Trustedd Solaris 2.5 Man Pages: 1BTSOL and 1TSOL User Commands

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

A Sun Microsystems, Inc. Business

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

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 commands and application programs

AVAILABILITY

SUNWman

NOTE

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

DESCRIPTION

Section 1 of the *Trusted Solaris Reference Manual* describes, in alphabetical order, commands available with 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 1TSOL suffix describe commands that are either new or modified to work within Trusted Solaris security policy. An example of a new Trusted Solaris command added to the combined base Solaris, CDE, and Solstice functionality is **getlabel**, which is described on the **getlabel**(1TSOL) man page. The **getlabel** command allows users to see the *label* of a file.

Modified commands are commands from any of the base products that have been modified to work within the Trusted Solaris *security policy*, such as: **tar**, which has a new –**s** option that maintains security attributes, such as labels, on archives. 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 **tar**(1TSOL), also add descriptions for any new features, options, and arguments added to the base.

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 printed Solaris section 1 man pages are in the Solaris reference manual. 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 **tar**(1) man page instead of the modified **tar**(1TSOL) man page, you would enter: **man –s1 tar.** To get a list of sections that contain man pages with the same name, enter **man –l** followed by the name of the man page, for example: **man –l tar**.

Section 1 subsections with their identifying suffixes are defined below:

1B	Solaris commands found only in the *SunOS/BSD Compatibility Package*. Refer to the *Source Compatibility Guide* for more information.
1C	Unmodified Solaris commands for communicating with other systems.
1F	Unmodified Solaris commands associated with *Form and Menu Language Interpreter* (FMLI).
1S	Unmodified Solaris commands specific to the SunOS system.
1TSOL	Trusted Solaris commands that were either added to the base or modified to work within the Trusted Solaris security policy.

OTHER SECTIONS

See these sections of the *Trusted Solaris Reference Manual* for more information.

- Section 1M in this manual for unmodified Solaris system maintenance commands.
- Section 1MTSOL in this manual for Trusted Solaris system administration commands.
- Section 4 of this manual for information on unmodified Solaris file formats.
- Section 4TSOL of this manual for information on Trusted Solaris configuration file formats.
- Section 5 of this manual for descriptions of unmodified Solaris versions of publicly available files and miscellaneous information pages.
- Section 5TSOL of this manual for information on Trusted Solaris macros.
- Section 6 in this manual for unmodified Solaris versions of computer demonstrations.

For tutorial information about these commands and procedures, see:

- *Trusted Solaris user’s document set*
- *Solaris Advanced User’s Guide*
- *Programming Utilities Guide*

Manual Page Command Syntax

Unless otherwise noted, commands described in the **SYNOPSIS** section of a manual page accept options and other arguments according to the following syntax and should be interpreted as explained below.

name [–*option*...] [*cmdarg*...]
where:

[ ]	Surround an *option* or *cmdarg* that is not required.
...	Indicates multiple occurrences of the *option* or *cmdarg*.
name	The name of an executable file.
{ }	The options and / or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.

option (Always preceded by a “–”.)
noargletter ... or,
argletter optarg[**,**...]

noargletter A single letter representing an option without an option-argument. Note that more than one *noargletter* option can be grouped after one “–” (Rule 5, below).

argletter A single letter representing an option requiring an option-argument.

optarg An option-argument (character string) satisfying a preceding *argletter*. Note that groups of *optargs* following an *argletter* must be separated by commas, or separated by a tab or space character and quoted (Rule 8, below).

cmdarg Path name (or other command argument) *not* beginning with “–”, or “–” by itself indicating the standard input.

Command Syntax Standard: Rules

These command syntax rules are not followed by all current commands, but all new commands will obey them. **getopts**(1) should be used by all shell procedures to parse positional parameters and to check for legal options. It supports Rules 3-10 below. The enforcement of the other rules must be done by the command itself.

1. Command names (*name* above) must be between two and nine characters long.
2. Command names must include only lower-case letters and digits.
3. Option names (*option* above) must be one character long.
4. All options must be preceded by “–”.
5. Options with no arguments may be grouped after a single “–”.
6. The first option-argument (*optarg* above) following an option must be preceded by a tab or space character.
7. Option-arguments cannot be optional.
8. Groups of option-arguments following an option must either be separated by commas or separated by tab or space character and quoted (−**o xxx,z,yy** or −**o** "**xxx z yy**").
9. All options must precede operands (*cmdarg* above) on the command line.
10. “––” may be used to indicate the end of the options.
11. The order of the options relative to one another should not matter.
12. The relative order of the operands (*cmdarg* above) may affect their significance in ways determined by the command with which they appear.
13. “–” preceded and followed by a space character should only be used to mean standard input.

