilege is needed to use this command.

OPTIONS

–**c** Report 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** Send the **SIGKILL** signal to each process. Because this option spawns kills for each process, the kill messages may not show up immediately. [See **kill**(2).]

–**u** Display the user login name in parentheses following the process ID.

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

ps(1), **mount**(1MTSOL), **kill**(2TSOL), **signal**(3C)

NOTES

Because it works with a snapshot of the system image, **fuser** 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

getauditpsa – display audit preselection attributes for a file

SYNOPSIS

getauditpsa *file* . . .

AVAILABILITY

Available only on Trusted Solaris 2.x systems

DESCRIPTION

For each argument that is a regular file, directory, multilevel directory, special file, or named pipe, **getauditpsa** displays the file audit preselection attributes.

This command may be executed on a filesystem that does not support file audit preselection attributes. It reports the file audit preselection attributes based on the filesystem mount options.

When multiple files are specified on the command line, a blank line separates the file audit preselection attributes for each file. The format of a file audit preselection attributes set is:

 # file: filename
user::success-audit:failure-audit:success-alarm:failure-alarm
user:uid:success-audit:failure-audit:success-alarm:failure-alarm
group::success-audit:failure-audit:success-alarm:failure-alarm
group:gid:success-audit:failure-audit:success-alarm:failure-alarm
other:success-audit:failure-audit:success-alarm:failure-alarm

The first line shows the filename.

The user entry without a user ID indicates the entry that is used when the owner of the file is accessing the file. One or more additional user entries indicate the entry that is used when the specified user is accessing the file. The group entry without a group ID indicates the entry that is used when the owning group of the file is accessing the file. One or more additional group entries indicate the entry that is used when the specified group is accessing the file. The other entry indicates the audit attributes that are used when others access the file.

The *uid* is a login name or a user ID if there is no entry for the *uid* in the system's password file. The *gid* is a group name or a group ID if there is no entry for the *gid* in the system's group file.

The *success-audit* field denotes which successful operations are audited. The *failure-audit* field denotes which failed operations are audited. The *success-alarm* field denotes which successful operations are alarmed. The *failure-alarm* field denotes which failed operations are alarmed.

The *success-audit, failure-audit, success-alarm*and *failure-alarm* fields are each 3 character strings composed of the letters [a | n | d][a | n | d][a | n | d]. The first character in each field is for read access, the second character in each field is for write access, and the third character in each field is for execute access. Each character position can have one of the following values:

[a] always

[n] never

[d] use process default

This allows for any [successful | failed] file [read | write | execute] accesses to be [audited | alarmed] to the granularity of a [owner | group owner | user | group | other].

The entries display in the order in which they are evaluated when an audit operation is performed.

EXAMPLES

Given file **foo**, with a file audit preselection attributes set five entries long, the command

example% getauditpsa foo

prints:

```
# file: foo
user::nan:aaa:ddd:ddd
user:spy:aaa:aaa:aaa:aaa
user:mookie:ddd:aaa:ddd:ddd
group::ddd:aaa:ddd:ddd
other::aaa:aaa:ddd:ddd
```

FILES

/etc/passwd
/etc/group

SEE ALSO

setauditpsa(1MTSOL), **auditf**(2TSOL), **apsasort**(3TSOL)

NOTE

The output from **getauditpsa** is in the correct format for input to the **setauditpsa** command. If the output from **getauditpsa** is redirected to a file, the file may be used as input to **setauditpsa**. In this way, a user may easily assign one file's entries to another file.

NAME

getfsattr – display file system security attributes

SYNOPSIS

getfsattr [**–a** | **–d** | **–f** | **–i** | **–I** | **–l** | **–L** | **–m** | **–p** | **–P** | **–s** | **–S**] *pathname*

AVAILABILITY

SUNWtsolu

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.

OPTIONS

–a Display the file system access ACL.

–d Display the file system default ACL.

–f Display the file system attribute flags.

–i Display the file system information label in short form.

–I Display the file system information label in long form.

–l Display the file system 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.

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**(2TSOL)

SEE ALSO

getfsattr(2TSOL)

NAME

halt, poweroff – stop the processor

SYNOPSIS

/usr/sbin/halt [–**lnqy**]

/usr/sbin/poweroff [–**lnqy**]

AVAILABILITY

SUNWcsu

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

–**l** Suppress sending a message to the system log daemon, **syslogd**(1M), about who executed **halt**.

–**n** Prevent the **sync**(1M) 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

SUMMARY OF TRUSTED SOLARIS CHANGES

This command requires the **sys_boot** privilege in order to run.

SEE ALSO

init(1M), **reboot**(1M), **shutdown**(1M), **sync**(1M), **syslogd**(1M)

NOTES

halt does not execute the **rc0** scripts as do **shutdown**(1M) and **init**(1M).

poweroff is equivalent to **init 5**.

NAME

hextoalabel – Convert a hexadecimal label to its ASCII coded equivalent

SYNOPSIS

/usr/sbin/hextoalabel [*hexadecimal_CMW_label*]

/usr/sbin/hextoalabel –**c** [*hexadecimal_clearance*]

/usr/sbin/hextoalabel –**i** [*hexadecimal_information_label*]

/usr/sbin/hextoalabel –**s** [*hexadecimal_sensitivity_label*]

AVAILABILITY

SUNWtsolu

DESCRIPTION

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

OPTIONS

–**c** Identifies the hexadecimal label as a clearance.

–**i** Identifies the hexadecimal label as an information label.

–**s** Identifies the hexadecimal label as a sensitivity label.

RETURN VALUES

Upon success, this command exits with **0**. Upon failure, this command exits with **1** and writes diagnostics to the standard error file.

FILES

/etc/security/tsol/label_encodings
The label encodings file containing the **CLASSIFICATIONS**, **WORDS**, constraints, and values for the defined labels of this system

SEE ALSO

label_encodings(4TSOL)

UNKNOWN TITLE ABBREVIATION: CMWTF

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**]

AVAILABILITY

SUNWcsu

DESCRIPTION

ifconfig is used to assign an address to a network interface and / or to configure network interface parameters. **ifconfig** must be used at boot time to define the network address of each interface present on a machine. **ifconfig** may also be used at a later time to redefine an interface's address or other operating parameters. Used without options, **ifconfig** displays the current configuration for a network interface. If a protocol family is specified, **ifconfig** reports the details specific to only that protocol family. **ifconfig** needs the **sys_net_config** privilege in order to modify the configuration of a network interface.

The *interface* parameter is 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**. Three special interface names, –**a,** –**ad,** and –**au,**are**reserved**and**refer**to 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.

–**au** Apply the commands to all "up" interfaces in the system.

Because 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 **inet** is assumed.

For the TCP / IP family (**inet**), the address is either a host name present in the host name database [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**.

If supplied in addition to the *address* parameter, the *dest_address* parameter specifies the address of the correspondent on the other end of a point-to-point link.

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.

arp Enable the use of the Address Resolution Protocol in mapping between network level addresses and link level addresses (default). This option is currently implemented for mapping between TCP/IP addresses and 10Mb/s Ethernet addresses.

–**arp** Disable the use of the Address Resolution Protocol.

auto-revarp Use the Reverse Address Resolution Protocol (RARP) to automatically acquire an address for this interface.

down Mark an interface "down."When an interface is marked "down," the system will not attempt to transmit messages through that interface. If possible, the interface will be reset to disable reception as well. This action does not automatically disable routes using the interface.

plumb Open the device associated with the physical interface name and setup the streams needed for TCP/IP to use the device. Before this operation 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 Tell the **in.routed** routing daemon that the interface should not be advertised.

–**private** Specify unadvertised interfaces.

trailers This flag formerly caused a nonstandard encapsulation of **inet** packets on certain link levels. Drivers supplied with this release no longer use this flag, but it is ignored for compatibility.

–**trailers** Disable the use of a "trailer" link-level encapsulation.

up Mark an interface "up." This marking happens automatically when setting the first address on an interface. The **up** option enables an interface after an **ifconfig down**,**reinitializing**the**hardware.**

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. Giving a **+** (plus sign) 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; and therefore,

ifconfig –a netmask + broadcast +

and

ifconfig –a broadcast + netmask +

may result the assignment of different values for the interfaces' broadcast addresses.

metric *n* Set the routing metric of the interface to *n*, default **0**. The routing metric is used by the routing protocol. Higher metrics make 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 subnetworks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. In the 32-bit address, the mask contains 1's for the bit positions that are to be used for the network and subnet parts, and 0's for those used 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: a) as a single hexadecimal number with a leading 0x. b) as a dot-notation address, c) as a '**+**' (plus sign) address, or d) as a pseudo host name / pseudo network name listed in the network table **networks** (4). If a '**+**' (plus sign) is given as the netmask value, the mask is looked up in the **netmasks database** using the interface network number as the key.

If a pseudo host name / pseudo network name is supplied as the netmask value, netmask data may be located in the **hosts** or **networks** table. **ifconfig** looks for the name in the **hosts** table first. If an entry is found, the host IP address is used as the netmask. If the entry isn't found there, **ifconfig** looks for the name in the **networks** table. (If the entry is found there, the IP network address is padded with a zero.) The **hosts** and **netmasks** tables are not designed for storage of netmasks; adding a netmask entry by using this option is inadvisable because doing so may confuse other programs. 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.

LOGICAL INTERFACES

Solaris TCP / IP allows associating multiple logical interfaces with a physical network interface. This association 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*; 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** subcommand, for example,

ifconfig le0 plumb

Once a physical interface has been "plumbed," additional local interfaces can be configured by simply naming them in subsequent **ifconfig** commands. Logical interfaces do not need to be "plumbed." Simply mentioning their names in an **ifconfig** command is sufficient. For example, the command

ifconfig le0:1

will allocate a logical interface associated with the physical interface le0.

A logical interface can be configured with parameters (address, netmask, etc.) different from the physical interface with which it is associated. And 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 be configured only after the physical interface **le0** has been plumbed.

EXAMPLES

If your workstation is not attached to an Ethernet, the **le0** interface should be marked "down" as follows:

example% ifconfig le0 down

To print 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 succeed. 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

SEE ALSO

in.routed(1MTSOL), **netstat**(1MTSOL), **ethers**(3N), **hosts**(4), **netmasks**(4), **networks**(4), **nsswitch.conf**(4), **arp**(7P)

DIAGNOSTICS

Messages indicate that the specified interface does not exist, the requested address is unknown, or the user is not privileged and tried to alter an interface's configuration.

NOTES

Avoid selecting **broadcast, down, private, trailers, up,** and the other possible option names when choosing host names. Choosing any one of these names as host names causes bizarre problems that can be extremely difficult to diagnose.

NAME

in.ftpd, ftpd – File-transfer protocol server

SYNOPSIS

in.ftpd [–**dl**] [–**t** *timeout*]

AVAILABILITY

SUNWcsu

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**(1M).
–**l**	Each FTP session is logged to the system log daemon **syslogd**(1M).
–**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 these 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	List files in a directory. (**ls** –**lg**)
MKD	Make a directory.
MODE	Specify data transfer mode.
NLST	List names 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 interpretation allows users to utilize these metacharacters: ∗ **? [ ] { } ~**

in.ftpd authenticates users according to five rules.

1) 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.
2) If the user name appears in the file **/etc/ftpusers**, **ftp** access is denied.
3) **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
4) 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 password 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 that entry a valid password (use NP in the encrypted password field of the shadow file).
5) 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**(2TSOL) 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. This directory should not be on a file system mounted with the **nosuid** option.

~ftp/bin Make this directory owned by the super-user 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 the super-user and unwritable by anyone. Copy these shared libraries from **/usr/lib** into this directory:

ld.so*
libc.so*
libdl.so*
libintl.so*
libw.so*
libnsl.so*
libsocket.so*
nss_nis.so*
nss_nisplus.so*
nss_dns.so*
nss_files.so*
straddr.so*

~ftp/etc Make this directory owned by the super-user 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 777 and owned by **ftp**. Users should then place in this directory files that are to be accessible via the anonymous account.

~ftp/dev Make this directory owned by the super-user and unwritable by anyone. First perform **ls –lL** on the device files listed 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 the super-user. Copy its

contents from **/usr/share/lib/zoneinfo**. This setup enables **ls –l** to display time and date stamps correctly.

EXAMPLES

To set up anonymous ftp, add the following entry to the **/etc/passwd** file. In this case, **/export/ftp** was chosen to be the anonymous FTP area, and the shell is the nonexistent file **/nosuchshell**. This setup prevents users from logging in as the **ftp** user.

ftp:x:30000:30000:Anonymous FTP:/export/ftp:/nosuchshell

Add the following entry to /etc/shadow:

ftp:NP:6445::::::

The following shell script will set up the anonymous FTP area. The shell script assumes that names are resolved using NIS.

```
#!/bin/sh
# script to setup anonymous ftp area
#
# handle the optional command line argument
case $# in

  # the default location for the anon ftp comes from the passwd file
  0) ftphome="`grep '^ftp:' /etc/passwd | cut -d: -f6`"
     ;;

  1) if [ "$1" = "start" ]; then
       ftphome="`grep '^ftp:' /etc/passwd | 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

# This script assumes that ftphome is neither / nor /usr so ...
if [ "${ftphome}" = "/" -o "${ftphome}" = "/usr" ]; then
  echo "$0: ftphome must not be / 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
echo Setting up anonymous ftp area ${ftphome}

# Ensure that the /usr/bin directory exists
if [ ! -d ${ftphome}/usr/bin ]; then
   mkdir -p ${ftphome}/usr/bin
fi

cp /usr/bin/ls ${ftphome}/usr/bin
chmod 111 ${ftphome}/usr/bin/ls

# 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
if [ -r ${ftphome}/bin ]; then
   mv -f ${ftphome}/bin ${ftphome}/Obin
fi
ln -s usr/bin ${ftphome}

# Ensure that the /usr/lib and /etc directories exist
if [ ! -d ${ftphome}/usr/lib ]; then
   mkdir -p ${ftphome}/usr/lib
fi
if [ ! -d ${ftphome}/etc ]; then
   mkdir -p ${ftphome}/etc
fi

#Most of the following are needed for basic operation, except
#for libnsl.so, nss_nis.so, libsocket.so, and straddr.so which are
#needed to resolve NIS names.

cp /usr/lib/ld.so /usr/lib/ld.so.1 ${ftphome}/usr/lib

for lib in libc libdl libintl libw libnsl libsocket \
```

```
  nss_nis nss_nisplus nss_dns nss_files
do
  cp /usr/lib/${lib}.so.1 ${ftphome}/usr/lib
  rm -f ${ftphome}/usr/lib/${lib}.so
  ln -s ./${lib}.so.1 ${ftphome}/usr/lib/${lib}.so
done

cp /usr/lib/straddr.so.2 ${ftphome}/usr/lib
rm -f ${ftphome}/usr/lib/straddr.so
ln -s ./straddr.so.2 ${ftphome}/usr/lib/straddr.so

cp /etc/passwd /etc/group /etc/netconfig ${ftphome}/etc

# 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) | cpio -imdu
  find . -print | xargs chmod 555
  find . -print | xargs chown root
)

chmod 555 ${ftphome}/usr/lib/*
chmod 444 ${ftphome}/etc/*

# Now set the ownership and modes
chown root ${ftphome}/usr/lib ${ftphome}/etc
chmod 555 ${ftphome}/usr/lib ${ftphome}/etc

# 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
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

if [ ! -d ${ftphome}/pub ]; then
  mkdir -p ${ftphome}/pub
fi
chown ftp ${ftphome}/pub
chmod 777 ${ftphome}/pub
```

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 login.

SEE ALSO

ftp(1), **ls**(1), **sh**(1), **aset**(1M), **inetd**(1MTSOL), **mknod**(1M), **syslogd**(1M), **chroot**(2TSOL), **getsockopt**(3NTSOL), **group**(4), **inetd.conf**(4TSOL), **netconfig**(4), **netrc**(4), **passwd**(4), **services**(4),

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**.

Info Severity

These messages are logged only if the **–l** flag is specified.

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 the user 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 account is inherently dangerous and should be avoided when possible.

The server must run as the super-user to create sockets with privileged port numbers. It maintains an effective user ID of the logged-in user, reverting to the super-user only when binding addresses to sockets. The possible security holes have been extensively scrutinized but are possibly 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

in.named, named, named-xfer – Internet domain name server

SYNOPSIS

in.named [–**b** *bootfile*] [–**d** *level*] [–**p** *port*]

named-xfer

AVAILABILITY

SUNWcsu

DESCRIPTION

in.named is the Internet domain name server. It is used by hosts on the Internet to provide access to the Internet distributed naming database. See RFC 1034 and RFC 1035 for more details. With no arguments, **in.named** reads **/etc/named.boot** for any initial data, and listens for queries on a privileged port.

named-xfer is called by **in.named** whenever **in.named** needs to perform a Zone Transfer. **named-xfer** should not be called independently.

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.

/etc/named.boot File Entries

The following is a sample of **/etc/named.boot** file entries:

```
;
;        boot file for name server
;
; type          domain          source file or host
;
domain          berkeley.edu
primary         berkeley.edu    named.db
secondary       cc.berkeley.edu 10.2.0.78 128.32.0.10
cache           .               named.ca
```

The **domain** line specifies that **berkeley.edu** is the domain of the given server.

The **primary** line states that the file **named.db** contains authoritative data for **berkeley.edu**. The file **named.db** contains data in the Zone file format, described in RFC 1035, except that all domain names are relative to the origin; in this case, **berkeley.edu** (see **Zone File Format** below for a more detailed description).

The **secondary** line specifies that all authoritative data under **cc.berkeley.edu** is to be transferred from the name server at **10.2.0.78**. If the transfer fails it will try **128.32.0.10**, and continue for up to ten tries at that address. The secondary copy is also authoritative for the domain.

The **cache** line specifies that data in **named.ca** is to be placed in the cache (typically such data as the locations of root domain servers). The file **named.ca** is in the same format as **named.db**.