Rules for the Display and Entering of Labels

The Trusted Solaris system always displays *label*s in uppercase. Users may enter labels in any combination of uppercase and lowercase. Depending on how the system is configured and how the user is set up, a user may see *information labels* only, *sensitivity labels* only, the complete *CMW label*, or no labels at all in the top frame of each window and in the *trusted stripe*, among other places in the user's workspace. If information labels alone are configured to display, they display alone. If sensitivity labels alone are configured to display, they display within brackets, in the long form (within the window system).

When both the information label and the sensitivity label are displayed, the full name of the *classification* portion of the information label is shown, while the short name of the classification portion of the sensitivity label is shown.

Note If you need to enter labels on the command line, see the expanded **Rules for the Display and Entering of Labels** in **Intro**(1MTSOL).

DEFINITIONS

ACL See *access control list*

Access Control List A type of *discretionary access control* based on a list of entries that the owner can specify for a file or directory. An access control list (ACL) can restrict or permit access to any number of individuals and groups, allowing finer-grained control than provided by the standard UNIX *permission bits*.

Accreditation Range Actually not a range, but a set made up of labels. See *user accreditation range* and *system accreditation range* for more about the two types of accreditation ranges in the Trusted Solaris system.

Allocatable Device A device to which access is controlled in the Trusted Solaris system by making the device allocatable to a single user at a time. Not all devices are allocatable. Allocatable devices include tape drives, floppy drives, audio devices, and CD-ROM devices. (See *device allocation*.)

Authorization A right granted to a user to perform an action that would otherwise not be allowed by the Trusted Solaris *security policy.* Certain commands require the user to have certain authorizations to succeed. Similar to the use of *privilege* on programs.

CDE action A bundling mechanism used in Trusted Solaris to allow one or more commands to be specified for a particular task that in turn may be assigned to one or more users. A CDE action can have a set of options and arguments specified along with each of the command(s) and can use a dialog box to prompt the user for additional arguments. Each CDE action usually has its own icon, is assigned its own set of *security attributes*, and may be specified in an *execution profile.*

CMW Label Consists of an *information label* followed by a *sensitivity label* in brackets, in the form: INFORMATION LABEL [SENSITIVITY LABEL].

Classification The hierarchical portion of a *sensitivity label*, *information label*, or *clearance*, each of which has only one classification. In a sensitivity label assigned to a file or directory, a classification indicates a relative level of protection based on the sensitivity of the information contained in the file or directory. In a clearance assigned to a user and to *process*es that execute applications and commands on behalf of the user, a classification indicates a level of trust.

Clearance The upper bound of the set of labels at which a user may work, whose lower bound is the *minimum label* assigned by the security administrator as the *initial label*. There are two types of clearance, the *user clearance* and the *session clearance*.

Compartments A set of words in a *sensitivity label*, *information label*, or *clearance.* The compartment represents areas of interest or work groups associated with the labels that contain them and with the files that are assigned the labels and the individuals that work with them.

DAC See *discretionary access control.*

Discretionary Access Control The type of access granted or denied by the owner of a file or directory at the discretion of the owner. The Trusted Solaris system provides two kinds of discretionary access controls (DAC): *permission bits* and *access control lists*

Device Allocation A mechanism for protecting the information on an *allocatable device* from access by anybody except the user who allocates the device. Until a device is deallocated, no one but the user who allocated a device can access any information associated with the device. Device clean scripts may be run when the device is deallocated to clean information from the device before the device may be accessed again by another user. For a user to allocate a device, that user must have been granted the device allocation *authorization* by the *security administrator*, and the user process' sensitivity label must be within the device's *label range*. Upon deallocation of a storage device, such as a tape or floppy drive, the system prompts the user to remove the storage media and supplies an information label that the user is prompted to write upon a physical label along with the sensitivity label, for guidance on how the media should be handled, if information labels and sensitivity labels are configured for display.

Dominate When any type of label (*sensitivity label*, *information label*, or *clearance*) has a security level equal to or greater than the security level of another label to which it is being compared, the first label is said to dominate the second. The *classification* of the dominant label must equal or be higher than the classification of the second label, and the dominant label must include all the words (*compartments* and *markings*, if present) in the other label. Two equal labels dominate each other. Sensitivity labels are compared for dominance when MAC decisions are being made. See *strictly dominate*.

Execution Profile A bundling mechanism for commands and *CDE actions* and for the security attributes assigned to the commands and CDE actions. Execution profiles allow Trusted Solaris administrators to control who can execute which commands and to control the attributes these commands have when they are executed. When a user logs in, all execution profiles assigned to that user are in effect, and the user has access to all the commands and CDE actions assigned in all of that user's profiles.

File Access

Because in UNIX systems just about everything (including a spreadsheet, a printer, a letter, a chapter of a book, or a mail message) is handled as a file, which is stored in a directory—to do just about anything the user must access files and directories. The conditions for access are described here. (Even though devices are treated as files in the UNIX system, devices have slightly different mandatory access rules than files or directories, and these rules are separately described in this section.) A file, directory, or device may be accessed in three ways:

- The *name* of the file, directory, or device may be *viewed*,
- The *contents* or the *attributes* of the file, directory, or device may be *viewed*, or
- The *contents* or the *attributes* of the file, directory, or device may be *modified.*

In the Trusted Solaris system, each of these types of access is granted or denied based not only on whether the basic UNIX *discretionary access control* checks have been passed but also on whether the *mandatory access control* checks have been passed.

All types of access require that the *sensitivity label* of the *process dominate*s the sensitivity label of all directories in the pathname and that the owner of the process (the person who executed the command) has discretionary search access for each directory in the pathname. View access to the name of the file, directory or device requires only that this part of the check is passed.

For view access (read access) to the contents or attributes of a file or a directory, the process' sensitivity label must dominate the sensitivity label of the file or directory. For view access to the contents of a device (for example, so you can read information stored on a tape in a tape drive), the process' sensitivity label must be equal to the sensitivity label of the device. The owner of the process also must have discretionary read access to the file, directory, or device.

For a process to write into a file or to modify the file's attributes, the sensitivity label of the file must dominate the sensitivity label of the process and must be within the process' clearance, which is set to be the *session clearance*. For a process to write into a directory (create a file), the sensitivity label of the process must equal the sensitivity label of the directory. For a process to write to a device (for example, store information on a tape in a tape drive), the sensitivity label of the process must also equal the sensitivity label of the device. The owner of the process must have discretionary write access to the file, directory, or device.

For each type of failure of a MAC or DAC check, a specific override *privilege* may be applied to the command, depending on the type of access being denied. A privilege can be made available to a command only by the action of a security administrator, because the security administrator must ensure that the user who executes the command is cleared to, or that the command may be trusted to, use the privilege in a trustworthy manner.

These conditions and the listed override privileges apply to any type of access:

- If the sensitivity label of the process does not dominate the sensitivity label of a directory in the pathname, then the process must have the privilege to search up (search a directory whose sensitivity label dominates the sensitivity label of the process), which is *file_mac_search*.
- If the user executing the comand does not have discretionary search permission for a directory in the pathname, then the process must have the privilege to override search restrictions when accessing a directory, which is *file_dac_search*.

These conditions and the listed override privileges apply to view (read) access:

- If the sensitivity label of the process does not dominate the sensitivity label of a file or equal the sensitivity label of a directory or device, then the process must have the privilege to override MAC read restrictions, which is *file_mac_read*.
- If the user executing the command does not have discretionary read permission for the file or directory, then the process must have the privilege to override DAC read restrictions, which is *file_dac_read*.

These conditions and the listed override privileges apply to modify (write) access:

- If the sensitivity label of file does not dominate or if the sensitivity label of a directory or device does not equal the sensitivity label of the process, the process must have the privilege that overrides MAC write restrictions, allowing the user to write up and to write above the user's clearance, which is *file_mac_write*.
- If the user executing the command does not have discretionary write permission for the file or directory, then the process must have the privilege to override DAC write restrictions, which is *file_dac_write*.

Information Label

A label that signifies the actual security level of the information contained in a file or directory, and which may be used in deciding whether to downgrade the *sensitivity label* of the file or directory, how to physically label information stored on backup media, and how to handle printed output or mail.

Information Label Floating

A conjoining of two *information labels* that occurs when a file or directory with one information label is accessed by a *process* that has another information label, the resulting information label reflects the combined *security level* of both information labels.

Initial Label

The user's *minimum label* set by the security administrator when specifying a user's security attributes, this is the *sensitivity label* of the first workspace that comes up after the user's first login.

Label

A security identifier assigned to a file or directory based on the level at which the information being stored in that file or directory should be protected. Depending on how the *security administrator* has configured the system, users may see the complete *CMW label*, only the *sensitivity label* portion, only the *information label* portion, or no labels at all.

Label Range

A set of sensitivity *label*s assigned to *commands,* file systems, and *allocatable device*s, specified by designating a maximum label and a minimum label. For commands, the minimum and maximum labels limit the sensitivity labels at which the command may be executed. For file systems, the minimum and maximum labels limit the sensitivity labels at which information may be stored on each file system. (Trusted Solaris systems have multilabel file systems configured with a label range from the lowest sensitivity label to the highest sensitivity label. Remote hosts that do not recognize labels are assigned a single sensitivity label, along with any other hosts that the security administrator wishes to restrict to a single label; the label range on a file system mounted from such a host is configured to be restricted to the same sensitivity label as the remote host's sensitivity label.) For allocatable devices, the minimum and maximum labels limit the sensitivity labels at which devices may be allocated and restrict the sensitivity labels at which information can be stored or processed using the device.

MAC

See *mandatory access control.*

MLD

See *multilevel directory*.