Zone File Format

The Zone file consists of entries of the form:

$INCLUDE < *filename* >
$ORIGIN < *domain* >
< *domain* > < *opt_ttl* > < *opt_class* > < *type* > < *resource_record_data* >

where *domain* is "**.**" for the root, "**@**" for the current origin, or a standard domain name. If *domain* is a standard domain name that does not end with "**.**", the current origin is appended to the domain. Domain names ending with "**.**" are unmodified.

The *opt_ttl* field is an optional integer number for the time-to-live field. It defaults to zero.

The *opt_class* field is currently one token, **IN** for the Internet.

The *type* field is one of the following tokens; the data expected in the *resource_record_data* field is in parentheses.

A	A host address (dotted quad).
CNAME	The canonical name for an alias (domain).
HINFO	Host information (cpu_type OS_type).
MB	A mailbox domain name (domain).
MG	A mail group member (domain).
MINFO	Mailbox or mail list information (request_domain error_domain).
MR	A mail rename domain name (domain).
MX	A mail exchanger (domain).
NS	An authoritative name server (domain).
NULL	A null resource record (no format or data).
PTR	A domain name pointer (domain).
SOA	Marks the start of a zone of authority (5 numbers). See RFC 1035.
TXT	Arbitrary number of strings.
WKS	A well know service description (not implemented yet).

OPTIONS

–b *bootfile*	Use *bootfile* rather than **/etc/named.boot**.
–d *level*	Print debugging information. *level* is a number indicating the level of messages printed.
–p *port*	Use a different *port* number.

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_nofloat**, **proc_setclr**, **sys_trans_label**, **sys_net_config**, and **sys_config.**

FILES

/etc/named.boot	name server configuration boot file
/etc/named.pid	the process ID
/var/tmp/named.run	debug output
/var/tmp/named_dump.db	dump of the name servers database

These files have a sensitivity label of ADMIN_LOW.

SEE ALSO

kill(1), **resolver**(3NTSOL), **signal**(3B), **resolv.conf**(4)

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.

NOTES

The following signals have the specified effect when sent to the server process using the **kill**(1) command.

SIGHUP	Reads **/etc/named.boot** and reloads database.
SIGINT	Dumps the current database and cache to **/var/tmp/named_dump.db**.
SIGUSR1	Turns on debugging; each subsequent SIGUSR1 increments debug level.
SIGUSR2	Turns off debugging completely.

NAME

in.rarpd, rarpd – DARPA Reverse Address Resolution Protocol server

SYNOPSIS

/usr/sbin/in.rarpd [–**d**] –**a**

/usr/sbin/in.rarpd [–**d**] *device unit*

AVAILABILITY

SUNWcsu

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 run as root.

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

–**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

The following command starts an **in.rarpd** for each network interface name returned from **/dev/ip**:

example# /usr/sbin/in.rarpd –a

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

FILES

/etc/ethers	file or NIS+ map
/etc/hosts	file or NIS+ map
/tftpboot	
/dev/ip	
/dev/arp	

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.

SEE ALSO

boot(1M), **ifconfig**(1MTSOL), **ethers**(4), **hosts**(4), **netconfig**(4), **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.rexecd, rexecd – remote execution server

SYNOPSIS

in.rexecd

AVAILABILITY

SUNWcsu

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. **in.rexecd** is invoked automatically as needed by **inetd**(1MTSOL), 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 it is not zero, the number received in Step 1 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 fails, 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 login.

SEE ALSO

inetd(1MTSOL), **inetd.conf**(4TSOL), **rexec**(3N)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the **stderr**; then any network connections are closed. An error is indicated by a leading byte with a value of **1**. (**0** is returned in Step 7 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.
Try again.	A **fork** by the server 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

AVAILABILITY

SUNWcsu

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**(1MTSOL) when a remote login connection is established, 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.

2) 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** option. The login process then proceeds with the authentication process as described in **in.rshd**(1MTSOL); but if automatic authentication fails, the process reprompts the user to login.

The **-U** option is used to pass the UID of the client to **login**(1) and 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/ 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).

SEE ALSO

login(1)C , **rlogin**(1TSOL)C , **in.rshd**(1MTSOL)C , **inetd**(1MTSOL)C , **inetd.conf**(4TSOL)C , **environ**(4)C , **hosts**(4)C , **hosts.equiv**(4)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the **stderr**; then 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 assumption is insecure but is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

NAME

in.rshd, rshd – Remote shell server

SYNOPSIS

in.rshd *host.port*

AVAILABILITY

SUNWcsu

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**(1MTSOL) 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 it is not zero, the number received in Step 1 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.

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, the process does a **chdir** to root (*/*). If the user is not the super-user, (user ID 0), 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. If the lookup fails, or if the user is the super-user, 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**. The values of the trusted path, label view, and label-translation process attributes from the client process are propagated to the shell by **in.rshd**.

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

SEE ALSO

rsh(1), **inetd**(1MTSOL), **inetd.conf**(4TSOL), **hosts**(4)

DIAGNOSTICS

The following diagnostic messages are returned on the connection associated with **stderr**; then any network connections are closed. An error is indicated by a leading byte with a value of **1** in Step 9. (**0** is returned 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 earlier 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 assumption is insecure but is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

NAME

in.tftpd, tftpd – Internet Trivial File Transfer Protocol server

SYNOPSIS

in.tftpd [–**s**] [*homedir*]

AVAILABILITY

SUNWcsu

DESCRIPTION

tftpd is a server that supports the Internet Trivial File Transfer Protocol (TFTP). This server is normally started by **inetd**(1MTSOL) 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**(4TSOL).

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*.

FILES

/etc/inetd.conf

SUMMARY OF TRUSTED SOLARIS CHANGES

in.tftpd should be started from the Trusted Path with a UID 0; it must inherit the **proc_chroot**, **proc_owner**, and **proc_setid** privileges.

SEE ALSO

tftp(1), **inetd**(1MTSOL), **netconfig**(4)

Sollins, K.R., *The TFTP Protocol (Revision 2)*, RFC 783, Network Information Center, SRI International, Menlo Park, Calif., June 1981.

NAME

inetd – Internet services daemon

SYNOPSIS

inetd [–**d**] [–**s**] [–**t**] [–**r** *count interval*] [*configuration-file*]

AVAILABILITY

SUNWcsu

DESCRIPTION

inetd is the server process for the Internet standard services. **inetd** 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**(4TSOL) 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 services listed in the configuration file. When a request arrives, **inetd** executes the server program associated with the service. An **inetd** profile entry is now contained in **tsolprof**. This entry defines the privileges and minimum and maximum labels for servers started by **inetd**.

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 reads its configuration file and the **inetd** profile entry in **tsolprof** first when **inetd** 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 or the **inetd** profile, and then sending **inetd** a **SIGHUP** signal.

OPTIONS

–**d** Runs **inetd** in the foreground and enables debugging output.

–**s** Allows you to run **inetd** "stand-alone," 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 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 cannot 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" servers. The –**r** flag has this 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 behaves 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.

RETURN VALUES

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**(4TSOL). See that man page for more detail.

inetd registers RPC servers as multilevel servers with **rpcbind**.

The time, discard, daytime, and chargen internal services get replies with an ADMIN_LOW information label. The echo-service replies have an information label equal to the information label of the data received from the client.

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.

SEE ALSO

in.ftpd(1MTSOL), **in.rexecd**(1MTSOL), **in.rshd**(1MTSOL), **in.tftpd**(1MTSOL), **sac**(1M), **inetd.conf**(4TSOL)

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 configuration would cause a race condition in which the **inetd** program selects on the socket and the server program reads from the socket. Many server programs would be forked and performance would 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

init, telinit – process control initialization

SYNOPSIS

/sbin/init [**0123456abcQqSs**]

/etc/telinit [**0123456abcQqSs**]

AVAILABILITY

SUNWcsu

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.

OPTIONS

0	Go into firmware.
1	Put the system in system administrator mode. All 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. When this occurs, the terminal that executed this command becomes the system console. 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 the system comes up to **S** or **s**, file systems for users' files are not mounted and only essential kernel processes are running. When the system comes down to **S** or **s**, all mounted file systems remain mounted, and all processes started by **init** 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. Other processes not started directly by **init** will remain running. For example, **cron** remains running.

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	
/dev/console	system console device
/etc/default/init	environment variables.

SEE ALSO

login(1), **sh**(1), **stty**(1), **who**(1), **shutdown**(1M), **su**(1M), **ttymon**(1M), **ioctl**(2), **kill**(2), **ctime**(3C), **inittab**(4), **utmp**(4), **utmpx**(4), **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

install_scripts, add_install_client, rm_install_client, setup_install_server, check – scripts used to install the Solaris software

SYNOPSIS

cdrom-mnt-pt/**add_install_client** [−**i** *IP_address*] [−**e** *Ethernet_address*] [−**s** *server_name*:*path*] [−**c** *server_name*:*path*] [−**T** *server_name*:*path*] [−**n** [*server*]:*name_service*[(*netmask*)]] *host_name platform_name*

cdrom-mnt-pt/**rm_install_client** *host_name*

cdrom-mnt-pt/**setup_install_server** [−**b**] *install_dir_path*

cdrom-mnt-pt/**check** [−**p**] [−**r** *rulesfile*] *install_dir_path*

AVAILABILITY

SUNWcdrom (Solaris CD)

DESCRIPTION

These commands are located on the Solaris CD in the **/cdrom/cdrom0/s0** directory. (If the Solaris CD has been copied to a local disk, these scripts will be in the path to that directory.) They can be used for a variety of installation tasks. Specifically,

- Use **add_install_client** and **rm_install_client** to add or remove clients for network installation.
- Use **setup_install_server** to copy the Solaris CD to a disk or to copy just the boot software of the Solaris CD to a disk (for instance, to set up a boot server).
- Use **check** to validate the rules in a **rules** file (this is only necessary if a custom JumpStart installation is being set up).

OPTIONS

The following options are supported:

add_install_client

−**i** *IP_address* Specify the IP address of the client to be installed.

−**e** *Ethernet_address* Specify the Ethernet address of the system to be installed.

−**s** *server_name:path* This option is required only when using **add_install_client** from a boot server. Specify the name of the server and the absolute path of the Solaris installation image that will be used for this installation. *path* is either the path to a mounted Solaris CD or a path to a directory with a copy of the Solaris CD.

−**c** *server_name*:*path* This option is required only to specify a JumpStart directory for a custom JumpStart installation. *server_name* is the host name of the server with a JumpStart directory. *path* is the absolute path to the JumpStart directory.

−**T** *server_name*:*path* This option is used to specify a Trusted Solaris configuration directory for a network or JumpStart installation. *server_name* is the host name of the server with a Trusted Solaris configuration directory. *path* is the absolute path to the Trusted Solaris configuration directory.

−**n** [*server]:name_service*[(*netmask*)]

This option specifies which name service should be used during system configuration. This sets the **'ns' bootparams**(4) keyword.

name_service
Valid entries are '**nis**', '**nisplus**', and '**none**'.

server The name of the server or IP address of the specified name service. If the server specified is on a different subnet, then the netmask may be needed to enable to client to contact the server.

netmask A series of four numbers separated by periods, specifying which portion of an IP address is the network part, and which is the host part.

rm_install_client

–b This option sets up the server only as a boot server.

check

–p *install_dir_path* Specifies the absolute path to the Solaris installation image (either the mounted Solaris CD-ROM or a copy of the Solaris CD-ROM on the local disk). Using this option ensures that the most recent check program is being used to validate the **rules** file.

–r *rulesfile* Specifies a rules file other than the one named rules. Using this option, the validity of a rule can be tested before integrating it into the rules file. **check** will report whether or not the rule is valid, but it will not create the **rules.ok** file necessary for a custom JumpStart installation.

OPERANDS

The following operands are supported:

add_install_client

host_name This is the name of the client to be installed.

platform_name Vendor-defined grouping of hardware platforms for the purpose of distributing specific software. Examples of valid platform names are:

System	Platform Name
x86	**i86pc**
SPARCstation 1+	**sun4c**
SPARCstation LX	**sun4c**
PowerPC	**prep**

rm_install_client

host_name This is the name of the client to be removed.

platform_name Vendor-defined grouping of hardware platforms for the purpose of distributing specific software. (See previous list for valid arguments.)

setup_install_server

install_dir_path	Specify the absolute path of the directory in which the Solaris software is to be copied.
platform_name	Vendor-defined grouping of hardware platforms for the purpose of distributing specific software.

EXAMPLES

The following **add_install_client** commands add clients for network installation from a mounted Solaris CD on an install server.

example# cd /cdrom/cdrom0/s0

example# ./add_install_client system_1 sun4c

example# ./add_install_client system_2 i86pc

The following **add_install_client** commands add clients for network installation from a mounted Solaris CD on an install server. The **–c** option specifies a server and path to a JumpStart directory that has a rules and profile files for performing a custom JumpStart installation.

example# cd /cdrom/cdrom0/s0

example# ./add_install_client –c install_server:/jumpstart system_1 sun4c

example# ./add_install_client –c install_server:/jumpstart system_2 i86pc

The following **rm_install_client** commands remove system information about the named clients on the install server.

example# cd /cdrom/cdrom0/s0

example# ./rm_install_client holmes

example# ./rm_install_client watson

The following **setup_install_server** command copies the mounted Solaris CD to a directory named /export/install on the local disk. (This requires approximately 200 Mbytes of disk space.)

example# cd /cdrom/cdrom0/s0

example# ./setup_install_server /export/install

The following **setup_install_server** command copies the boot software of a mounted Solaris CD to a directory named /boot_dir on system that is going to be a boot server for a subnet. The command must be entered once for each client architecture to be installed on the subnet.

example# cd /cdrom/cdrom0/s0

example# ./setup_install_server –b

example# ./setup_install_server –b

The following **check** command validates the syntax of the rules file used for a custom JumpStart installation.

example# cd *jumpstart_dir_path*

example# ./check –p /cdrom/cdrom0/s0

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 An error has occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**T** option is added to the **add_install_client** script to allow the specifying of Trusted Solaris configuration information during network or JumpStart installations. The other scripts are unaltered.

SEE ALSO

bootparams(4)

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*]

AVAILABILITY

SUNWcsu

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.

FILES

/etc/security/device_allocate
/etc/security/device_maps
/etc/security/dev/*
/usr/security/lib/*

SEE ALSO

allocate(1MTSOL), **deallocate**(1MTSOL), **device_allocate**(4TSOL), **device_maps**(4TSOL)

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*]]

AVAILABILITY

SUNWlpu

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*, *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 a 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* (see **lpstat**(1TSOL)). The LP print service does not interpret this comment.

–e *printer*$_1$
Copy the interface program of an existing *printer*$_1$ 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_8	move back one character, except at beginning of line
tab	11_8	move to next tab stop
linefeed (newline)	12_8	move to beginning of next line
form feed	14_8	move to beginning of next page
carriage return	15_8	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*

Each **–o** *option* in the list below is the default given to an interface program if the option is not taken from a preprinted form description or is not explicitly given by the user submitting a request (see **lp**(1TSOL)). The only **–o** *options* that can have defaults defined are as follows:

length=*scaled-decimal-number*
width=*scaled-decimal-number*
cpi=*scaled-decimal-number*
lpi=*scaled-decimal-number*
stty=*'stty-option-list'*

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 first four default 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 *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.

For each *option* not specified, the defaults for the following attributes are defined in the **terminfo** entry for the specified printer type.

length
width
cpi
lpi

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'

You can set any of the **–o** options to the default values (which vary for different types of printers), by typing them without assigned values, as follows:

length=
width=
cpi=
lpi=
stty=

–o nobanner
Allow a user to submit a print request specifying that no banner page be printed.

–o 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; you must specify **–o nobanner** if you want to allow users to be able to specify **–o nobanner** with the **lp** command.

–P *paper-name*
Specifiy 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 must be listed in the systems table (**/etc/lp/Systems**). *printer-name* is the name used on the remote system for that printer. For example, if you want to access *printer* $_1$ on *system* $_1$ and you want it called *printer* $_2$ on your system:

–**p** *printer* $_2$ –**s** *system* $_1$!*printer* $_1$

–**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 **(***integer*$_1$ **requests)**
integer$_2$ **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 *integer*$_1$ listed next to each printer is the number of requests eligible for the printer. The number *integer*$_2$ 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **lpadmin** command requires the **administer printing** authorization.

FILES

/var/spool/lp/*
/etc/lp
/etc/lp/alerts/printer fault handler for **lpadmin**.

SEE ALSO

enable(1TSOL), **lp**(1TSOL), **lpstat**(1TSOL), **stty**(1), **accept**(1MTSOL), **lpsched**(1MTSOL), **lpsystem**(1MTSOL), **dial**(3N), **terminfo**(4)

UNKNOWN TITLE ABBREVIATION: SYSADMIN2, UNKNOWN TITLE ABBREVIATION: TSOLADMINTASK,

NAME

lpfilter – administer filters used with the LP print service

SYNOPSIS

/usr/sbin/lpfilter –**f** *filter-name* –**F** *path-name*
/usr/sbin/lpfilter –**f** *filter-name* – | –**i** | –**x** | –**l**

AVAILABILITY

SUNWlpu

DESCRIPTION

The **lpfilter** command is used to add, change, delete, and list a filter used with the LP print service. These filters convert the content type of a file to a content type acceptable to a printer.

The argument **all** can be used instead of a *filter-name* with any of these options. When **all** is specified with the –**F** or – option, the requested change is made to all filters. Using **all** with the –**i** option has the effect of restoring to their original settings all filters for which predefined settings were initially available. Using the **all** argument with the –**x** option results in all filters being deleted, and using it with the –**l** option produces a list of all filters.

OPTIONS

–**F** *path-name*	To add or change a filter. (–**F** *path-name* or – for standard input).
–**f** *filter-name*	Adds *filter-name* to the filter table.
–**i**	To reset an original filter to its factory setting.
–**x**	To delete a filter.
–**l**	To list a filter description.

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.

The filter description is taken from the *path-name* if the –**F** option is given, or from standard input if the – option is given. One of the two must be given to define or change a filter. If the filter named is one originally delivered with the LP print service, the –**i** option will restore the original filter description.