Mandatory Access Control

A type of control based on comparing the *sensitivity label* of a file, directory, or device to the sensitivity label of the *process* that is trying to access it. Even though directories and devices are managed like files in the UNIX system, different MAC rules apply to directories and devices than the rules that apply to files. Before a file may be accessed for writing, MAC checks ensure that the sensitivity label of the file dominates the sensitivity label of the process—a policy called *write up.* A process cannot write to a file whose sensitivity label is higher than the process' clearance, which is set to be equal to the *session clearance*. (The write up policy also includes *write equal*.) Before a directory or a device may be accessed for writing, MAC checks ensure that the sensitivity label of the directory or device is equal to the sensitivity label of the process—a policy called *write equal.* Before a file or directory may be accessed for viewing (reading or searching), MAC checks ensure that the sensitivity label of the process dominates the sensitivity label of the file or directory—a policy called *read down.* Before a device may be accessed for viewing, MAC checks ensure that the sensitivity label of the process equals the sensitivity label of the device—a policy called *read equal.* (The read down policy also includes *read equal*.)

The rule that applies when a process at one sensitivity label attempts to read or write a file at another sensitivity label is *write up, read down* (WURD). The rule that applies when a process at one sensitivity label attempts to write a directory at another sensitivity label is *write equal, read down.* The rule that applies when a process at one sensitivity label attempts to write a device at another sensitivity label is *read equal, write equal.*

Markings

The codewords, handling caveats, control and release markings and the associated bits that apply to labeled information, markings are only contained in *information labels*.

Minimum Label

For users, the lower bound of the *sensitivity label*s at which a particular user can work, which is specified as the "initial label" by the security administrator while setting the user's account. For the system, the sensitivity label specified in the minimum label field by the security administrator in the **label_encodings** file sets the lower bound for all users.

Multilevel Directory

A directory in which information at differing *sensitivity labels* is maintained in separate subdirectories called *single-level directories* (SLDs), while appearing to most interfaces to be a single directory under a single name. In the Trusted Solaris system, directories that are used by multiple standard applications to store files at varying labels, such as the **/tmp** directory, **/var/spool/mail** , and users' $HOME directories, are set up to be MLDs. A user working in an MLD sees only files at the sensitivity label of the user's process.

Permission Bits

A type of *discretionary access control* in which the owner specifies a set of bits to signify who can read, write, or execute a file or directory. Three different sets of permissions are assigned to each file or directory: one set for the owner; one set for all members of the group specified for the file or directory; and one set for all others. See also *access control lists.*

Privilege

A right granted to a process executing a command that allows the command or one or more of its options to bypass some aspect of *security policy*. A privilege is only granted by a site's *security administrator* after the command itself or the person using it has been judged to be able to use that privilege in a trustworthy manner.

Profile Mechanism

A mechanism that allows site security administrators to bundle commands, *CDE actions*, and the *security attributes* associated with those commands and actions into an *execution profile*, which may then be assigned to one or more users depending on the tasks that they need to perform.

Process

An action executing a command on behalf of the user who invokes the command, a process receives a number of security attributes from the user, including the the user ID (UID), the group ID (GID), the supplementary group list, and the user's audit ID (AUID). *Security attribute*s received by a process include any privileges available to the command being executed, the process clearance (which is set to be the same as the *session clearance),* the sensitivity label of the current workspace, and an information label. If the option RESET IL ON EXEC is selected [see **system**(4)], the information label is set to be the lowest viewable label in the system when a new process is started. The information label floats if any information at a higher information label is accessed by the process.

SLD

See *single-level directory*.

Security Administrator

In an organization where sensitive information must be protected, the person or persons who define and enforce the site's *security policy* and who are cleared to access all information being processed at the site. In the Trusted Solaris software environment, an administrative role that is assigned to one or more individuals who have the proper clearance and whose task is to configure the security attributes of all users and machines so that the software enforces the site's security policy.

Security Attribute

An attribute used in enforcing the Trusted Solaris *security policy*. Various sets of security attributes, both from the base Solaris and the Trusted Solaris systems, are assigned to *process*es, users, files, directories, file systems, hosts on the trusted network, allocatable devices, and other entities. Security attributes for users from the base Solaris system include the user ID (UID), audit ID (AUID), group ID (GID), supplementary group IDs (SGIDs). Security attributes for users from the Trusted Solaris system include the *clearance, minimum label* (*initial label*), and any *authorization*s. An important Trusted Solaris security attribute for files is the CMW label, the sensitivity label portion of which is used in access decisions and the information label portion of which may be used to track the real sensitivity of the information contained in the file. A *label range* security attribute is assigned to file systems, to allocatable devices and to printers. A UID, GID, a label range, and one or more *privilege*s may be associated with commands and *CDE actions* by security administrators in *execution profiles*. The mentioned security attributes and others are assigned to hosts in Trusted Network databases, which are used to control the security of communications in a Trusted Solaris distributed system.

Security Policy

In the Trusted Solaris environment, the set of DAC, MAC, and information labeling rules that define how information may be accessed. At a customer site, the set of rules that define the sensitivity of the information being processed at that site and the measures that are used to protect the information from unauthorized access.

Sensitivity Label

A security *label* assigned to a file or directory or process, which is used to limit access based on the security level of the information contained therein.