When an existing filter is changed with the –**F** or – option, items that are not specified in the new information are left as they were. When a new filter is added with this command, unspecified items are given default values. (See below.)

Filters are used to convert the content of a request into a data stream acceptable to a printer. For a given print request, the LP print service will know the following:

- the type of content in the request,
- the name of the printer,
- the type of the printer,
- the types of content acceptable to the printer, and
- the modes of printing asked for by the originator of the request.

It uses this information to find a filter or a pipeline of filters that will convert the content into a type acceptable to the printer.

Below is a list of items that provide input to this command, and a description of each item. All lists are comma or space separated.

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 types of content that can be accepted by the filter. (The default is **any**.)

Output types This gives the types of content that the filter can produce from any of the input content types. (The default is **any**.)

Printer types This gives the type of printers for which the filter can be used. The LP print service will restrict the use of the filter to these types of printers. (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. They are run while unconnected to 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**. Fast filters are generally those that convert their input quickly, or those that must be connected to the printer when run. These 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.

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).

lp Option	Characteristic	*keyword*	Possible *pattern*s
–T	Content type (input)	INPUT	*content-type*
N / A	Content type (output)	OUTPUT	*content-type*
N / A	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	CHARSET	*character-set-name*
	Print wheel	CHARSET	*print-wheel-name*
–f	Form name	FORM	*form-name*
–y	Modes	MODES	*mode*
–n	Number of copies	COPIES	*integer*

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* **–w\1** 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.

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*
Printers: *printer-list*
Filter type: *filter-type*
Command: *shell-command*
Options: *template-list*

If the command fails, an error message is sent to standard error.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **lpfilter** command requires the **administer printing** authorization.

SEE ALSO

lp(1TSOL), **lpadmin**(1MTSOL), **regexp**(5)

UNKNOWN TITLE ABBREVIATION: SYSADMIN2, Trusted Solaris Administrator's Procedures

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*]

AVAILABILITY

SUNWlpu

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**(1TSOL)). 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 *option* (–, –**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–number*$_1$
Page width: *scaled–decimal–number*$_2$
Number of pages: *integer*
Line pitch: *scaled–decimal–number*$_3$
Character pitch: *scaled–decimal–number*$_4$
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–number*$_1$ long, and *scaled–decimal–number*$_2$ 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. $scaled\text{-}decimal\text{-}number_3$ is in lines-per-centimeter if a **c** is appended, and lines-per-inch otherwise; similarly, $scaled\text{-}decimal\text{-}number_4$ 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**(1TSOL)).

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**(1TSOL) 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 **(***integer*$_1$ **requests).**
integer$_2$ **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 *integer*$_1$ listed next to each printer is the number of requests eligible for the printer. The number *integer*$_2$ 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **lpforms** command requires the **administer printing** authorization.

FILES

/etc/lp/alerts/form fault handler for **lpform**.

SEE ALSO

lp(1TSOL), **lpadmin**(1MTSOL), **lpstat**(1TSOL), **terminfo**(4)

UNKNOWN TITLE ABBREVIATION: SYSADMIN2, Trusted Solaris Administrator's Procedures

NAME

lpsched, lpshut, lpmove – start / stop the LP print service and move requests

SYNOPSIS

/usr/lib/lp/lpsched
lpshut
lpmove *requests dest*
lpmove *dest*$_1$ *dest*$_2$

AVAILABILITY

SUNWlpu

DESCRIPTION

lpsched starts the LP print service; this command must be run from the Trusted Path.

If the –**X** option is specified, **lpsched** does not perform discretionary access control checks on print-queue-listing requests from **lpstat** and **lpq**. **lpsched** must inherit these privileges: **file_chown**, **file_dac_read**, **file_dac_write**, **file_dac_search**, **file_mac_read**, **file_mac_search**, **file_mac_write**, **file_owner**, **file_setdac**, **file_setid**, **net_downgrade_sl**, **net_upgrade_il**, **net_setpriv**, **net_setid**, **proc_setclr**, **proc_setsl**, **proc_setid**, **proc_audit_tcb**, **proc_nofloat**, **proc_owner**, **proc_mac_write**, **sys_trans_label**.

lpshut shuts down the print service. All printers that are printing at the time **lpshut** is invoked will stop printing. When **lpsched** is started again, requests that were printing at the time a printer was shut down will be reprinted from the beginning.

lpshut requires the **administer printing** authorization.

lpmove moves requests that were queued by **lp**(1) between LP destinations. **lpmove** can only be used among local printers. **lpmove** requires the **administer printing** authorization.

The first form of the **lpmove** command moves the named *requests* to the LP destination *dest*. *requests* are request-IDs as returned by **lp**. If a request was originally queued for a class, or the special destination **any**, the destination of the request will be changed to *dest*. The request will be printable only on *dest* and not on other members of the **class** or other acceptable printers.

The second form of the **lpmove** command attempts to move all requests for destination *dest*$_1$ to destination *dest*$_2$; **lp** will then reject any new requests for *dest*$_1$.

Note that when moving requests, **lpmove** never checks the acceptance status of the new destination (see **accept**(1MTSOL)). Also, the request-IDs of the moved request are not changed, so that users can still find their requests. The **lpmove** command will not move requests that have options (content type, form required, and so on) that cannot be handled by the new destination.

FILES

/var/spool/lp/*

SUMMARY OF TRUSTED SOLARIS CHANGES

lpsched must be started from the Trusted Path and must inherit appropriate privileges. The –**X** option turns off discretionary access controls on print queue displays.

SEE ALSO

enable(1TSOL), **lp**(1TSOL), **lpstat**(1TSOL), **accept**(1MTSOL), **lpadmin**(1MTSOL)
UNKNOWN TITLE ABBREVIATION: SYSADMIN2, Trusted Solaris Administrator's Procedures

NAME

lpsystem – register remote systems with the print service

SYNOPSIS

lpsystem [–**t** *type*] [–**T** *timeout*] [–**R** *retry*] [–**y** "*comment*"] *system-name* [*system-name* ...]
lpsystem –**l** [*system-name* ...]
lpsystem –**r** *system-name* [*system-name* ...]
lpsystem –**A**

AVAILABILITY

SUNWlpu

DESCRIPTION

The **lpsystem** command is used to define parameters for the LP print service, with respect to communication with remote systems. Use of the **lpsystem** command requires the **administer printing** authorization.

Specifically, the **lpsystem** command is used to define remote systems with which the local LP print service can exchange print requests. These remote systems are described to the local LP print service in terms of several parameters that control communication: type, retry and timeout. These parameters are defined in **/etc/lp/Systems**. You can edit this file with a text editor (such as **vi**) but editing is not recommended.

The *type* parameter defines the remote system as one of two types: **s5** (SunOS 5.x operating system), or **bsd**. The default type is **s5**.

The *timeout* parameter specifies the length of time (in minutes) that the print service should allow a network connection to be idle. If the connection to the remote system is idle (that is, there is no network traffic) for *N* minutes, then drop the connection. (When there is more work the connection will be re-established.) Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0. The value **n** means ‘‘never time out’’; **0** means ‘‘as soon as the connection is idle, drop it.’’ The default is **n**.

The *retry* parameter specifies the length of time to wait before trying to re-establish a connection to the remote system, when the connection was dropped abnormally (that is, a network error). Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0 and it means ‘‘wait *N* minutes before trying to reconnect. (The default is 10 minutes.) The value **n** means ‘‘do not retry dropped connections until there is more work’’; **0** means ‘‘try to reconnect immediately.’’

The *comment* argument allows you to associate a free form comment with the system entry. This is visible when **lpsystem** –**l** is used.

system-name is the name of the remote system from which you want to be able to receive jobs, and to which you want to be able to send jobs. If the *system-name* is a plus sign ("+"), then anonymous client support is enabled. That is, your system will accept remote print jobs from any other print client (**bsd** or **s5**). This is enabled by default in **/etc/lp/Systems**; any other entries in the **/etc/lp/Systems** file will be superfluous. The other parameters listed on the line beginning with the *plus sign* are for reference only, and will not actually change the behavior of **lpsched**(1MTSOL).

The command **lpsystem** –**l** [*system-name*] will print out a description of the parameters associated with *system-name* (if a system has been specified), or with all the systems in its database (if *system-name* has not been specified).

The command **lpsystem** –**r** *system-name* will remove the entry associated with *system-name*. The print service will no longer accept jobs from that system or send jobs to it, even if the remote printer is still defined on the local system.

The command **lpsystem** –**A** will print out the TCP/IP address of the local machine in a format to be used when configuring the local port monitor to accept requests from a SunOS system.

OPTIONS

–**t** *type*	Specifies the remote system type.
–**T** *timeout*	Specifies the time allowed for a network connection to be idle. *timeout* is in minutes. Default is to never time out.
–**R** *retry*	Specifies time to wait before trying to reestablish a connection for a remote system.
–**y** *comment*	The comment argument allows you to associate a free form comment with the system entry.
–**l** *[system-name]*	Prints out a description of the parameters associated with *system-name*, or with all the systems in its database.
–**r** *system-name*	Removes the entry associated with *system-name*.
–**A**	Prints out the TCP/IP address in a format.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **lpsystem** command requires the **administer printing** authorization

FILES

/var/spool/lp/∗ **/etc/lp/**∗

SEE ALSO

lpsched(1MTSOL), **nlsadmin**(1M), **sacadm**(1M), **netdir**(3N), **hosts**(4), **netconfig**(4), **services**(4)

Solaris Naming Administration Guide
UNKNOWN TITLE ABBREVIATION: SYSADMIN2, Trusted Solaris Administrator's Procedures

NOTES

With respect to **/etc/lp/Systems**, this information is relatively minimal with respect to controlling network communications. Network addresses and services are handled by the **Netconfig** and **Netdir** facilities (see the *Solaris Naming Administration Guide* for a discussion of network addresses and services.) Port monitors handle listening for remote service requests and routing the connection to the print service (see the *Solaris 1.x to 2.x Transition Guide* for a discussion of port monitors.)

If the **Netconfig** and **Netdir** facilities are not set up properly, out-bound remote print service probably will not work. Similarly, if the local port monitors are not set up to route remote print requests to the print service, then service for remote systems will not be provided. See the chapters on managing printers in the *UNKNOWN TITLE ABBREVIATION: SYSADMIN2* for instructions.

With respect to the semantics of the *timeout* and *retry* values, the print service uses one process for each remote system with which it communicates, and it communicates with a remote system only when there is work to be done on that system or work being sent from that system.

The system initiating the connection is the ''master'' process and the system accepting the connection is the ''slave'' process. This designation serves only to determine which process dies (the slave) when a connection is dropped. This helps prevent there from being more than one process communicating with a remote system. Furthermore, all connections are bi-directional, regardless of the master / slave designation. You cannot control a system's master / slave designation. Now, keeping all this information in mind, if a master process times out, then both the slave and master will exit. If a slave times out, then it is possible that the master may still live and retry the connection after the retry interval. Therefore, one system's resource management strategy can effect another system's strategy.

With respect to **lpsystem** **–A**: a SunOS 4.x system (described with **–t bsd**) can be connected to your system only via TCP / IP, and print requests from a SunOS system can come in to your machine only via a special port (515). The address given to you from **lpsystem** will be the address of your system and port 515. This address is used by your TCP / IP port monitor (see **sacadm**(1M) and **nlsadmin**(1M)) to ''listen'' on that address and port, and to route connections to the print service. (This procedure is discussed in the *UNKNOWN TITLE ABBREVIATION: SYSADMIN2*.) The important point here is that this is where you get the address referred to in that procedure.

The command **lpsystem** **–A** will not work if your system name and IP address are not listed in **/etc/inet/hosts**, (see **hosts**(4)), and the printer service is not listed in **/etc/inet/services**, (see **services**(4)).

The file **/etc/lp/Systems** is set by default to support anonymous print clients. This feature can be disabled if one wishes greater security for print jobs. However, it should be noted that this will increase the amount of work required of the system administrator. A good backup of this file is strongly recommended if anonymous print client support is disabled.

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**

AVAILABILITY

SUNWlps

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**(1TSOL)), 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **lpusers** command requires the **administer printing** authorization.

SEE ALSO

lp(1TSOL)

NAME

modload – Load a kernel module

SYNOPSIS

modload [–**p**] [–**e** *exec_file*] *filename*

AVAILABILITY

SUNWcsu

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 path name, then the file specified by that absolute path is loaded. If *filename* does not begin with a slash (/), 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

then the kernel will look for **./drv/foo**. If you type

example# modload –p drv/foo

then 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. This information is module-specific: 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.

SEE ALSO

ld(1), **add_drv**(1MTSOL), **kernel**(1M), **modinfo**(1M), **modunload**(1MTSOL), **system**(4), **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), not **modload**, to add device drivers. See *Writing Device Drivers* for procedures on adding device drivers.

NAME

modunload – Unload a module

SYNOPSIS

modunload –**i** *module_id* [–**e** *exec_file*]

AVAILABILITY

SUNWcsu

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**, **modunload** unloads all modules that were autoloaded and are unloadable. Modules loaded by **modload**(1MTSOL) 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). Two additional arguments 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 file systems, 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 need the **sys_devices** privilege.

SEE ALSO

modinfo(1M), **modload**(1MTSOL)

NAME

mount, umount – mount or unmount file systems and remote resources

SYNOPSIS

mount [–**p** | –**v**]
mount [–**F** *FSType*] [*generic_options*] [–**o** *specific_options*] [–**O**] [–**S** *attribute_list*] *special* | *mount_point*
mount [–**F** *FSType*] [*generic_options*] [–**o** *specific_options*] [–**O**] [–**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. . .*]

AVAILABILITY

SUNWcsr

DESCRIPTION

mount attaches a file system to the file-system hierarchy at the *mount_point*, which is the path name 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 *FSType*, *special,* or *mount_point* are missing, **mount** searches **/etc/vfstab** to fill in the missing arguments. 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* and *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 slash (/), 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 may be used to specify mount time security attributes. If the –**S** option is not used, **mount** also searches **/etc/security/tsol/vfstab_adjunct** for any security attributes that may be specified there to be associated with the file system being mounted.

Note: File system *objects* (files and directories) in the file system being mounted may have security attributes of their own. For example, almost all file system objects have a UID, GID, and mode, and may have an access ACL and default ACL, while objects in file systems that have the **tsol_attr** flag set, in addition to the base set of attributes, have a sensitivity label and an information label and may have a file attribute flag, forced and allowed privileges. In addition to any security attributes on objects, the *file system* itself

may be assigned values for security attributes either while the file system is being created using **newsecfs**(1MTSOL) or by using **setfsattr**(1MTSOL) after file system creation to set or change any of a file system's attributes. Security attributes may be specified at mount time, either with the **–S** option on the command line or in the **vfstab_adjunct**(4TSOL) file to override filesystem-wide security attributes. However, mount time attributes never override security attributes on the files and directories. When access control decisions are made, any security attributes on a file or directory always take precedence over security attributes specified either at the filesystem level or at mount time. For any other attributes not obtainable either from the object, or at mount time, or from the file system, values of **none** or **empty** are used.

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**. 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**(2TSOL). 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

–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 unmount 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 file systems 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**.

–r Mount the file system read-only.

–o Specify *FSType*-specific options in a comma-separated list (without spaces) 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."

–S *attribute_list* Specify in *attribute_list* a semicolon-separated list of security attributes to associate with the file-system mount, which may optionally be used to override any attributes specified for the underlying file system but not to override any attributes obtainable from the files and directories within the file system. The optional attributes consist of access ACL, default ACL, mode, user ID, group ID, information label, sensitivity label, forced privileges, allowed privileges, a file attribute flag, label range, and multilevel directory prefix.

Each attribute is specified with a value assigned to a keyword in semicolon-separated fields. The keyword fields follow the format:

keyword=value

where *keyword* is one of the following:

acc_acl Sets the access ACL on all files or directories in the file system. See **setfacl**(1) for the format.

def_acl Sets a default ACL on all directories in the file system, so that any new files created in those directories are given an access ACL equal to the default ACL. See **setfacl**(1) for the format.

mode Sets a file system access mode. Use a five digit octal number.

attr_flg Sets an attribute flag that applies 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 events for either MAC (AUE_MAC) or use of privilege (AUE_UPRIV) are included in a preselected audit class and if the operation involves the preselected event (either MAC or 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), **fstatfs**(2), **getcmwfsrange**(2), **getcmwlabel**(2), **getfpriv**(2), **getmldadorn**(2), **getsldname**(2), **lgetcmwlabel**(2), **lstat**(2), **mldlstat**(2), **mldstat**(2), **open**(2)—read only, **pathconf**(2), **readlink**(2), **stat**(2), **statfs**(2), **statvfs**(2). See *Trusted Solaris Audit Administration Manual* 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 useful only if the type of file system being mounted does not have GIDs on its files or directories.)

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 useful only if the type of file system being mounted does not have UIDs on its files or directories.)

ilabel Sets the information label for all objects in the file system. Specify the information label in ASCII format.

slabel Sets the sensitivity label for all objects in the file system. Specify the sensitivity label in ASCII format.

forced Sets forced privileges for all executable files in the file system. Specify privilege names in a comma-separated ASCII list (such as: **forced=file_audit, file_chown;**) or use **all** to indicate all privileges. See **priv_desc**(4TSOL). Note: The forced privileges must be a subset of the allowed privileges.

allowed Sets allowed privileges for all executable files in

the file system. Specify privilege names in a comma-separated ASCII list (such as: **allowed=file_audit, file_chown;**) or use **all** to indicate all privileges. See **priv_desc**(4TSOL). Note: The allowed privileges must be a superset of the forced privileges.

low_range Sets the lower bound of the file system label range. Specify the value as a sensitivity label in ASCII format.

hi_range Sets the upper bound of the file system label range. Specify the value as a sensitivity label in ASCII format.

mld_prefix Set an alternate prefix for multilevel directories (MLDs) in Trusted Solaris file systems. The default is (**.MLD.**) Specify the value in ASCII format (for example, **.mld.**). Note: On unlabeled (fixed attribute) file systems, the MLD prefix generally has no useful effect—with the following exception. An **mld_prefix** should be supplied if another file system that has the **tsol_attr** flag is being mounted on the unlabeled file system and the root of that filesystem is an MLD. If no MLD prefix is supplied; the default is an empty string.

mnt_flag Reserved.

audit_psa Reserved.

Any of the above keywords may be omitted.

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.

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.

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) that can override or set missing filesystem-wide attributes, but mount-time attributes cannot override any attributes on any file system object (file or directory). 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**.

FILES

/etc/mnttab	Mount table
/etc/default/fs	Default local file system type. Default values can be set for the following flag in **/etc/default/fs**: **LOCAL**: The default type for a command if no *FSType* is specified. For example: **LOCAL=ufs**
/etc/vfstab	List of default parameters for each file system.
/etc/security/tsol/vfstab_adjunct	Mount-time attributes for file systems

SEE ALSO

setfacl(1), **getmldadorn**(1TSOL), **mount_cachefs**(1M), **mount_hsfs**(1M), **mount_nfs**(1MTSOL), **mount_tmpfs**(1M), **mount_ufs**(1MTSOL), **mountall**(1M), **newsecfs**(1MTSOL), **setfsattr**(1MTSOL), **umountall**(1M), **setmnt**(1M), **mnttab**(4), **default_fs**(4), **vfstab**(4), **priv_desc**(4TSOL), **vfstab_adjunct**(4TSOL), "Managing Files and File Systems" in *Trusted Solaris Administrator's Procedures*.

NOTES

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 on top of the symbolic link itself.

NAME

mount_nfs – mount remote NFS resources

SYNOPSIS

mount [–**F nfs**] [–**r**] [–**m**] [–**o** *specific_options*] [–**O**] [–**S** *attribute_list*] *resource*
mount [–**F nfs**] [–**r**] [–**m**] [–**o** *specific_options*] [–**O**] [–**S** *attribute_list*]
mount_point

AVAILABILITY

SUNWcsr

DESCRIPTION

mount attaches a named *resource* to the file-system hierarchy at the path-name 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**.

mount maintains a table of mounted file systems in **/etc/mnttab**, described in **mnttab**(4). See **mount**(1MTSOL) for more details.

OPTIONS

–**r**	Mount the specified file system read-only.
–**m**	Do not append an entry to the **/etc/mnttab** table of mounted file systems
–**o** *specific_options*	Set file-system-specific options according to a comma-separated list chosen from words listed and explained in *specific_options***for -o**.
–**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**(1MTSOL) man page.

specific_options for -o

rw | **ro**	*resource* is mounted read-write or read-only. The default is **rw**.
suid | **nosuid**	Setuid execution allowed or disallowed. The default is **suid**.
devices | **nodevices**	Allow (disallow) access to character and block devices. The default is **devices**.
	Note: In a Trusted Solaris system, 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-**noprivs**option to the **share**(1M) command, either on the command line or in the **dfstab**(4) file.
remount	If a file system is mounted read-only, remount the file system read-write.
bg | **fg**	If the first attempt fails, retry in either the background or the foreground. The default is **fg**.
quota	This option is not supported in Trusted Solaris; any attempt to set this option is ignored.
noquota	This option is not supported in Trusted Solaris; any attempt to set this option is ignored.
retry=*n*	The number of times to retry the mount operation. The default is **10000**.
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, then the NFS mount will use NFS Version 2.
	Note: File systems being mounted from Trusted Solaris 1.x servers should be specified with **vers=2**. Because the Trusted Solaris 2.x system 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.
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 the 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.
port=*n*	The server IP port number. The default is **NFS_PORT**.
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**(2TSOL) and **mkdir**(2TSOL).] 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 the parent directory.
rsize=*n*	Set the read-buffer size to *n* bytes. The default value is **32768** when using Version 3 of the NFS protocol. When using Version 2, the default value is **8192**.
wsize=*n*	Set the write-buffer size to *n* bytes. The default value is **32768** when using Version 3 of the NFS protocol. When using Version 2, the default value is **8192**.
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 **100** tenths of a second for connection-oriented transports.
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.
soft | **hard**	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**.
secure	Use DES authentication for NFS transactions.
posix	Request POSIX.1 semantics for the file system. Requires a mount Version 2 **mountd**(1MTSOL) on the server
kerberos	Use Kerberos authentication for NFS transactions.
noac	Suppress data and attribute caching.
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.

NFS File Systems

Background versus Foreground

If file systems are mounted with the **bg** option, **mount** is to retry in the background if the server's mount daemon [**mountd**(1MTSOL)] 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.

Authenticated Requests

The server may require authenticated NFS requests from the client. Either **secure** or **kerberos** authentication may be required.

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 (on 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=***n* disables attribute caching on the client. Therefore, every reference to attributes will be satisfied directly from the server although file data will still be cached. Although it guarantees that the client always has the latest file attributes from the server, setting **actimeo=***n* 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. Although it guarantees that data written by an application will be written directly to a server, where the data can be viewed immediately by other clients, setting the **noac** option 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

To mount an NFS file system:

```
example# mount serv:/usr/src /usr/src
```

To mount an NFS file system read-only with no suid privileges:

example# mount –r –o nosuid serv:/usr/src /usr/src

To mount an NFS file system read write with no devices, no privileges, and specify a default ACL , an information label, and a sensitivity label:

```
example# mount -o “nodevices,nopriv,rw -S def_acl=user::rwx,user:55:rw-, \
        group::rwx, other::r-s,mask::rw-,;ilabel=admin_low; \
        slabel=[c];” serv:/export/docs /export/docs
```

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

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**o quota** option has been removed; and 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** 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**, **proc_setil**, and **sys_trans_label**.

FILES

/etc/mnttab	Table of mounted file systems
/etc/dfs/fstypes	Default distributed file system type
/etc/vfstab	Table of automatically mounted resources
/etc/security/tsol/vfstab_adjunct	Mount-time attributes for file systems

SEE ALSO

setfacl(1), **mountall**(1M), **mount**(1MTSOL), **mountd**(1MTSOL), **mkdir**(2TSOL), **mmap**(2), **mount**(2TSOL), **open**(2TSOL), **umount**(2TSOL), **mnttab**(4), **lofs**(7FS) **vfstab.adjunct**(4TSOL)

NOTES

The sensitivity label and information 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**, and any mount-time information label must always be equal to any assigned **def_il** , if one is specified, in the NFS server's combination **tnrhdb**(4TSOL) / **tnrhtp**(4TSOL) 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 multithreaded processing, which transparently performs parallel read-ahead and write-behind.

The **devices** | **nodevices** and the **priv** | **nopriv** options are recognized but are not supported in Trusted Solaris 2.x.

NAME

mount_tmpfs – mount tmpfs file systems

SYNOPSIS

mount [–**F tmpfs**] [–**o size=***sz*] [–**O**] [–**S** *attribute_list*] *special mount_point*

AVAILABILITY

SUNWcsr

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 (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. See **mount**(1MTSOL) for more details.

OPTIONS

-o 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.

–**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**(1MTSOL) man page.

FILES

/etc/mnttab Table of mounted file systems
/etc/security/tsol/vfstab_adjunct Mount-time attributes for file systems

SUMMARY OF TRUSTED SOLARIS CHANGES

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**.

SEE ALSO

mount(1MTSOL), **mkdir**(2TSOL), **mount**(2TSOL), **open**(2TSOL), **umount**(2TSOL), **mnttab**(4), **tmpfs**(7FS) **mount**(1M), **mkdir**(2), **mount**(2), **open**(2), **umount**(2), **mnttab**(4), **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** *FSType-specific_options*] [–**O**] [–**S** *attribute_list*] *special* | *mount_point*
mount –**F ufs** [*generic_options*] [–**o** *FSType-specific_options*] [–**O**] [–**S** *attribute_list*] *special mount_point*

AVAILABILITY

SUNWcsr

DESCRIPTION

mount 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 *FSType-specific_options*. See **mount**(1MTSOL) for more details.

If *special* and *mount_point* are specified without any *FSType-specific_options*, the default is **rw**.

Note: File system objects (files and directories) in the file system being mounted may have security attributes of their own. For example, almost all file system objects have a UID, GID, and mode, may have an access ACL and default ACL, while objects in file systems that have the **tsol_attr** flag set also have a sensitivity label and an information label, and may have the file attribute flag, forced and allowed privileges. In addition to any security attributes on objects, the *file system* itself may be assigned values for security attributes either while the file system is being created using **newsecfs**(1MTSOL) or by using **setfsattr**(1MTSOL) after file system creation to set or change any of a file system's attributes. When access control decisions are made, any security attributes on a file or directory always take precedence over security attributes specified either at the filesystem level or at mount time. Security attributes specified at mount time, either with the –**S** option on the command line or in the **vfstab_adjunct**, may be supplied to override filesystem-wide security attributes but can never override security attributes on the files and directories. For any other attributes not obtainable either from the object, or at mount time, or from the file system, values of **none** or **empty** are used.

OPTIONS

See **mount**(1MTSOL) for the list of supported *generic_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:

onerror=*action* where *action* = **panic | lock | umount | repair**. This option specifies the action that UFS should take to recover from an internal inconsistency on a file system. These cause a forced system shutdown, a file system lock to be applied to the file system, the file system to be forcibly unmounted, or an automatic **fsck**, respectively. The

default is **panic**. The **repair** option downgrades to "panic" if the UFS-aware service daemon (**ufsd**) is not installed on the system. **ufsd** is a component of the unbundled Disk-Suite product.

toosoon=*number*[**s | m | h | d | w | y**]
This option specifies the minimum time that must elapse between detection of inconsistencies on a file system. If an inconsistency is detected within this time period the system is forced to shut down. This prevents pathologic repairing of a file system which is damaged repeatedly. The optional unit key letter sets the units to be **s**econds, **m**inutes, **h**ours, **d**ays, **w**eeks, or **y**ears, respectively. The default value is **1w** (1 week).

f Fake an **/etc/mnttab** entry, but do not actually mount any file systems. Parameters are not verified.

m Mount the file system without making an entry in **/etc/mnttab**.

quota This option is not supported in Trusted Solaris; any attempt to set this option is ignored.

rw | **ro** Read-write or read-only. Default is **rw**.

rq Read-write with quotas turned on. Equivalent to **rw, quota**.

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. **nosuid** can also be used to disallow setuid when mounting devices.

remount Used in conjunction with **rw**. A file system mounted read-only can be *remounted* read-write. Fails if the file system is not currently mounted or if the file system is mounted **rw**.

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**.

devices | **nodevices** Allow (disallow) opens on character and block devices. The default is **devices**.

Note: In a Trusted Solaris system, 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**.

–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 attribute list in the **mount**(1TSOL) man page.

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

SEE ALSO

mount(1MTSOL), **mountall**(1M), **mount**(2TSOL), **mnttab**(4), **vfstab**(4TSOL)

SUMMARY OF TRUSTED SOLARIS CHANGES

The **–o quota** option has been removed; and 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.

Security attributes may be specified (either by using **mount** with the **–S** option on the command line or setting attributes in the **vfstab_adjunct**(4TSOL) file) that can override or set missing filesystem-wide attributes but that cannot override any attributes on any file system object (file or directory). 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**.

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.

The **devices** | **nodevices** and the **priv** | **nopriv** options are recognized but are not supported in Trusted Solaris 2.x.

NAME

mountd – NFS mount-request server

SYNOPSIS

/usr/lib/nfs/mountd [–**n**]

AVAILABILITY

SUNWcsu

DESCRIPTION

mountd is an RPC server that answers file-system mount requests. **mountd** reads the file **/etc/dfs/sharetab** [described in **sharetab**(4)] to determine which file systems are available for mounting by which machines. **mountd** also provides information as to which file systems are mounted by which clients. This information can be printed using the **dfmounts**(1M) command.

The **mountd** daemon is automatically invoked in run level 3.

The user must have **sys_nfs**, **net_mac_read**, and **net_reply_equal** privileges to run the **mountd** daemon.

OPTION

–**n** Disable the check on mount-RPC requests that the request UID is **0** and that the originating port is a privileged port.

SUMMARY OF TRUSTED SOLARIS CHANGES

The **sys_nfs**, **net_mac_read**, and **net_reply_equal** privileges are required to run this daemon. This daemon must be run with UID **0** at a label ADMIN_LOW[ADMIN_LOW]. It must be started from the Trusted Path. 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 UID equal to **0** and must bind to a privileged port.

FILES

/etc/dfs/sharetab Shared file system table

SEE ALSO

dfmounts(1M), **sharetab**(4)

NOTES

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

NAME

ndd – Get or set driver-configuration parameters

SYNOPSIS

ndd [–**set**] *driver parameter* [*value*]

AVAILABILITY

SUNWcsu

DESCRIPTION

ndd gets or sets selected configuration parameters in some kernel drivers. Currently, **ndd** supports only the drivers that implement the TCP/IP Internet protocol family. Each driver chooses which parameters to make visible through **ndd**. Because 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 the value. If the –**set** option is given, **ndd** passes *value*, which must be specified, down to the named *driver*, which assigns the value to the named *parameter*.

By convention, drivers that support **ndd** also support a special read-only *parameter* named "?" to list the parameters supported by the driver.

To set driver parameters successfully, the –**set** option needs to inherit the **sys_net_config** privilege.

EXAMPLES

To see which parameters are supported by the TCP driver, use this command:

example% ndd /dev/tcp \?

NOTE: The parameter name "?" may need to be escaped with a backslash to prevent its being interpreted as a shell metacharacter.

This command sets the value of the **ip_forwarding** parameter in the IP driver to zero, which disables IP packet forwarding:

example% ndd –set /dev/ip ip_forwarding 0

To view the current IP forwarding table, use this 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.

SEE ALSO

ioctl(2), **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 basing 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**] [*system*] [*core*]

netstat [–**s** | –**g** | –**m** | –**p** | –**f** *address_family*] [–**P** *protocol*] [–**n**] [*system*] [*core*]

netstat –**i | –I** *interface* [*interval*] [*system*] [*core*]

netstat –**r** [–**anv**] [*system*] [*core*]

netstat –**M** [–**ns**] [*system*]] *core*]

DESCRIPTION

netstat displays the contents of various network-related data structures in various formats, depending on the options you select.

Each form in the SYNOPSIS displays something different: the first displays a list of active sockets for each protocol; the second selects one from among various other network data structures; the third displays interfaces; the fourth displays the routing table; and the fifth displays the multicast routing table.

This program should be run at the ADMIN_HIGH sensitivity label to access kernel and network configuration information. This restriction can be overridden by the **file_mac_read** privilege.

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 either **inet** For the **inet** *address_family* **unix** For the **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*.

DISPLAYS

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

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 "*" (asterisk). For more information regarding the Internet naming conventions, refer to **inet**(7P).

These are possible state values for TCP sockets:

CLOSED	Closed. The socket is not being used.
LISTEN	Listening for incoming connections
SYN_SENT	Actively trying to establish connection
SYN_RECEIVED	Initial synchronization of the connection under way
ESTABLISHED	Connection has been established.
CLOSE_WAIT	Remote shut down; waiting for the socket to close
FIN_WAIT_1	Socket closed; shutting down connection
CLOSING	Closed, then remote shutdown; awaiting acknowledgement
LAST_ACK	Remote shut down, then closed; awaiting acknowledgement
FIN_WAIT_2	Socket closed; waiting for shutdown from remote
TIME_WAIT	Wait after close for remote shutdown retransmission.

netstat [**–s** | **–g** | **–m** | **–p** | **–f** *address_family*] [**–P** *protocol*] [**–n**] [*system*] [*core*]

The form of the display depends upon which of the **–i**, **–g**, **–m**, **–p**, or **–s** options you select. **netstat** displays the information for each option you specify.

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 shows 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; for all other routes the count 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.

netstat –**M** [–**ns**] [*system*]] *core*]

The multicast routing table consists of the virtual interface table and the actual routing table.

SUMMARY OF TRUSTED SOLARIS CHANGES

To access kernel and network configuration information, this program should run at the ADMIN_HIGH sensitivity label. The **file_mac_read** privilege can override this restriction.

SEE ALSO

ifconfig(1MTSOL), **iostat**(1M), **vmstat**(1M), **hosts**(4), **networks**(4), **protocols**(4), **services**(4)

NOTES

The kernel's tables can change while **netstat** is examining them, creating incorrect or partial displays.

NAME

nfsd – NFS daemon

SYNOPSIS

/usr/lib/nfs/nfsd [–**a**] [–**c** *#_conn*] [–**p** *protocol*] [–**t** *device*] [*nservers*]

AVAILABILITY

SUNWcsu

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

These options are supported:

–**a** Start an NFS daemon over all available connectionless and connection-oriented transports, including UDP and TCP.

–**c** *#_conn* Set the maximum number of connections allowed to the NFS server over connection-oriented transports. By default, the number of connections is unlimited.

–**p** *protocol* Start an NFS daemon over the specified protocol.

–**t** *device* Start an NFS daemon for the transport specified by the given device.

OPERANDS

This operand is 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 Trusted Solaris, 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

RETURN VALUES

If the daemon started successfully, the exit status is **0**. If the daemon failed to start, the exit status is **1**.

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.

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

SEE ALSO

ps(1), **mountd**(1M), **sharetab**(4), **system**(4)

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 start and restart, 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**]

AVAILABILITY

SUNWcsu

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.

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_net_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**(1MTSOL).

The following mount flags are set by mount options:

auth (one of the following values):

none No authentication.
unix UNIX style authentication (UID, GID).
short Short hand UNIX style authentication.
des **des** style authentication (encrypted timestamps).
kerb **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.

SUMMARY OF TRUSTED SOLARIS CHANGES

The **sys_net_config** privilege is required to use the **-z** option.