Session Clearance

A *clearance* that is in effect only during a particular login session, this type of clearance is set by the user when starting a session. Each process started during a session has a *process clearance* equal to the session clearance. The session clearance may be set either to be the same as or lower than the *user clearance*.

Single-level Directory

A directory within an MLD containing files at only a single *sensitivity label*. When a user working at a particular sensitivity label changes into an MLD, the user's working directory actually changes to a single-label directory within the MLD, whose sensitivity label is the same as the sensitivity label at which the user is working.

System Accreditation Range

The set of all valid (well-formed) labels created according to the rules defined by each site's security administrator in the **label_encodings** file, plus the two administrative labels that are used in every Trusted Solaris system, ADMIN_LOW and ADMIN_HIGH.

Strictly Dominate

When any type of label (*sensitivity label*, *information label*, or *clearance*) has a security level greater than the security level of another label to which it is being compared, the first label strictly *dominate*s the second label. Strict dominance is dominance without equality, which occurs either when the classification of the first label is higher than that of the second label and the first label contains all the compartments in the second label or when the classifications of both labels are the same while the first label contains all the compartments in the second label plus one or more additional compartments.

Trusted Stripe

A region that cannot be spoofed along the bottom of the screen, which by default provides the following as visual feedback about the state of the window system: a trusted path indicator, the input information label and window sensitivity label. When either *sensitivity label*s or *information label*s are configured to not be viewable for a user, then the type of label that is viewable is displayed and the other is not. When neither *sensitivity label*s nor *information label*s are configured to be displayed for a user, the trusted stripe is reduced to an icon that displays only the trusted path indicator.

User Accreditation Range

The set of all possible labels at which any normal user may work on the system, as defined by each site's security administrator. The rules for well-formed labels that define the *system accreditation range* are additionally restricted by the values specified in the ACCREDITATION RANGE section of the site's **label_encodings**(4TSOL) file: the upper bound, the lower bound, the combination constraints and other restrictions.

User Clearance

The *clearance* assigned by the *security administrator* that sets the upper bound of the set of labels at which one particular user may work at any time. The user may decide to accept or further restrict that clearance during any particular login session, when setting the *session clearance* after log in.

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 ability of the UNIX super-user to bypass access restrictions, to execute restricted commands, and to use some command options not available to other users has been replaced with the *profile mechanism*, which allows the security administrator to assign to various users different sets of commands and to assign different privileges to the commands using *execution profiles*. When a command or one of its options needs a privilege in order to succeed, that privilege is a *required* privilege; if the required privilege is not given to the command in a user's execution profile by the security administrator, the command won't work. 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 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 (usually the root ID 0) or group ID, one that allows access to the file or directory based on its permissions or its ACL.

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 process acting on behalf of a user attempts to access a file or directory, *mandatory access* checks also must be passed. For each possible type of access failure, a specifc override *privilege* 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

getopts(1), **wait**(1), **exit**(2), **getopt**(3C), **wait**(3B),**the** *Trusted Solaris user's document set*, and the *Trusted Solaris administrator's document set.*

DIAGNOSTICS

Upon termination, each command returns two bytes of status, one supplied by the system and giving the cause for termination, and (in the case of "normal" termination) one supplied by the program [see **wait**(3B) and **exit**(2)]. The former byte is 0 for normal termination; the latter is customarily 0 for successful execution and non-zero to indicate troubles such as erroneous parameters, or bad or inaccessible data. It is called variously "exit code", "exit status", or "return code", and is described only where special conventions are involved.

WARNINGS

Some commands produce unexpected results when processing files containing null characters. These commands often treat text input lines as strings and therefore become confused upon encountering a null character (the string terminator) within a line.

Name	**Description**
adornfc(1TSOL)	display the pathname with the final component adorned
at(1TSOL)	execute commands at a later time
atq(1TSOL)	display the jobs queued to run at specified times
atrm(1TSOL)	remove jobs spooled by at or batch
cancel(1TSOL)	See **lp**(1TSOL)

chgrp(1TSOL)	change file group ownership
chmod(1TSOL)	change the permissions mode of a file
chown(1TSOL)	change file ownership
crontab(1TSOL)	user crontab file
disable(1TSOL)	See **enable**(1TSOL)
enable(1TSOL)	enable / disable LP printers
find(1TSOL)	find files
getfacl(1TSOL)	display discretionary file information
getfattrflag(1TSOL)	display the security attributes flag for a file
getfpriv(1TSOL)	display the privileges for a file
getlabel(1TSOL)	display the CMW label of a file
getmldadorn(1TSOL)	display the multilevel directory adornment of a file
getsldname(1TSOL)	display the single-level directory name of a directory
ipcrm(1TSOL)	remove a message queue, semaphore set, or shared memory ID
ipcs(1TSOL)	report status of interprocess communication facilities
ld(1TSOL)	link editor for object files
login(1TSOL)	sign on to the system
lp(1TSOL)	send / cancel requests to an LP print service
lpc(1BTSOL)	line printer control program
lpq(1BTSOL)	display the queue of printer jobs
lpr(1BTSOL)	send a job to the printer
lprm(1BTSOL)	remove jobs from the printer queue
lpstat(1TSOL)	print information about the status of the LP print service
mkdir(1TSOL)	make directories
mldpwd(1TSOL)	display the pathname of the current directory, including adornment and SLD
mldrealpath(1TSOL)	display the pathname of a file or directory, including adornment and SLD
pattr(1TSOL)	get the viewable process attribute flags
pclear(1TSOL)	get process clearance
plabel(1TSOL)	get the CMW label of a process
ppriv(1TSOL)	get the effective privileges of a process
pprivtest(1TSOL)	test the effective privilege set of the process
rm(1TSOL)	remove directory entries