SEE ALSO

mount_nfs(1MTSOL)
Solaris Advanced Installation Guide
UNKNOWN TITLE ABBREVIATION: x86INSTALL

NAME

nis_cachemgr – NIS+ utility to cache location information about NIS+ servers

SYNOPSIS

/usr/sbin/nis_cachemgr [**–i**] [**–n**] [**–v**]

AVAILABILITY

SUNWnisu

DESCRIPTION

The **nis_cachemgr** daemon maintains a cache of the NIS+ directory objects. The cache contains location information necessary to contact the NIS+ servers that serve the various directories in the name space. 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 which access NIS+ on that 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. 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.

Note: 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**(1MTSOL)). See the WARNINGS section, below. Additionally, a "**keylogin -r**" needs to be done on the machine.

nisshowcache(1M) can be used to look at the cached objects.

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.

–n Run **nis_cachemgr** in an *insecure* mode. By default, before adding a directory object to the shared cache, on the request of another process on the machine, it checks the encrypted signature on the request to make sure that the directory object is a valid one and is sent by an authorized server. In this mode, **nis_cachemgr** adds the directory object to the shared cache without making this check.

–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**.

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+

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.

SEE ALSO

keylogin(1), **nisaddcred**(1M), **nisinit**(1M), **nisshowcache**(1M), **rpc.nisd**(1MTSOL), **mmap**(2), **syslog**(3), **nisfiles**(4)

DIAGNOSTICS

The **nis_cachemgr** daemon logs error messages and warnings using syslog (see **syslog**(3)). Error messages are logged to the **DAEMON** facility with a priority of **LOG_ERR,** and warning messages with a priority of **LOG_WARNING.** Additional status messages can be obtained using the **–v** option.

WARNINGS

If the host principal does not have the proper security credentials in the **cred.org_dir** table for its domain, then running this program without the '–n' insecure mode option may significantly *degrade* the performance of processes issuing NIS+ requests.

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*]]

AVAILABILITY

SUNWtsol

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**(1MTSOL).

The table argument accepts standard names that are used in the administration of Solaris systems and nonstandard *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**, **shadow**, **tsoluser**, **tnrhtp**, **tnrhdb**, **tntime**, and **tsolprof**. Note that the **shadow** table is used only when populating from files. The nonstandard 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, the shell script 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 password 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 if the password is not specified, the shell script will use “nisplus,” the default password. **nispopulate** will not overwrite any existing credential entries in the credential table. Use **nisclient**(1M) to overwrite the entries in the **cred** table. **nisclient**(1M) 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 a NIS+ credential table with level-2 authentication (DES) from the password and hosts tables of the specified domain. The valid table arguments for this operation are **passwd** and **hosts**. If this argument is not specified, the shell script will use both **passwd** and **hosts** as the input source.

If **nispopulate** was earlier used with the **-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 used *only* with the –**Y** option.
–**C**	Populates the NIS+ credential table from passwordd and hosts tables using **DES** authentication (security level 2).
–**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 host name from which the NIS maps are copied. This option is used *only* with the –**Y** option. This host must already exist in either the NIS+ **hosts** table or **/etc/hosts** file. If the host name 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 option is used *only* when you are populating the **hosts** and **passwd** tables. The default password 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 option is used *only* with the –**Y** option.
–**p** <*directory_path*>	Specifies the directory where the files are stored. This option is used *only* 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.

–u	Updates (adds, deletes, modifies) the NIS+ tables 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 merely prints the commands that it would have executed without actually executing them. The default is off.
–Y	Populates the NIS+ tables from NIS maps.
–y <*NIS_domain*>	Specifies the NIS domain from which to copy the NIS maps. This option is used *only* with the **–Y** option. The default domain name is the same as the local domain name.

EXAMPLES

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.

Update all of the NIS+ standard tables from the same NIS domain and hosts shown in the previous example:

**nis_server# /usr/lib/nis/nispopulate -Y -u -y yp.sun.COM –h yp_host **
–d xyz.sun.com.

Populate the **hosts** table in domain **xyz.sun.com.** from the hosts file in the **/var/nis/files** directory and use "somepasswd" as the network password for key encryption:

nis_server# /usr/lib/nis/nispopulate -F -p /var/nis/files -l somepasswd hosts

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

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

Create and populate a nonstandard *key-value*-type NIS+ table, "private", from the file **/var/nis/files/private**: (**nispopulate** assumes that the **private.org_dir** *key-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

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

SUMMARY OF TRUSTED SOLARIS CHANGES

nissetup creates the following additional tables: **tsolprof**,**tsoluser**, **tnrhdb**,**tnrhtp**, and **tntime**.

SEE ALSO

nis+(1), **nistbladm**(1), **nisaddcred**(1M), **nisaddent**(1M), **nisclient**(1M), **nisserver**(1M), **nissetup**(1MTSOL), **rpc.nisd**(1M), **ypxfr**(1M)

NAME

nissetup – Initialize a NIS+ domain

SYNOPSIS

/usr/lib/nis/nissetup [–**Y**] [*domain*]

AVAILABILITY

SUNWtsolu

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 an NIS+ domain and an NIS domain using the backward-compatibility flag. This option 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.x **nissetup** creates the following additional tables: **tsolprof**,**tsoluser**, **tnrhdb**,**tnrhtp**, and **tntime**.

SEE ALSO

nis+(1), **nismkdir**(1), **nistbladm**(1), **nisaddent**(1M), **nisinit**(1M) **nisserver**(1M),

NOTES

Although this command creates the default tables, it does not initialize them with data. This initialization 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*]

AVAILABILITY

SUNWcsu

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 does not rescan the **/etc/nsswitch.conf** file; if this file is changed, the machine should be rebooted or **nscd** stopped and restarted, as shown in the EXAMPLES below.

nscd runs at the sensitivity label ADMIN_LOW. However, it can communicate with 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

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**, **proc_nofloat**, **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

SEE ALSO

gethostbyname(3N), **group**(4), **hosts**(4), **nscd.conf**(4), **nsswitch.conf**(4), **passwd**(4)

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 [–**opt**] [*host* | –] [*server*]

AVAILABILITY

SUNWcsu

DESCRIPTION

nslookup is an interactive program to query ARPA Internet domain name servers. The user can contact servers to request information about a specific host, or print a list of hosts in the 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

–**opt**	Allows you to set options as in the interactive **set** option shown below. For example –**querytype=HINFO** to ask for host information instead of the default query type of **A** for address information.
host	Look up the host *host* directly and do not go into interactive mode.
–	Use the name server specified next on the command line instead of the servers in **/etc/resolv.conf**. Note that if both **host** and **server** are specified on the command line this – argument becomes optional.
server	Use the name server specified. This can be either a name or an Internet address. If this fails, it will default to using the entries in the **/etc/resolv.conf** file.

USAGE

Overview

The Internet domain name-space is tree-structured, with five top-level domains at present:

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 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 will print 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** command to see who is on the system, or to finger a specific person. To get other information about the host, use the

set querytype = *value*

command to change the type of information desired and request another lookup. (**finger** requires the type to be **A**.)

Commands

To exit, type CTRL-D (EOF).

The command line length must be less than 80 characters.

An unrecognized command will be interpreted as a host name.

host [*server*]
Look up information for *host* using the current default server, or using *server* if it is specified.

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.

root Change the default server to the server for the root of the domain name space. Currently, the host **ns.nic.ddn.mil** is used; this command is a synonym for **lserver ns.nic.ddn.mil**. The name of the root server can be changed with the **set root** command.

finger [*name*]
Connect with the finger server on the current host, which is defined by a previous successful lookup for a host's address information (see the **set** *querytype* **= A**

command). As with the shell, output can be redirected to a named file using > and >>.

ls [–**ah**]
List the information available for *domain*. The default output contains host names and their Internet addresses. The –**a** option lists aliases of hosts in the domain. The –**h** option lists CPU and operating system information for the domain. As with the shell, output can be redirected to a named file using > and >>. When output is directed to a file, hash marks are printed for every 50 records received from the server.

view *filename*
Sort and list the output of the **ls** command with **more**(1).

help
? Print a brief summary of commands.

set *keyword* [**=** *value*] This command is used to change state information that affects the lookups. Valid keywords are:

all Print the current values of the various options to **set**. Information about the current default server and host is also printed.

[**no**]**deb**[**ug**]
Turn debugging mode on. A lot more information is printed about the packet sent to the server and the resulting answer. The default is **nodebug**.

[**no**]**def**[*name*]
Append the default domain name to every lookup. The default is **nodefname**.

do[**main**]**=** *filename*
Change the default domain name to *filename*. The default domain name is appended to all lookup requests if **defname** option has been set. The default is the value in **/etc/resolv.conf**.

q[**querytype**] **=** *value*
Change the type of information returned from a query to one of:

A	The host's Internet address (the default).
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.

(Other types specified in the RFC 883 document are valid, but are not very useful.)

[**no**]**recurse**
Tell the name server to query other servers if it does not have the information. The default is **recurse**.

ret[**ry**] **=** *count*
Set the number of times to retry a request before giving up to *count*. When a reply to a request is not received within a certain amount of time (changed with **set timeout**), the request is resent. The default is *count* is **2**.

ro[**ot**] **=** *host*
Change the name of the root server to *host*. This affects the **root** command. The default root server is **ns.nic.ddn.mil**.

t[**timeout**] **=** *interval*
Change the time-out for a reply to *interval* seconds. The default *interval* is **10** seconds.

[**no**]**v[c]**
Always use a virtual circuit when sending requests to the server. The default is **novc**.

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

SEE ALSO

nstest(1M), **resolver**(3NTSOL), **resolv.conf**(4)

RFC 882, RFC 883

DIAGNOSTICS

If the lookup request was not successful, an error message is printed. Possible errors are:

Time-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 information Depending on the query type set with the **set querytype** command, no information about the host was available, though the host name is valid.

Non-existent domain
The host or domain name does not exist.

Connection refused
Network is unreachable
The connection to the name or finger server could not be made at the current time. This error commonly occurs with **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.

The following error should not occur and it indicates a bug in the program.

Format error

The name server found that the request packet was not in the proper format.

NAME

nstest – DNS test shell

SYNOPSIS

nstest [**–d**] [**–i**] [**–r**] [**–v**] [**–p** *port*] [*inet_addr* [*logfile*]]

AVAILABILITY

SUNWcsu

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**(3NTSOL) for a description of the **RES_IGNTC** flag.

–r Turns off the **RES_RECURSE** flag on the queries it makes. See **resolver**(3NTSOL) 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**(3NTSOL) 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

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
```

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

To look up a PTR record, enter:

$ nstest 192.9.5.1
> p5.5.9.192.in-addr.arpa

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

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)

SEE ALSO

nslookup(1M), **resolver**(3NTSOL)

NAME

pbind – Control and query bindings of processes to processors

SYNOPSIS

pbind –**b** *processor_id pid ...*
pbind –**u** *pid ...*
pbind [–**q**] [*pid ...*]

AVAILABILITY

SUNWcsu

DESCRIPTION

pbind binds all the LWPs (light-weight processes) of a process to a processor, or removes or displays the bindings.

OPTIONS

–**b** *processor_id*	Binds all the LWPs of the specified processes to the processor *processor_id*.
–**u**	Removes the bindings of all LWPs of the specified processes.
–**q**	Displays the bindings of the specified processes or of all processes.

USAGE

The –**b** option binds all of the LWPs of the specified processes to the specified processor. The processor must be present and on line, a condition which can be determined by the **psrinfo**(1M) command.

When it is bound to a processor, an LWP 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 to 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.

The –**u** option removes the bindings from all the LWPs of the specified processes, allowing them to be executed on any on-line processor.

The –**q** option displays the bindings of the specified processes. If a process is composed of multiple LWPs that have different bindings, the bindings of only one of the bound LWPs will be shown.

EXAMPLES

Binding processes

This command binds processes 204 and 223 to processor 2:

```
example% pbind –b 2 204 223
```

This command generates these messages:

```
process id 204: was 2, now 2
process id 223: was 3, now 2
```

Unbinding a process

To unbind process 204:

example% pbind –u 204

Querying Bindings

The command

example% pbind –q 1 149 101

generates this output:

process id 1: 0
process id 149: 3
process id 101: not bound

This 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.

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

psrinfo(1M), **psradm**(1MTSOL), **processor_bind**(2), **processor_info**(2), **sysconf**(3C)

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 offline.

NAME

pfsh – Profile shell

SYNOPSIS

pfsh [–**acefhiknprstuvx**] [*argument*...]

AVAILABILITY

SUNWtsolu

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 [–**hpniu**] Displays a 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 an ASCII list of the privileges assigned to each command in the command list.
- **-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**.)

SEE ALSO

sh(1), **tsolprof**(4TSOL), **tsoluser**(4TSOL)

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 not expected to change between minor releases of Trusted Solaris systems, these interfaces may change.

NAME

praudit – Print contents of an audit-trail file

SYNOPSIS

praudit [–**lrs**] [–**d***del*] [*filename* . . .]

AVAILABILITY

SUNWcsu

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**(4TSOL). By default, times, security labels, user and group IDs (UIDs and GIDs) 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 system_high or system_low binary labels are translated to their ASCII equivalents. See **pattr**(1TSOL) and **getpattr**(2TSOL) for more information.

OPTIONS

–**l** Print one line per record. The record type and event fields are always converted to their short ASCII representation as they are 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 mutually exclusive. If both are used, a format-usage error message results.

–**s** Print records in their short form. All numeric fields are converted to ASCII and displayed. The short ASCII representations for the record type and event fields are used. This option and the –**r** option are mutually exclusive. If both are used, a format-usage error message results.

–**d***del* Use *del* as the field delimiter instead of the default delimiter, which is the comma. If it has special meaning for the shell, *del* must be quoted. The maximum size of a delimiter is four characters.

SUMMARY OF TRUSTED SOLARIS CHANGES

This function uses the **PAF_LABEL_VIEW** rather than the **PAR_LABEL_VIEW** process attribute flag and converts security labels as well as times and IDs.

FILES

/etc/security/audit_event
/etc/security/audit_class

SEE ALSO

bsmconv(1MTSOL), **pattr**(1TSOL), **audit**(2TSOL), **getauditflags**(3TSOL), **getpattr**(2TSOL), **audit.log**(4TSOL), **audit_class**(4TSOL), **audit_event**(4TSOL) **group**(4), **passwd**(4)

NOTES

This functionality is active only if the audit module has been enabled. By default, this module is enabled on Trusted Solaris systems. See **bsmconv**(1MTSOL) for more information.

NAME

prtconf – Print system configuration

SPARC SYNOPSIS

/usr/sbin/prtconf [**–F**] [**–p**] [**–P**] [**–v**] [**–V**]

x86 SYNOPSIS

/usr/sbin/prtconf [**–P**] [**–v**]

AVAILABILITY

SUNWcsu

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

–P Include information about pseudo devices. By default, information regarding pseudo devices is omitted.

–v Specifies verbose mode. This option is available during command execution only from an administrative role.

–F (SPARC only). Return the device path name of the console frame buffer if one exists. If there is no frame buffer, **prtconf** returns a nonzero exit code. This flag overrides all others and returns only the name of the console, frame buffer device, or a nonzero 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 (SPARC only). Displays information derived from the device tree provided by the firmware (PROM).

–V (SPARC only). Display platform-dependent PROM version information. This flag must be used by itself because it overrides all other flags. The output is a string. The format of the string is arbitrary and platform-dependent.

EXIT STATUS

If successful, **prtconf** returns **0**. If an error occurs, **prtconf** prints an error message and returns **1**. For example, when an illegal option is specified, **prtconf** returns **1**. On a SPARC system, when the **–F** option is specified and the console output device is not a framebuffer, **prtconf** returns **1**.

SPARC EXAMPLES

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
```

x86 EXAMPLES

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%
```

SUMMARY OF TRUSTED SOLARIS CHANGES

The **–v** option of this command is intended for execution only from an administrative role.

SEE ALSO
SPARC Only

modinfo(1M), **sysdef**(1M)
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 – Set processors on line or off line

SYNOPSIS

psradm –**f** | –**n** [–**v**] *processor_id* . . .
psradm –**a** –**f** | –**n** [–**v**]

AVAILABILITY

SUNWcsu

DESCRIPTION

psradm takes a processor off line or brings it on line. An offline processor will do little or no work. The actual effect of being offline may vary from machine to machine.

OPTIONS

–**f** Take the specified processors off line.

–**n** Bring the specified processors on line.

–**a** Perform the action on all processors or on as many as possible.

–**v** Output a message giving the results of each attempted operation.

USAGE

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 must remain on line.

EXAMPLES

Set processors 2 and 3 off line:

psradm -f 2 3

Set all processors on line:

psradm -n -a

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, this command needs the **sys_config** privilege.

FILES

/etc/wtmp For records logging processor status changes

SEE ALSO

psrinfo(1M), **p_online**(2)

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 online processor in the system, or is needed by the system to provide some essential service.

psradm: processor 0: Not owner
The user does not have permission to change processor status.

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.

SUMMARY OF TRUSTED SOLARIS CHANGES

This program needs to inherit the **sys_config** privilege to run properly.

NAME

reboot – restart the operating system

SYNOPSIS

/usr/sbin/reboot [–**dlnq**] [*boot-arguments*]

AVAILABILITY

SUNWcsu

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 the super-user 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** Does not perform the **sync**(1M) command. 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.

On x86 systems only, note that currently, boot arguments are not passed on to the **boot**(1M) program, so they have no effect. You must type in the arguments when responding to the boot prompt ">" to have the desired effect.

EXAMPLES

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

SUMMARY OF TRUSTED SOLARIS DIFFERENCE

This command requires the **sys_boot** privilege in order to run.

SEE ALSO

boot(1M), **crash**(1M), **fsck**(1M), **halt**(1MTSOL), **init**(1M), **shutdown**(1M), **sync**(1M), **syslogd**(1M), **uadmin**(2), **reboot**(3C)

NAME

rem_drv – Remove a device driver from the system

SYNOPSIS

rem_drv [–**b** *basedir*] *device_driver*

AVAILABILITY

SUNWcsu

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* Set 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

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

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

add_drv(1MTSOL), **drvconfig**(1M), **kernel**(1M)

NAME

route – Manually manipulate the routing tables

SYNOPSIS

route [–**fn**] **add** | **delete** [**host** | **net**] *destination* [*gateway* [*metric*]]

AVAILABILITY

SUNWcsu

DESCRIPTION

route manually manipulates the network routing tables normally maintained by the system routing daemon, **routed**(1M), or through default routes and redirect messages from routers. **route**, provided it inherits the **sys_net_config** privilege, operates directly on the routing table for the specific host or network indicated by *destination*. **default** (set up by creating the */etc/defaultrouter* file) is available for gateways to use after all other routes have been attempted. The *gateway* argument, if present, indicates the network gateway to which packets should be addressed. The *metric* argument indicates the number of "hops" to the *destination*. The *metric* is required for **add** commands; it must be either **0** if the destination is on a directly attached network, or nonzero if the route utilizes one or more gateways.

A *metric* of **0** implies that the route refers not to a gateway but 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, a *metric* of **0** is used to specify *all destinations are local*, meaning that a host should ARP for all addresses by adding a default route containing a *metric* of **0**, as illustrated in this example:

route add default *hostname* **0**

hostname is the name or IP address associated with the network interface over which all packets should be sent. On a host with a single network interface, *hostname* is normally the same as the *nodename* returned by **uname** –**n**.

The **add** command instructs **route** to add a route to *destination*. **delete** deletes a route.

Routes to a particular host must be distinguished from those to a network. The optional keywords **net** and **host** force the destination to be interpreted as a network or a host, respectively. Without these keywords, the route is assumed to be to a network if the destination has a "local address part" of **INADDR_ANY**; if not, the route is assumed to be to a host. If the route is to a destination connected by a gateway, the *metric* parameter should be greater than **0**. When a route with metric **0** is added, the gateway given is the address of this host on the common network, indicating the interface to be used directly for transmission. Except **default**, all symbolic names specified for a *destination* or *gateway* are looked up in the hosts database through **gethostbyname**(3N). If this lookup fails, the name is looked up in the networks database through **getnetbyname**(3N). **default** is a valid destination used for all routes if there is no host nor network route specified.

OPTIONS

–**f** Flush the routing tables of all gateway entries. If this option is used in conjunction with one of the commands described previously, **route** flushes the gateways before performing the command.

–**n** Prevent attempts to print host and network names symbolically when reporting actions. This option is useful, for example, when all name servers on your local

net are down and you need a route before you can contact the name server.

NOTES

All destinations are local assumes that the routers implement **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.

SUMMARY OF TRUSTED SOLARIS CHANGES

To open the IP device for adding or deleting a route, this program must inherit the **sys_net_config** privilege and run at a sensitivity label of ADMIN_HIGH, and effective user ID of **0** or in *sys* group. The **file_mac_read** privilege can override the ADMIN_HIGH MAC policy. The **file_dac_read** privilege can override the UID **0** or *sys* group DAC requirement.

FILES

/etc/hosts
/etc/networks

SEE ALSO

netstat(1MTSOL), **routed**(1M), **ioctl**(2), **gethostbyname**(3N), **getnetbyname**(3N), **routing**(4)

DIAGNOSTICS

add [**host** | **net**] *destination*:*gateway*
The specified route is being added to the tables. The values printed are from the routing table entry supplied in the **ioctl**(2) call.

delete [**host** | **net**] *destination*:*gateway*
The specified route is being deleted.

destination **done**
When the –**f** flag is specified, each deletion of a routing-table entry produces 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 deletion operation was attempted for an entry that is not in the table.

routing table overflow
An addition operation was attempted, but the system was unable to allocate memory to create the new entry.

NAME

rpc.bootparamd, bootparamd – boot parameter server

SYNOPSIS

/usr/sbin/rpc.bootparamd [–**d**]

AVAILABILITY

SUNWcsu

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**(1MTSOL) or directly from the command line.

OPTIONS

–**d** Display debugging information.

SUMMARY OF TRUSTED SOLARIS CHANGES

rpc.bootparamd should be started from the Trusted Path with a UID 0 and sensitivity label ADMIN_LOW.

FILES

/etc/bootparams	boot parameter data base
/etc/nsswitch.conf	configuration file for the name-service switch

SEE ALSO

inetd(1MTSOL), **bootparams**(4), **nsswitch.conf**(4)

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 a 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

rpc.nisd, nisd – NIS+ service daemon

SYNOPSIS

/usr/sbin/rpc.nisd [–**ACDFhlv**] [–**Y** [–**B** [–**t** *netid*]]] [–**d** *dictionary*] [–**L** *load*] [–**S** *level*]

AVAILABILITY

SUNWnisu

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 2.x 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.

–**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** Put the server into NIS (YP) compatibility mode. When operating in this mode, the NIS+ server will respond to NIS Version 2 requests using the version 2 protocol. Because the YP protocol is not authenticated, only those items that have read access to nobody (the unauthenticated request) will be visible through the V2 protocol. It supports only the standard Version 2 maps in this mode (see –**B** option and **NOTES** in **ypfiles**(4)).

ypserve and other NIS (YP) compatibility is not supported in Trusted Solaris. Using this option may put the daemon in an unknown state.

–**B** Provide ypserv compatible DNS forwarding for NIS host requests. The DNS resolving process, **rpc.nisd_resolv**, is started and controlled by **rpc.nisd**. This option 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**(4TSOL) and **nslookup**(1M).

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 accepts only **AUTH_DES** credentials for authentication and authorization. This is the highest level of security currently provided by the **NIS+** service. This is the default security level if the –**S** option is not used.

EXAMPLES

The following example sets up the NIS+ service.

```
example% rpc.nisd
```

The following example sets up the NIS+ service, emulating YP with DNS forwarding.