rmdir(1TSOL)	See **rm**(1TSOL)
setfacl(1TSOL)	modify the access control list (ACL) for a file
setfattrflag(1TSOL)	set the security attribute flags for a file
setfpriv(1TSOL)	change the privilege sets for a file
setlabel(1TSOL)	set the CMW label for a file
tar(1TSOL)	create tape archives and add or extract files
testfpriv(1TSOL)	check or test the privilege sets for a file
tfind(1TSOL)	See **find**(1TSOL)
uname(1TSOL)	print name of current system

NAME

adornfc – display the pathname with the final component adorned

SYNOPSIS

adornfc *pathname*

AVAILABILITY

SUNWtsolu

DESCRIPTION

adornfc adorns the final component of *pathname* unless it is already adorned. *pathname* is a path name to a filesystem object.

DIAGNOSTICS

adornfc exits with one of the following values:

0 Success

1 Usage error

2 Failure, error message is the system error number from **adornfc**(3TSOL).

SEE ALSO

adornfc(3TSOL)

NAME

at, batch – execute commands at a later time

SYNOPSIS

at [–c | –k | –s | –p] [–m] [–f *file*] **[–q** *queuename*] **–t** *time*
at [–c | –k | –s | –p] [–m] [–f *file*] **[–q** *queuename*] *timespec. . .*
at –l [–q *queuename*] **[***at_job_id. . .***]**
at –r *at_job_id. . .*

batch

AVAILABILITY

at

SUNWcsu

batch

SUNWesu

DESCRIPTION

at

The **at** utility reads commands from standard input and groups them together as an *at-job*, to be executed at a later time.

The at-job will be executed in a separate invocation of the shell, running in a separate process group with no controlling terminal, except that the environment variables, current working directory, file creation mask (see **umask**(1)), and system resource limits (for **sh**, **ksh**, and**pfsh** only, see **ulimit**(1)) in effect when the **at** utility is executed will be retained and used when the at-job is executed.

When the at-job is submitted, the *at_job_id* and scheduled time are written to standard error. The *at_job_id* is an identifier that will be a string consisting solely of alphanumeric characters and the period character. The *at_job_id* is assigned by the system when the job is scheduled such that it uniquely identifies a particular job.

User notification and the processing of the job's standard output and standard error are described under the **–m** option.

Users are permitted to use **at** and **batch** (see below) if their name appears in the file **/usr/lib/cron/at.allow**. If that file does not exist, the file **/usr/lib/cron/at.deny** is checked to determine if the user should be denied access to **at**. If neither file exists, no user is allowed to submit a job. If only **at.deny** exists and is empty, global usage is permitted. The **at.allow** and **at.deny** files consist of one user name per line.

batch

The **batch** utility reads commands to be executed at a later time. It is the equivalent of the command:

```
at -q b -m now
```

where queue **b** is a special **at** queue, specifically for batch jobs. Batch jobs will be submitted to the batch queue for immediate execution. For Trusted Solaris the **at** and **batch** commands allow a user to create an **atjob** file that is installed in the appropriate SLD that matches the invoking process' sensitivity label. The **at** command also allows a user to list or remove the at-jobs owned by the current user at the invoking process' sensitivity label. A user can list or remove an at-job belonging to another user under either of two conditions. The first condition is when the name of the at-job's owner appears in the **/etc/cron.d/at.admin** file (which contains a list of administrative users for **at**) or is a role user; *and* the invoking user has the TSOL_AUTH_AT_ADMIN authorization. The second condition is when the at-job owner's username neither appears in the **/etc/cron.d/at.admin** file, nor is a role user; *and* the invoking user has the TSOL_AUTH_AT_USER authorization.

OPTIONS

The following options are supported. Note that a user cannot specify the **–c**, **–k**, or **–s** option if the user's login shell is **pfsh**. If neither the **–c**, **–k**, **–s**, nor **–p** option is specified and **pfsh** is the login shell, **pfsh** is used to run the at-job. However, if **pfsh** is not the user's login shell, the SHELL environment variable by default determines which shell to use. If $SHELL is null, **sh** is used by default.

–c C shell. **csh**(1) is used to execute the at-job.