```
example% rpc.nisd –YB
```

ENVIRONMENT

NETPATH The transports that the NIS+ service will use can be limited by setting this environment variable (see **netconfig**(4)).

FILES

/var/nis/parent.object This file contains an XDR encoded NIS+ object that describes the namespace above a root server. This parent namespace may be another NIS+ namespace or a foreign namespace such as one served by the Domain Name Service. It is only present on servers that are serving the root of the namespace.

/var/nis/root.object This file contains an XDR encoded NIS+ object that describes

	the root of the namespace. It is only present on servers that are serving the root of the namespace.
/etc/init.d/rpc	initialization script for NIS+

SUMMARY OF TRUSTED SOLARIS CHANGES

A Trusted Solaris 2.x 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.

SEE ALSO

nis_cachemgr(1MTSOL), **nisinit**(1M), **nissetup**(1M), **nslookup**(1M), **syslogd**(1M), **rpc.nisd_resolv**(1M), **rpc_nispasswdd**(1MTSOL), **nis_db**(3N), **netconfig**(4), **nisfiles**(4), **resolv.conf**(4TSOL), **ypfiles**(4), **ticots**(7D)

NAME

rpc.nisd_resolv, nisd_resolv – NIS+ service daemon

SYNOPSIS

rpc.nisd_resolv [**–v** | **–V**] [**–F** [**–C** *fd*]] [**–t** *xx*] [**–p** *yy*]

AVAILABILITY

SUNWnisu

DESCRIPTION

rpc.nisd_resolv is an auxiliary process to **rpc.nisd** used to provide ypserv compatible DNS forwarding for NIS host requests. It is generally started by invoking **rpc.nisd** with the **–B** option. **rpc.nisd_resolv** can also be started independently with the following options.

This command is not supported in Trusted Solaris because **ypserv** and other NIS(YP) compatibility is unsupported.

OPTIONS

–F	Run in foreground.
–C *fd*	Use *fd* for service xprt (from **nisd**).
–v	Verbose. Send output to the syslog daemon.
–V	Verbose. Send output to stdout.
–t *xx*	Use transport *xx*.
–p *yy*	Use transient program# *yy*.

SUMMARY OF TRUSTED SOLARIS CHANGES

This command is not supported in Trusted Solaris because **ypserv** and other NIS(YP) compatibility is unsupported.

SEE ALSO

nslookup(1M), **rpc.nisd**(1M), **resolv.conf**(4).

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**]

AVAILABILITY

SUNWnisu

DESCRIPTION

rpc.nispasswdd daemon is an ONC+ RPC service that services password update requests from **nispasswd**(1) and **yppasswd**(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.

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** 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.

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 the user if they do not have one. By specifying this option, if the user does not have a credential, then one will be generated for them 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.

FILES

/etc/init.d/rpc initialization script for NIS+

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.

SEE ALSO

nispasswd(1), **passwd**(1), **yppasswd**(1), **rpc.nisd**(1M), **syslogd**(1M), **nsswitch.conf**(4)

NAME

rpc.tbootparamd – Trusted Solaris boot parameter server

SYNOPSIS

/usr/sbin/rpc.tbootparamd

AVAILABILITY

SUNWtsolu

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.

SEE ALSO

tbootparam(1MTSOL), **chstate**(2TSOL)

NAME

rpcbind – Maps universal addresses to RPC program number

SYNOPSIS

rpcbind [**–d**] [**–w**]

AVAILABILITY

SUNWcsu

DESCRIPTION

rpcbind is a server that converts RPC program numbers into universal addresses. **rpcbind** must be running on the host to be able to make RPC calls on a server on that machine.

When it is started, an RPC service tells **rpcbind** the address at which it is listening, and the RPC program numbers it is prepared to serve. When it wishes to make an RPC call to a given program number, a client first contacts **rpcbind** on the server machine to determine the address to which 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 it is started, **rpcbind** checks that certain name-to-address-translation calls function correctly. If they fail, the network configuration databases may be corrupt. Because RPC services cannot function correctly in this situation, **rpcbind** reports the condition and terminates.

OPTIONS

–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 it aborts or terminates on **SIGINT** or **SIGTERM**, **rpcbind** 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. Thus **rpcbind** can resume operation without requiring that all RPC services 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
/tmp/rpcbind.file

SEE ALSO

rpcinfo(1MTSOL), **rpcbind**(3NTSOL)

NOTES

Terminating **rpcbind** with **SIGKILL** will prevent the warm-start files from being written.

All RPC servers must be restarted in any of these circumstances: **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

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*

AVAILABILITY

SUNWcsu

DESCRIPTION

rpcinfo makes an RPC call to an RPC server and reports what it finds.

In the first synopsis, **rpcinfo** lists the RPC services registered 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 the RPC services registered with **rpcbind**, version 2. Note that the format of the information is different in the first and the second synopsis 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 that 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.

EXAMPLES 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.

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.

–**b** Make an RPC broadcast to procedure 0 of the specified *prognum* and *versnum* and report all hosts that respond. If *transport* is specified, this option 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**.

–**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 option 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 it is not specified, *host* 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 it is not specified, *host* 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

Show all of the RPC services registered on the local machine:

example% rpcinfo

Show all of the RPC services registered with **rpcbind** on the machine named **klaxon**:

example% rpcinfo klaxon

The information displayed by this commands can be quite lengthy. Use the **–s** option to display a more concise list:

example% rpcinfo –s klaxon

program	version(s)	netid(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

Show whether the RPC service with program number *prognum* and version *versnum* is registered on the machine named **klaxon** for the transport TCP:

example% rpcinfo –T tcp klaxon *prognum versnum*

Show all RPC services registered with version 2 of the **rpcbind** protocol on the local machine:

example% rpcinfo –p

Delete the registration for version 1 of the **walld** (program number **100008**) service for all transports:

example# rpcinfo –d 100008 1

or

example# rpcinfo –d walld 1

SUMMARY OF TRUSTED SOLARIS CHANGES

The **–M** option is added. The **–b** option requires **net_broadcast** privilege; and the **–d** option requires **net_setid** privilege if the requestor does not own the mapping being deleted, and **net_privaddr** if the mapping is for a privileged port.

SEE ALSO

rpcbind(1MTSOL), **rpc**(3NTSOL), **netconfig**(4), **rpc**(4)

NAME

runpd - run a command for privilege debugging

SYNOPSIS

/usr/sbin/runpd [–**p**] *command* [*args*]

AVAILABILITY

SUNWtsolu

DESCRIPTION

The **runpd** command is a debugging tool 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.

runpd returns the exit code it receives from *command*.

OPTIONS

–**p** Execute *command* with the trusted_path process attribute. This option is useful when testing a program (*command*) that requires the attribute.

SEE ALSO

syslog.conf(4), **system**(4), **pattr**(1TSOL)

NOTES

To enable privilege debugging with **runpd** on the system, the tsol:tsol_privs_debug kernel variable in **/etc/system** must be set to 1, and an entry for kern.debug and / or local7.debug must be added to **/etc/syslog.conf**.

The **runpd** command is uncommitted, which means that it may change between minor releases of Trusted Solaris 2.x.

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*

AVAILABILITY

SUNWcsu

DESCRIPTION

rwall reads a message from standard input until EOF. **rwall** then sends this message, preceded by the line:

Broadcast Message ...

to all users logged in on the specified host machines. With the –**n** option, **rwall** 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.

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

inetd(1MTSOL), **listen**(1M), **pmadm**(1M), **sacadm**(1M), **wall**(1M)

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**] [–**bi**] [–**bm**] [–**bp**] [–**bs**] [–**bt**] [–**bv**]
[–**B** *type*] [–**C** *file*] [–**d** *X*] [–**F** *fullname*] [–**f** *name*] [–**h** *N*] [–**M** *id*] [–**n**]
[–**o** *xvalue*] [–**p** *protocol*] [–**q** [*time*]] [–**q** *Xstring*] [–**r** *name*] [–**t**] [–**v**]
[–**X** *logfile*] [*address* . . .]

AVAILABILITY

SUNWcsu

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 preformatted 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. **sendmail** 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 by using the YP name service 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. 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," which includes "john" in the expansion, then the letter will not be delivered to john.

sendmail will also route mail directly to other known hosts in a local network. The list of hosts to which mail is directly sent is maintained in the file **/usr/lib/mailhosts**.

If a letter is found to be undeliverable, it is returned to the sender with diagnostics that indicate the location and nature of the failure; or the letter 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. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the –**bd** option.

–**bd** Run as a daemon, waiting for incoming SMTP connections. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the **net_mac_read**, **net_privaddr**, **proc_nofloat**, and **proc_setil** privileges.

–**bi** Initialize the **aliases**(4) database. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the –**bd** option.

–**bm** Deliver mail in the usual way. (default)

–bp	Print a summary of the mail queue. Only messages queued at sensitivity labels dominated by the process are printed.
–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. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the **–bd** option.
–bt	Run in address-test mode. This mode reads addresses and shows the steps in parsing; it is used for debugging configuration tables. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the **–bd** option.
–bv	Verify names only—do not try to collect or deliver a message. Verify mode is normally used for validating users or mailing lists. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the **–bd** option.
–B *type*	Indicate body *type* (**7BIT** or **8BITMIME**).
–C *file*	Use alternate configuration file.
-d *X*	Set debugging value to *X*. This option is ignored if **sendmail** was not invoked from the trusted path.
–F *fullname*	Set the full name of the sender.
–f *name*	Set the name of the "from" person (the sender of the mail). **–f** can be used only by trusted users (who are listed in the configuration file).
–h *N*	Set the hop count to *N*. The hop count is incremented every time the mail is processed. When the hop count reaches a limit, the mail is returned with an error message, the victim of an aliasing loop.
–M *id*	Attempt to deliver the queued message with message-id *id*. This option is supported for backward compatibility but the **–qI** option is preferred. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the **–q** option.
–n	Do not do aliasing.
–o *xvalue*	Set option *x* to the specified *value*. See **Processing Options**.
–p *protocol*	Set the sending protocol. Programs are encouraged to set this 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** both set the timeout to one hour, thirty minutes. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the **file_mac_read**, **file_mac_search**, **proc_nofloat**, and **proc_setil** privileges.

–**q** *Xstring* Run the queue once, limiting the jobs to those matching *Xstring*. The key letter *X* can be any of these:

I To limit based on queue identifier (See –**M** option.)
R To limit based on recipient (See –**R** option.)
S To limit based on sender

A particular queued job is accepted if one of the corresponding addresses contains the indicated *string*. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the –**q** option.

–**r** *name* An alternate and obsolete form of the –**f** flag

–**R** *string* Go through the queue of pending mail and attempt to deliver any message with a recipient containing the specified string. This option is useful for clearing out mail directed to a machine that has been down for a while. This option is supported for backward compatibility, but the –**qR** option is preferred. To use this option, **sendmail** must be invoked from the trusted path at sensitivity label ADMIN_LOW and must inherit the same privileges as for the –**q** option.

–**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.

–**v** Go into verbose mode. Alias expansions will be announced and so on.

–**X** *logfile* Log all traffic in and out of **sendmail** in the indicated *logfile* for debugging mailer problems. This process produces a lot of data very quickly and should be used sparingly. This option is ignored if not invoked from the trusted path.

Processing Options

There are a number of “random ” options that can be set from a configuration file. Options are represented by single characters. The syntax of this line is

O*ovalue*

This line sets option *o* 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.)
- A time interval

These options are supported:

a*N* Wait up to *N* minutes for an "@:@" entry to exist in the **aliases**(4) database before starting up. If the entry does not appear in *N* minutes, either rebuild the database (if the **D** option is also set) or issue a warning.

A*file* Specify possible alias file(s).

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 response invites the sender to try again later. The optional *M* is a maximum message size advertised in the **ESMTP EHLO** response. The optional *M* is currently otherwise unused.

B*c* Set the blank substitution character to *c*. Unquoted spaces in addresses are replaced by this character. Defaults to SPACE (no change is made)

c If an outgoing mailer is marked as being expensive, do not connect immediately.

C*N* Checkpoint the queue every *N* (default **10**) addresses sent. If your system crashes during delivery to a large list, this option prevents retransmission to any but the last *N* recipients.

d*x* Deliver in mode *x*. Legal modes are

i Deliver interactively (synchronously).
b Deliver in background (asynchronously).
q Just queue the message (deliver during queue run)

Defaults to **b** if no option is specified, or to **i** if it is specified but given no argument (that is, **Od** is equivalent to **Odi**).

D 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 unless **newaliases**(1) is invoked.

e*x* Dispose of errors using mode *x*. The values for *x* are

p Print error messages. (default)
q Print no messages; just give exit status.
m Mail back errors.
w Write back errors. (Mail if user is not logged in.)
e Mail back errors and give zero exit status always.

E*file/message* Prepend error messages with the indicated message. If it begins with a slash, the argument is assumed to be the path name of a file containing a message. (This setting is recommended.) Otherwise, the argument 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 that does not exist or that is not readable, no message is printed.

f Save UNIX-style **From** lines at the front of headers. Normally these lines are assumed redundant and discarded.

F*mode* Set the file mode for queue files.

g*n* Set to *n* the default group ID for mailers to run in. Defaults to **1**. The value can also be given as a symbolic group name.

h*N* Set 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**

H*file* Specify the help file for SMTP.

i Ignore dots in incoming messages. This option is always disabled (dots are always accepted) when reading SMTP mail.

I Insist that the name server be running to resolve host names. If this option is not set and the name server is not running, the **/etc/hosts** file [see **hosts**(4)] will be considered complete. In general, you do want to set this option if your **/etc/hosts** file does not include all hosts known to you or if you are using the MX (mail forwarding) feature of the name server. The name server will still be consulted even if this option is not set, but **sendmail** will feel free to resort to reading **/etc/hosts** if the name server is not available. Thus, you should *never* set this option if you do not run the name server.

j Send error messages in MIME format. (See RFC1341 and RFC1344 for details.)

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. *path* can also be set to a sequence of paths separated by colons; **sendmail** stops at the first file that it can successfully and safely open. For example, **/var/forward/$u:$z/.forward** will search first in **/var/forward/** *username* and then (only if the first file does not exist) in *~username*/**.forward**.

k*N* Set the maximum number of open connections that will be cached at a time. The default is one. This setting delays closing the current connection until either this invocation of **sendmail** needs to connect to another host or **sendmail** terminates. Setting *N* to zero defaults to the old behavior; connections are closed immediately.

K*timeout* Set 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 it uses a cached connection, **sendmail** always sends a **NOOP** (no operation) command to check the connection; if this command fails, **sendmail** reopens the connection. This option 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.

l If there is an **Errors-To** header, send error messages to the addresses listed there. Error messages normally go to the envelope sender. Use of this option causes **sendmail** to violate RFC 1123.

L*n* Set the default log level to *n*. Defaults to **9**

m Send to me, too, even if I am in an alias expansion.

M*x value* Set the macro *x* to *value*. This option is intended for use only from the command line.

n Validate the RHS of aliases when rebuilding the **aliases**(4) database.

o Assume that the headers may be in old format, that spaces delimit names. This option actually turns on an adaptive algorithm: if any recipient address contains a comma, parenthesis, or angle bracket, commas will be assumed already to exist. If this flag is not on, only commas delimit names. Headers are always output with commas between the names.

O*options* Set server SMTP options. The options are *key=value* pairs. Known keys are

Port	Name / number of listening port (defaults to **smtp**)
Addr	Address mask (defaults to **INADDR_ANY**)
Family	Address family (defaults to **INET**)
Listen	Size of listen queue (defaults to **10**)

The **Addr**ess mask may be a numeric address in dot notation or a network name.

p *opt,opt,...* Set the privacy *opt*ions. "Privacy" is really a misnomer; many of these are just a way of insisting on stricter adherence to the SMTP protocol. These *opt*ions can be selected:

public	Allow open access.
needmailhelo	Insist on **HELO** or **EHLO** command before **MAIL**.
needexpnhelo	Insist on **HELO** or **EHLO** command before **EXPN**.
noexpn	Disallow **EXPN** entirely.
needvrfyhelo	Insist on **HELO** or **EHLO** command before **VRFY**.
novrfy	Disallow **VRFY** entirely.
restrictmailq	Restrict **mailq** command.

restrictqrun	Restrict –**q** command line flag.
goaway	Disallow essentially all SMTP status queries.
authwarnings	Put X-Authentication-Warning headers in messages.
tsoladminlowupgrade	Upgrade mail to user min label.
tsoladminlowaccept	Deliver mail at ADMIN_LOW.
tsoladminlowreturn	Return ADMIN_LOW mail to sender.
tsolotherlowupgrade	Upgrade mail to user min label.
tsolotherlowaccept	Deliver mail below user min label.
tsolotherlowreturn	Return mail below user min label to sender. (default)

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. Authentication Warnings add warnings about various conditions that may indicate attempts to spoof the mail system, such as using a nonstandard queue directory.

The **tsol** options set the desired action when a message is received at a sensitivity label of ADMIN_LOW or at some other sensitivity label below the recipient's minimum sensitivity label. In each case, there are three options that specify the possible actions that may taken. **upgrade** means to deliver the message at the recipient's minimum sensitivity label; **accept** means to deliver the message at the message's sensitivity label; **return** means to return the message to the sender.

P*postmaster* Send copies of error messages to the named *postmaster*. Only the header of the failed message is sent. Because most errors are user problems, this option is probably not a good idea on large sites and arguably contains all sorts of privacy violations; but this option seems to be popular with certain operating systems vendors.

q*factor* Use *factor* as the multiplier in the map function to decide when to queue up jobs rather than run them. This value is divided by the difference between the current load average and the load average limit (**x** flag) to determine the maximum message priority that will be sent. Defaults to **600000**

Q*dir* Use the named *dir* as the queue directory.

r *timeouts* Timeout reads after *time* interval. The *timeouts* argument is a list of *keyword=value* pairs. These are the recognized timeouts, their default values, and their minimum values specified in RFC 1123 section 5.3.2:

initial	Wait for initial greeting message. [5m, 5m]
helo	Reply to **HELO** or **EHLO** command. [5m, none]
mail	Reply to **MAIL** command. [10m, 5m]
rcpt	Reply to **RCPT** command. [1h, 5m]

datainit	Reply to **DATA** command. [5m, 2m]
datablock	Data block read [1h, 3m]
datafinal	Reply to final "." in data. [1h, 10m]
rset	Reply to **RSET** command. [5m, none]
quit	Reply to **QUIT** command. [2m, none]
misc	Reply to **NOOP** and **VERB** commands. [2m, none]
command	Command read [1h, 5m]
ident	**IDENT** Protocol timeout [30s, none]

All but **command** apply to client SMTP. For backward compatibility, a timeout with no *keyword=* part will set all of the longer values.

s Be super-safe when running things: 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 in any circumstances.

S*file* Log statistics in the named *file*.

t*tzinfo* Set the local time zone info to *tzinfo*—for example, "PST8PDT." Actually, if this option is not set, the **TZ** environment variable is cleared (so the system default is used); if the option is set but null, the user's **TZ** variable is used; if the option is set and not null, the **TZ** variable is set to this value.

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 *wtime* is missing or zero, then no warning messages are sent.

u*n* Set the default user ID for mailers to *n*. Mailers without the *S* flag in the mailer definition will run as this user. Defaults to **1**. The value can also be given as a symbolic user name.

v Run in verbose mode. If this mode is set, **sendmail** adjusts options **c** (don't connect to expensive mailers) and **d** (delivery mode) so that all mail is delivered completely in a single job so that you can see the entire delivery process. Option **v** should *never* be set in the configuration file; this option is intended for command-line use only.

V*fallbackhost* The *fallbackhost* acts like a very low priority MX on every host. This option is intended to be used by sites with poor network connectivity.

w If you are the "best" (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 connecting directly to that host. 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 *not* recommended.

x*LA* When the system-load average exceeds *LA*, just queue messages; don't try to send them. Defaults to **8**

X*LA* When the system-load average exceeds *LA*, refuse incoming SMTP connections. Defaults to **12**

y*fact* The indicated factor (*fact*) is added to the priority (thus *lowering* the priority of the job) for each recipient; this value penalizes jobs with large numbers of recipients. Defaults to **30000**

Y Deliver each job that is run from the queue in a separate process. Use this option if you are short of memory because the default tends to consume considerable amounts of memory while the queue is being processed.

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**

Z*fact* The 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 factor should be positive because hosts that are down are all too often down for a long time. Defaults to **90000**

7 Strip input to seven bits for compatibility with old systems. This option should not be necessary.

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 relinquishing are **b**, **d**, **e**, **E**, **i**, **L**, **m**, **o**, **p**, **r**, **s**, **v**, **C**, and **7**. **M** (define macro) when defining the **r** or **s** macros is also considered "safe."

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 between arguments.

If invoked as **newaliases**, **sendmail** rebuilds the alias database. **newaliases** must be invoked from the trusted path, must be used only at sensitivity label ADMIN_LOW, and must inherit the same privileges as for the **–bd** option. If invoked as **mailq**, **sendmail** prints the contents of the mail queue. Only messages queued at sensitivity labels dominated by the process are printed.

RETURN VALUES

sendmail returns an exit status describing what it did. These 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
X_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

Options –**ba**, –**bd**, –**bi**, –**bd**, –**bs**, –**bt**, –**bv**, –**M**, and –**q** options require that **sendmail** be invoked from the trusted path 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.

FILES

dead.letter	Unmailable text
/etc/mail/sendmail.cf	Defines environment for **sendmail**
/var/spool/mqueue/∗	Multilevel directory containing temp files and queued mail
~/.forward	List of recipients for forwarding messages

SEE ALSO

biff(1B), **mail**(1), **mailx**(1), **newaliases**(1), **aliases**(4) **hosts**(4)

Su, Zaw-Sing, and Jon Postel, *The Domain Naming Convention for Internet User Applications*, RFC 819, Network Information Center, SRI International, Menlo Park, CA, August 1982.

Postel, Jon, *Simple Mail Transfer Protocol*, RFC 821, Network Information Center, SRI International, Menlo Park, CA, August 1982.

Crocker, Dave, *Standard for the Format of* ARPA-Internet *Text* Messages, RFC 822, Network Information Center, SRI International, Menlo Park, CA, August 1982.

NAME

setaudit – run a command with the audit mask set

SYNOPSIS

setaudit[–**u** *username*] *command command_args*

AVAILABILITY

SUNWtsolu

DESCRIPTION

setaudit invokes a command using the audit characterisitcs 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 and **sys_audit** privileges in its set of effective privileges.

EXAMPLES

To execute the **cat** command on the file **/etc/system** as the user **maverick**, use this:

setaudit -u maverick /usr/bin/cat /etc/system

To execute the **ls** command on the current working directory from the system shell, use the following command. The command will execute with the audit characterisitcs of the owner of the invoking shell:

setaudit /sbin/sysh -c ls

SEE ALSO

audit_control(4TSOL), **audit_user**(4TSOL)

NAME

setauditpsa – modify the audit preselection attributes for a file or files

SYNOPSIS

setauditpsa –**s** *audit_entries file* ...

setauditpsa –**m** *audit_entries file* ...

setauditpsa –**d** *audit_entries file* ...

setauditpsa –**f** *audit_entries file* ...

AVAILABILITY

Available only on Trusted Solaris 2.x systems

DESCRIPTION

For each file specified, **setauditpsa** either replaces entirely the file's audit preselection attributes, or it adds, modifies, or deletes one or more audit preselection flag entries.

The –**s** option sets the audit preselection attributes to the entries specified on the command line. The –**f** option sets the audit preselection attributes to the entries contained within the file *audit_file*. The –**d** option deletes one or more specified entries from the file's audit preselection attributes. The –**m** option adds or modifies one or more specified audit preselection flag entries.

One of the options –**s**, –**m**, –**d**, or –**f** must be specified. If –**s** or –**f** are specified, other options are invalid. The –**m** and –**d** options may be combined.

For the –**m** and –**s** options, *audit_entries* are one or more comma-separated audit preselection flag entries selected from the following list. For the –**f** option, *audit_file* must contain audit preselection flag entries, one to a line, selected from the following list. Boldface indicates that characters must be typed as specified, brackets denote optional characters, and italicized characters are to be specified by the user.

u[**ser**]::*success-audit:failure-audit:success-alarm:failure-alarm*

u[**ser**]:*uid:success-audit:failure-audit:success-alarm:failure-alarm*

g[**roup**]::*success-audit:failure-audit:success-alarm:failure-alarm*

g[**roup**]:*gid:success-audit:failure-audit:success-alarm:failure-alarm*

o[**ther**]:*success-audit:failure-audit:success-alarm:failure-alarm*

For the –**d** option, *audit_entries* are one or more comma-separated audit preselection flag entries without **success-audit**, **failure-audit**, **success-alarm**, and **failure-alarm**, selected from the following list.

u[**ser**]:

g[**roup**]:

u[**ser**]:*uid*

g[**roup**]:*gid*

o[**ther**]:

The **success-audit** field denotes which successful operations are audited. The **failure-audit** field denotes which failed operations are audited. The **success-alarm** field denotes which successful operations are alarmed. The **failure-alarm** field denotes which failed

operations are alarmed.

success-audit, **failure-audit**, **success-alarm**, and **failure-alarm**, are each 3 character strings composed of the letters [a | n | d][a | n | d][a | n | d]. The first character in each field is for read access the second character in each field is for write access and the third character in each field is for execute access. Each character position can have one of the follwing values:

[a] always

[n] never

[d] use process default

This allows for any [**successful | failed**] file [**read | write | execute**] access to be [**audited | alarmed**] to the granularity of a [**owner | group owner | user | group | other**].

uid is a login name or user ID.

gid is a group name or group ID.

The options have the following meaning:

–s Set a file's audit preselection attributes. All old audit preselection attributes entries are removed and replaced with the newly-specified audit preselection attributes. There must be exactly one user entry specified for the owner of the file, exactly one group entry for the owning group of the file, and exactly one other entry specified. There must not be duplicate user entries with the same uid, or duplicate group entries with the same gid. The entries need not be in any specific order. They are sorted by the command before being applied to the file.

–m Add one or more new audit preselection flag entries to the file, and/or modify one or more existing audit preselection flag entries on the file. If an entry already exists for a specified uid or gid, the specified **success-audit**, **failure-audit**, **success-alarm**, and **failure-alarm** replaces the **success-audit**, **failure-audit**, **success-alarm**, and **failure-alarm**. If an entry does not exist for the specified uid or gid, an entry is created.

–d Delete one or more entries from the file.

–f Set a file's audit preselection attributes with the audit preselection flag entries contained in the file named *audit_file*. The same constraints on specified entries hold as with the **–s** option. The entries are not required to be in any specific order in the file. The character "#" in audit_file may be used to indicate a comment. All characters, starting with the "#", until the end of the line, are ignored. Note that if the *audit_file* has been created as the output of the **getauditpsa** command, any effective permissions, which follow a "#", are ignored.

EXAMPLES

To add one audit preselection flag entry to file "foo", adding user "shea" to audit only on successful read, enter:

```
setauditpsa -m user:shea:add:ddd:ddd:ddd foo
```

To set the same file audit preselection attributes on file "foo" as the file "bar", type:

getauditpsa bar | setauditpsa -f -

FILES

/etc/passwd
/etc/group

SEE ALSO

getauditpsa(1MTSOL), **apsacheck**(3TSOL), **apsasort**(3TSOL)

NAME

setfsattr, newsecfs – set security attributes on an existing or newly created file system

AVAILABILITY

SUNWtsolu

SYNOPSIS

/usr/sbin/setfsattr { [–**a** *access-acl*] | [–**d** *default-acl*] | [–**f** *attribute-flags*] | [–**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*] | [–**d** *default-acl*] | [–**f** *attribute-flags*] | [–**l** *sensitivity-level-range*] | [–**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. A normal user is not allowed to run **newsecfs**. When run without options, **newsecfs** behaves as if invoked with –**f tsol_attr**.

OPTIONS

–**a** *access-acl* Set the file system access ACL. The specified ACL must be a valid access ACL.

–**d** *default-acl* Set the file system default ACL. The specified ACL must be a valid default ACL.

–**f** *attribute-flags* Set the attribute-flags, specified as comma-separated text strings. Currently the only valid string is **tsol_attr**, a one-way flag that cannot be unset without recreating the file system. If any other options are specified, *attribute-flags* is automatically set for **tsol_attr** since the flag is required for setting attributes.

–**l** *sensitivity-level-range* Set the file system 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.

–**m** *MLD-prefix* Set the file system MLD prefix. The default is ".MLD.". The MLD prefix is the string that disables multilevel directory translation.

–**o** *newfs options* Set the file system **newfs** options. The options must be exactly the same as those expected by the **newfs**(1) 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 file system 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 file system CMW label. The file system sensitivity level range must have either been previously initialized or been included in the arguments of the current or a previous invocation of this command.

Except for the –**f** option, specifying any option also sets the **tsol_attr** *attribute-flags.*

EXAMPLES

To set an access ACL, use this command:

example% setfsattr -a user:foo:rw-,user::rwx,group::r--,mask::rw-,other::--- filename

To preview the initialization of **/rawdevice**, use this command:

example% newsecfs -l admin_low;admin_high -s admin_low[admin_high]

RETURN VALUES

setfsattr exits with one of these values:

0 The setting of the security attributes was successful.

1 The setting of the security attributes was unsuccessful.

SEE ALSO

mkfs(1M), **newfs**(1M), **fork**(2TSOL), **terminfo**(4)

NAME

setuname – Change machine information

SYNOPSIS

setuname [**–t**][**–n** *node*][**–s** *name*]

AVAILABILITY

SUNWcsu

DESCRIPTION

setuname 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

–t Temporary change. No attempt will be made to create a permanent change.

–n *node* Change 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* Change the system name. *name* specifies the new system name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

SUMMARY OF TRUSTED SOLARIS CHANGES

If a user other than root issues this command, these privileges are needed in order to update the **/etc/nodename** and the **/etc/rc2.d/S18setuname** files: **file_dac_read**, **file_dac_write**, **file_mac_read**, and **file_mac_write**.

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_nfs – Make local NFS file systems available for mounting by remote systems

SYNOPSIS

share [–**F nfs**] [–**o** *specific_options*] [–**d** *description*] *pathname*

AVAILABILITY

SUNWcsu

DESCRIPTION

The **share** command 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

–**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 sharing will be read-write to all clients by default. *specific_options* can be any combination of the options listed and explained in *specific_options* **for -o**.
–**d** *description*	Provide a comment that describes the file system to be shared.

specific_options for -o

aclok	Allow 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 unauthenticated users (**AUTH_DES** or **AUTH_KERB** authentication), or to be root if **AUTH_UNIX** authentication is used. By default, unknown users are given the effective user ID **UID_NOBODY**. If *uid* is set to –**1**, access is denied.
kerberos	Clients must use the **AUTH_KERB** authentication of RPC to be authenticated. **AUTH_UNIX** authentication is the default. See the **anon=***uid* option for information about how unauthenticated requests are handled.
nosub	Prevent clients from mounting subdirectories of shared directories. For example, if **/export** is shared with the **nosub** option on server **fooey**, then an NFS client will not be able execute this command: **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 ignore silently any attempt to enable the setuid or setgid mode bits.

nodev	By default, clients are allowed to create block and character special devices on the shared file system. Specifying **nodev** 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.
ro	Sharing will be read-only to all clients.
ro=*client*[**:***client*]. . .	Sharing will be read-only to the listed clients. This suboption overrides the **rw** suboption for the clients specified. Netgroup names may be used in place of client names unless the list is used to override an **rw** option.
root=*host*[**:***host*] . . .	Only root users from the specified hosts will have root access. By default, no host has root access.
rw	Sharing will be read-write to all clients.
rw=*client*[**:***client*]. . .	Sharing will be read-write to the listed clients. This suboption overrides the **ro** suboption for the clients specified. Netgroup names may be used in place of client names unless the list is used to override an **ro** option.
secure	Clients must use the **AUTH_DES** authentication of RPC to be authenticated. **AUTH_UNIX** authentication is the default. See the **anon=***uid* option for information about how unauthenticated requests are handled.

OPERANDS

This operand is supported:

pathname	The pathname of the file system to be shared

RETURN VALUES

Upon successful completion, **share_nfs** returns **0**. If an error occurred, **share_nfs** returns **>0**.

SUMMARY OF TRUSTED SOLARIS CHANGES

The **nodev** and **nopriv** options have been added. 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

SEE ALSO

mount(1M), **MOUNT**(1M), **nfsd**(1M), **share**(1M), **unshare**(1M)

NOTES

The command will fail if both **ro** and **rw** are specified for the same client name. If the same client name exists in both the **ro=** and **rw=** lists, the **rw** will override the **ro**, giving read / write access to the client specified.

ro=, **rw=**, and **root=** are guaranteed to work over UDP but may not work over other transport providers.

If a file system is shared with an **ro=** list and a **root=** list, any host that is on the **root=** list will be given only read-only access regardless of whether that host is specified in the **ro=** list unless **rw** is declared as the default or the host is mentioned in an **rw=** list. The same is true if the file system is shared with **ro** as the default. For example, the following **share** commands will give read-only permissions to **hostb**:

share –F nfs –o ro=hosta,root=hostb /var

share –F nfs –o ro,root=hostb /var

The following will give read / write permissions to **hostb**:

share –F nfs –o ro=hosta,rw=hostb,root=hostb /var

share –F nfs –o root=hostb /var

If the file system being shared is a symbolic link to a valid path name, the canonical path (the path that 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** (not **/export/foo**) being the shared path name:

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 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 run only if there is a file-system entry in **/etc/dfs/dfstab** when starting or rebooting the system.

NAME

snoop – Capture and inspect network packets

SYNOPSIS

snoop [–**aPDSvVNC**] [–**d** *device*] [–**s** *snaplen*] [–**c** *maxcount*] [–**i** *filename*]
[–**o** *filename*] [–**n** *filename*] [–**t** [**r** | **a** | **d**]] [–**p** *first* [, *last*]]
[–**x** *offset* [, *length*]] [*expression*]

AVAILABILITY

SUNWcsu

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 either be displayed as they are received or be saved to a file for later inspection.

snoop can display packets in a single-line summary form or in verbose multiline 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 by choosing either of the verbose options.

OPTIONS

–**a**	Listen to packets on **/dev/audio**. (Warning: this can be noisy.)
–**P**	Capture packets in nonpromiscuous mode, in which only broadcast, multicast, or packets addressed to the host machine will be seen.
–**d** *device*	Receive packets from the network using the interface specified by *device*, usually **le0** or **ie0**. When invoked with the –**i** flag, the **netstat**(1MTSOL) program lists all the interfaces that a machine has. Normally, **snoop** will automatically choose the first nonloopback interface it finds.
–**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. There is less chance of dropped packets because of buffer overflow during periods of high traffic. This option also saves disk space when capturing large traces to a capture file. To capture only headers for IP (no options), use a *snaplen* of 34; for UDP, use 42; for TCP, 54; for RPC, 80; for NFS, 120.
–**c** *maxcount*	Quit after capturing *maxcount* packets. Without this option, capturing continues until no disk space is left or until CTRL-C. interrupts.
–**i** *filename*	Display previously captured packets 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 the –**N** flag described hereafter.)

–**o** *filename*	Save captured packets in *filename* as they are captured. During packet capture, display a count of the number of packets saved in the file. If you wish just to count packets without saving to a file, name the file **/dev/null**.
–**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.
–**D**	On the summary line, display the number of packets dropped during capture.
–**S**	On the summary line, display the size, in bytes, of the entire Ethernet frame.
–**t [** r | a | d]	Time-stamp presentation. Time stamps are accurate to within 4 microseconds. The default presents times in (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. Using **r** with the **-p** option displays time relative to any selected packet.
–**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. Its degree of verbosity is halfway between summary mode and verbose mode. Instead of displaying just the summary line for the highest-level protocol in a packet, verbose summary mode displays a summary line for each protocol layer in the packet. For instance, for an NFS packet, verbose summary mode displays 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**
–**p** *first* [, *last*]	Select one or more packets to be displayed from a capture file. The *first* packet in the file is packet #1.
–**x** *offset* [, *length*]	Display packet data in hexadecimal and ASCII 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.
–**N**	Create an IP address-to-name file from a capture file. This option 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.

–C List the code generated from the filter expression for either the kernel packet filter or **snoop**'s own filter.

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, the condition is assumed to be true.

Given a filter expression, **snoop** generates code either for the kernel packet filter or for its own internal filter. If packets are being captured 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 to **snoop**. The kernel packet filter is very efficient because 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 filter expression in quotes. The next section lists and explains the primitives.

Primitives for *expression*

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. For example, "**pinky**" selects packets transmitted to or received from the host pinky; "**pinky and dinky**" selects packets exchanged between hosts pinky AND dinky. Normally the IP address is used. With the **ether** qualifier, for instance, "**ether pinky**", the ethernet address is used.

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; 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 interpretation, prepend a zero when specifying the address. For example, if the Ethernet address is "aa:0:45:23:52:44", specify it by adding a leading zero: "0aa:0:45:23:52:44".
from or **src**	This qualifier modifies the following **host**, **net**, *ipaddr*, *etheraddr*, **port**, or **rpc** primitive to match just the source address, port, or RPC reply.
to or **dst**	This qualifier modifies the following **host**, **net**, *ipaddr*, *etheraddr*, **port**, or **rpc** primitive to match just the destination address, port, or RPC call.
ether	This qualifier 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 in only the source or the destination address.
port *port*	True if either the source or destination port is *port*. The *port* may be either a port number or a 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 as only the source or the 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 procedure 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 either **>**, **<**, **>=**, **<=**, **=**, **!=**; 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, such as multiplication before addition, apply between the arithmetic operators. Parentheses may be used to control the order of evaluation. To use the value of a field in the packet, use this 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 size is 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 LP 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 of **or**; 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 .

RETURN VALUES

Unless **snoop** receives an error signal, its exit status is zero. All abnormal exits return **1**.

EXAMPLES

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 the packets as they are received:

example# snoop funky

Capture packets between **funky** and **pinky** and save the packets 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 (TSOL)
100  0.0046  sunroof -> boutique    NFS R GETATTR OK (TSOL)
101  0.0080  boutique -> sunroof    NFS C RENAME FH=8E6C MTra00192 to .nfs08 (TSOL)
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
```

Now select only those TSOL packets in this capture file:

example$ snoop –i pkts –p99,108 sectype tsol

```
 99  0.0027  boutique -> sunroof    NFS C GETATTR FH=8E6C (TSOL)
100  0.0046  sunroof -> boutique    NFS R GETATTR OK (TSOL)
101  0.0080  boutique -> sunroof    NFS C RENAME FH=8E6C MTra00192 to .nfs08 (TSOL)
```

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,  Sun
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 = 000016430000000100080000305A1C47
NFS:            597A00000080000 2046314AFC450000
NFS:  File name = MTra00192
NFS:  File handle = 000016430000000100080000305A1C47
NFS:            597A00000080000 2046314AFC450000
NFS:  File name = .nfs08
NFS:
```

View only 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 (TSOL)
  2  0.0046    sunroof -> boutique  NFS R GETATTR OK (TSOL)
  3  0.0080   boutique -> sunroof   NFS C RENAME FH=8E6C MTra00192 to .nfs08 (TSOL)
```

Save these packets to a new capture file:

```
snoop –i pkts –o pkts.nfs rpc nfs sunroof boutique
```

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 for Trusted Solaris.

SEE ALSO

netstat(1MTSOL)C , **bufmod**(7M), **dlpi**(7P), **ie**(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, use the **–o** option to output raw packets to a file 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. Because 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, **snoop** 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 required for packet interpretation for higher-level protocols. For complete NFS interpretation, do not set *snaplen* less than 120 bytes.

snoop requires information from an RPC request to interpret an RPC reply fully. 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.

NOTES

snoop requires an interactive interface.

NAME

spray – Spray packets

SYNOPSIS

/usr/sbin/spray [–**c** *count*] [–**d** *delay*] [–**l** *length*] [–**t** *nettype*] *host*

AVAILABILITY

SUNWcsu

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 because it uses unreliable, connectionless transports (upd for example). **spray** can report a large number of packets dropped when the drops were caused by **spray**'s 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 packets. The default is **0**.

–**l** *length* The *length* parameter is the numbers of bytes in the Ethernet packet that holds the RPC call message. Because the data is encoded using XDR, which deals only 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, 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**(3NTSOL) for a description of supported classes.

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

rpc(3NTSOL)

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**

AVAILABILITY

SUNWcsu

DESCRIPTION

swap provides a method of adding, deleting, and monitoring the system swap areas used by the memory manager.

OPTIONS

–**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). Because 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* is 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 that 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 1024-byte blocks) currently allocated for use as backing store

reserved The total amount of swap space (in 1024-bytes blocks) not currently allocated, but claimed by memory mappings for possible future use

used The total amount of swap space (in 1024-byte blocks) that is either allocated or reserved

available The total swap space (in 1024-byte blocks) 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.

SUMMARY OF TRUSTED SOLARIS CHANGES

When used with the **–a** or **–d** option, this command needs the **sys_mount** privilege to succeed.

SEE ALSO

pagesize(1), **mkfile**(1M), **shareall**(1M), **getpagesize**(3C), **vfstab**(4)

WARNINGS

No check is done to see if a swap area being added overlaps with an existing file system.

NAME

sysh – System shell

SYNOPSIS

sysh [–**acefhiknpPrstuvx**] [*argument*...]

AVAILABILITY

SUNWtsolr, SUNWtsolu

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.

The system shell can be run only from a process with the Trusted Path attribute.

USAGE

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 a 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 an ASCII list of the privileges assigned to each command in the command list.
- **-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**.)

SEE ALSO

sh(1), **tsolprof**(4TSOL)

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 not expected to change between minor releases of Trusted Solaris systems, these interfaces may change.

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*

AVAILABILITY

SUNWtsolu

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.

SEE ALSO

rpc.tbootparamd(1MTSOL), **chstate**(2TSOL).

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*]

AVAILABILITY

SUNWtsolu

DESCRIPTION

tnchkdb checks the syntax of the **tnrhtp**(4TSOL), **tnrhdb**(4TSOL), or **tnidb**(4TSOL) 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**.

FILES

/etc/security/tsol/tnidb
/etc/security/tsol/tnrhdb
/etc/security/tsol/tnrhtp

SEE ALSO

tnd(1MTSOL), **tnctl**(1MTSOL), **tnidb**(4TSOL), **tnrhdb**(4TSOL), **tnrhtp**(4TSOL)

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*]

/usr/sbin/tnctl –**I** *tnidb_path*
/usr/sbin/tnctl –**T** *tnrhtp_path*
/usr/sbin/tnctl –**H** *tnrhdb_path*

AVAILABILITY

SUNWtsolu

DESCRIPTION

tnctl provides an interface to send control and configuration messages either to the kernel directly or to **tnd**(1MTSOL).

If a local trusted-networking database file is modified, the administrator should issue **tnchkdb**(1MTSOL) 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**, **-I**, **-T**, and **-H** options, it must have the **sys_net_config** privilege. **tnctl** can be run at any sensitivity label. The **tnctl** executable has permission bits **555**, owner **root**, and group **sys**.

OPTIONS

–**v**	Turn on the 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 log-file 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, perform a **tnrh**(2TSOL) **DELETE** command.
–**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 **WARNING** 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.

FILES

/etc/security/tsol/tnidb
/etc/security/tsol/tnrhtp
/etc/security/tsol/tnrhdb
/etc/nsswitch.conf

SEE ALSO

tninfo(1MTSOL), **tnd**(1MTSOL), **tnchkdb**(1MTSOL), **tnidb**(4TSOL), **tnrhdb**(4TSOL), **tnrhtp**(4TSOL), **nsswitch.conf**(4)

NOTE

Currently, only level-1 debugging is supported.

WARNING

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*]

AVAILABILITY

SUNWtsolu

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**(4TSOL); the remote-host template database, **tnrhtp**(4TSOL); and the interface database, **tnidb**(4TSOL). These databases and their effect on the trusted network are described in their respective man pages. When **tnrhdb**(4TSOL) and **tnrhtp**(4TSOL) 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 changable by using **tnctl**(1MTSOL).

If a local trusted-networking database file is modified, the administrator should issue a **tnchkdb**(1MTSOL) 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.

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 log-file 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 30 minutes.

FILES

/etc/security/tsol/tnidb
/etc/security/tsol/tnrhdb
/etc/security/tsol/tnrhtp
/var/tsol/tndlog
/var/tsol/tnrhtp_c
/var/tsol/tnrhdb_c
/etc/nsswitch.conf

SEE ALSO

tnchkdb(1MTSOL)**, tninfo**(1MTSOL)**, tnctl**(1MTSOL), **tnidb**(4TSOL)**, tnrhdb**(4TSOL)**, tnrhtp**(4TSOL)**, tndlog**(4TSOL)**, nsswitch.conf**(4TSOL)

NAME

tninfo – Print information and statistics about kernel-level network

SYNOPSIS

/usr/sbin/tninfo [–**skc**] [–**i** [*if_name*]] [–**h** [*hostname*]] [–**t** [*template_name*]]

AVAILABILITY

SUNWtsolu

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 with permission bits **555**, owner **root**, and group **sys**.

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= ;**), then that field will not be displayed.

FILES

/etc/security/tsol/tnidb
/etc/security/tsol/tnrhdb
/etc/security/tsol/tnrhtp

SEE ALSO

tnd(1MTSOL), **tnctl**(1MTSOL), **tnidb**(4TSOL), **tnrhdb**(4TSOL), **tnrhtp**(4TSOL)

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*] [**–l** *logfile*]
[**–M** *hostname*] [**–x**]

AVAILABILITY

SUNWtsolu

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.

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.

–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

tokmapd(1MTSOL),

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 not expected to change between minor releases of Trusted Solaris systems, these interfaces may change.

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*]

AVAILABILITY

SUNWtsolu

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.

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
/etc/security/tsol/tokendb.dir
/etc/security/tsol/tokendb.ir
/etc/security/tsol/tokendb.hosts
/var/tsol/tokmapdlog

SEE ALSO

tokmapctl(1MTSOL)

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 not expected to change between minor releases of Trusted Solaris systems, these interfaces may change.

NAME

uadmin – Administrative control

SYNOPSIS

/sbin/uadmin *cmd fcn*

AVAILABILITY

SUNWcsu

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.

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

uadmin(2TSOL)

NAME

unshare_nfs – Make local NFS file systems unavailable for mounting by remote systems

SYNOPSIS

unshare [–**F nfs**] *pathname*

AVAILABILITY

SUNWcsu

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

–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
/etc/dfs/sharetab

SEE ALSO

share(1M)

NOTES

If the file system being unshared is a symbolic link to a valid pathname, the canonical path (the path that 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** (not **/export/foo**) being the unshared path name:

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**).

OPTIONS

c Replace existing current-label home-directory copies.
(The default is to skip over existing copies.)

i Ignore errors encountered.
(The default aborts on error.)

r Replace existing current-label home-directory copies or symbolic links.
This option implies options **c** and **s**.
(The default is to skip over existing copies or symbolic links.)

s Replace existing current-label home-directory symbolic links.
(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

.copy_files
.cshrc
.mailrc
.netscape/bookmarks.html
.link_files
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

NAME

writeaudit – Write an audit record

SYNOPSIS

writeaudit *event* [–**a** *type:value*] . . . [–**f** *type:filename*] . . .

AVAILABILITY

SUNWtsolu

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.

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**(4TSOL)].

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 input in hex (for example, **0xfff**), octal (for example, **0777**), or decimal.

AW_ILABEL Format

AW_ILABEL:*information_label*

information_label must be a valid ASCII–coded information label; for example, **S** or

**0x00000000000000000000000000000000 **
**00000000000000000000000000000000000000000000000000000000000000000 **
00000000000000000000000000

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 ASCII–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

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

For the event, write an **AUE_event** record containing the specified information label:

writeaudit AUE_event -a AW_ILABEL:Confidential

SEE ALSO

audit(2), **auditwrite**(3TSOL), **audit_event**(4TSOL)

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' sensitivity label, this command must have the **priv_sys_trans_label** privilege in its set of effective privileges.

This interface is uncommitted; although not expected to change between minor releases of Trusted Solaris systems, this interface may change.

Index

M

N

S

T

U

V