–k Korn shell. **ksh**(1) is used to execute the at-job.

–s Bourne shell. **sh**(1) is used to execute the at-job.

–p Profile shell. **pfsh**(1MTSOL) is used to execute the at-job.

–f *file* Specify the path of a file to be used as the source of the at-job, instead of standard input.

–l (The letter ell.) Report all jobs scheduled for the current user (or if the current user has the appropriate authorizations, report jobs for other users) at the invoking process' sensitivity label, if no *at_job_id* operands are specified. If *at_job_id*s are specified, report only information for these jobs. If the at-job is not owned by the current user, its job information will be displayed under either of two conditions. The first condition is when the username of the at-job's owner appears in the **/etc/cron.d/at.admin** file or is a role user; *and* the current user has the TSOL_AUTH_AT_ADMIN authorization. The second condition is when the username of the at-job's owner neither appears in the **/etc/cron.d/at.admin** file, nor is a role user; *and* the current user has the TSOL_AUTH_AT_USER authorization.

–m Send mail to the invoking user after the at-job has run, announcing its completion. Standard output and standard error produced by the at-job will be mailed to the user as well, unless redirected elsewhere. Mail will be sent even if the job produces no output.

If **–m** is not used, the job's standard output and standard error will be provided to the user by means of mail, unless they are redirected elsewhere; if there is no such output to provide, the user is not notified of the job's completion.

–q *queuename* Specify in which queue to schedule a job for submission. When used with the **–l** option, limit the search to that particular queue. Values for *queuename* are limited to the lower case letters **a** through **z**. By default, at-jobs will be scheduled in queue **a**. In contrast, queue **b** is reserved for batch jobs. Since queue **c** is reserved for cron jobs, it can not be used with the **–q** option.

–r *at_job_id* Remove the jobs with the specified *at_job_id* operands that were previously scheduled by the **at** utility. If the specified *at_job_id* is not owned by the current user, it is removed under either of two conditions. The first condition is when the uusername of the *at_job_id*'s owner appears in the **/etc/cron.d/at.admin** file or is a role user; *and* the current user has the TSOL_AUTH_AT_ADMIN authorization. The second condition is when the username of the *at_job_id*'s owner neither appears in the **/etc/cron.d/at.admin** file, nor is a role user; *and* the current user has the TSOL_AUTH_AT_USER authorization.

–t *time* Submit the job to be run at the time specified by the *time* option-argument, which must have the format as specified by the **touch**(1) utility.

OPERANDS

The following operands are supported:

at_job_id The name reported by a previous invocation of the **at** utility at the time the job was scheduled.

timespec Submit the job to be run at the date and time specified. All of the *timespec* operands are interpreted as if they were separated by space characters and concatenated. The date and time are interpreted as being in the timezone of the user (as determined by the **TZ** variable), unless a timezone name appears as part of *time*, below.

In the "C" locale, the following describes the three parts of the time specification string. All of the values from the **LC_TIME** categories in the "C" locale are recognized in a case-insensitive manner.

time The *time* can be specified as one, two or four digits. One- and two-digit numbers are taken to be hours, four-digit numbers to be hours and minutes. The time can alternatively be specified as two numbers separated by a colon, meaning *hour*:*minute*. An AM / PM indication (one of the values from the **am_pm** keywords in the **LC_TIME** locale category) can follow the time; otherwise, a 24-hour clock time is understood. A timezone name can follow to further qualify the time; see **TZ** on the **environ**(5) manual page. The *time* field can also be one of the following tokens in the "C" locale:

midnight Indicates the time 12:00 am (00:00).

noon Indicates the time 12:00 pm.

now Indicate the current day and time. Invoking **at now** will submit an at-job for potentially immediate execution (that is, subject only to unspecified scheduling delays).

date An optional *date* can be specified as either a month name (one of the values from the **mon** or **abmon** keywords in the **LC_TIME** locale category) followed by a day number (and possibly year number preceded by a comma) or a day of the week (one of the values from the **day** or **abday** keywords in the **LC_TIME** locale category). Two special days are recognized in the "C" locale:

today Indicates the current day.

tomorrow Indicates the day following the current day.

If no *date* is given, **today** is assumed if the given time is greater than the current time, and **tomorrow** is assumed if it is less. If the given month is less than the current month (and no year is given), next year is assumed.

increment The optional *increment* is a number preceded by a plus sign (**+**) and suffixed by one of the following: **minutes**, **hours**, **days**, **weeks**, **months**, or **years**. (The singular forms will be also accepted.) The keyword **next** is equivalent to an increment number of **+ 1**. For example, the following are equivalent commands:

```
at 2pm + 1 week
at 2pm next week
```

USAGE

The format of the **at** command line shown here is guaranteed only for the "C" locale. Other locales are not supported for **midnight**, **noon**, **now**, **mon**, **abmon**, **day**, **abday**, **today**, **tomorrow**, **minutes**, **hours**, **days**, **weeks**, **months**, **years**, and **next**.

Since the commands run in a separate shell invocation, running in a separate process group with no controlling terminal, open file descriptors, traps and priority inherited from the invoking environment are lost.

EXAMPLES

at

1. This sequence can be used at a terminal:

```
$ at –m 0730 tomorrow
sort < file >outfile
<EOT>
```

2. This sequence, which demonstrates redirecting standard error to a pipe, is useful in a command procedure (the sequence of output redirection specifications is significant):

```
$ at now + 1 hour <<!
diff file1 file2 2>&1 >outfile | mailx mygroup
!
```

3. To have a job reschedule itself, **at** can be invoked from within the at-job. For example, this "daily-processing" script named **my.daily** will run every day (although **crontab** is a more appropriate vehicle for such work):

```
# my.daily runs every day
at now tomorrow < my.daily
daily-processing
```

4. The spacing of the three portions of the "C" locale *timespec* is quite flexible as long as there are no ambiguities. Examples of various times and operand presentations include:

```
    at 0815am Jan 24
    at 8 :15amjan24
    at now "+ 1day"
    at 5 pm FRIday
    at '17
          utc+
          30minutes'
```

batch

1. This sequence can be used at a terminal:

```
$ batch
sort <file >outfile
<EOT>
```

2. This sequence, which demonstrates redirecting standard error to a pipe, is useful in a command procedure (the sequence of output redirection specifications is significant):

```
$ batch <<!
diff file1 file2 2>&1 >outfile | mailx mygroup
!
```

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, the **at** command requires the following forced privileges: **proc_audit_tcb**, **file_chown**, and **file_dac_read**.

An ancillary file is created in the **/var/spool/cron/atjobs** directory for each **atjob** file. By convention, the file is named **at_job_id.ad**; and it is used by the clock daemon to set up the at-job to run.

The at-jobs are run with the profile shell if the user's login shell in the passwd entry is the profile shell. Otherwise, the user's specified shell (by −**c**, −**s**, −**k**, or −**p** option), or SHELL environment variable (default **sh** if $SHELL is NULL) is used to run the at-jobs.

ENVIRONMENT

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

SHELL Determine a name of a command interpreter to use to invoke the at-job, when the user's login shell is not **pfsh**. If the variable is unset or **NULL**, **sh** will be used. If it is set to a value other than **sh**, the implementation will use that shell; a warning diagnostic will be printed telling which shell will be used.

TZ Determine the timezone. The job will be submitted for execution at the time specified by *timespec* or −**t** *time* relative to the timezone specified by the **TZ** variable. If *timespec* specifies a timezone, it will override **TZ**. If *timespec* does not specify a timezone and **TZ** is unset or **NULL**, an unspecified default timezone will be used.

DATEMSK If the environment variable **DATEMSK** is set, **at** will use its value as the full path name of a template file containing format strings. The strings consist of format specifiers and text characters that are used to provide a richer set of allowable date formats in different languages by appropriate settings of the environment variable **LANG** or **LC_TIME**. The list of allowable format specifiers is located in the **getdate**(3C) manual page. The formats described in the **OPERANDS** section for the *time* and *date* arguments, the special names **noon**, **midnight**, **now**, **next**, **today**, **tomorrow**, and the *increment* argument are not recognized when **DATEMSK** is set.

EXIT STATUS

The following exit statuses are returned:

0 The **at** utility successfully submitted, removed or listed a job or jobs.

>0 An error occurred, and the job will not be scheduled.

FILES

/usr/lib/cron/at.allow names of users, one per line, who are authorized access to the **at** and **batch** utilities

/usr/lib/cron/at.deny names of users, one per line, who are denied access to the **at** and **batch** utilities

/etc/cron.d/at.admin names of administrative users for **at**, one per line. Do not put roles in this file.

SEE ALSO

crontab(1TSOL), **csh**(1), **date**(1), **ksh**(1), **pfsh**(1MTSOL), **sh**(1), **touch**(1), **ulimit**(1), **umask**(1), **getdate**(3C), **environ**(5)

NOTES

Regardless of queue used, cron has a limit of 100 jobs in execution at any time.

There can be delays in cron at job execution. In some cases, these delays can compound to the point that cron job processing appears to be hung. All jobs will be executed eventually. When the delays are excessive, the only workaround is to kill and restart cron.

NAME

atq – display the jobs queued to run at specified times

SYNOPSIS

atq [–**c**] [–**n**] [*username*. . .]

AVAILABILITY

SUNWcsu

DESCRIPTION

atq displays the **at** jobs queued up for the user at the invoking process' sensitivity label. **at**(1) is a utility that allows users to execute commands at a later date.

If no options are given, the jobs are displayed in chronological order of execution.

When a user invokes **atq** without specifying *username*, the user's at-jobs at the invoking process's sensitivity label are displayed. If the invoking user's name is neither in the **/etc/cron.d/at.admin** file nor a role user *and* the user has the TSOL_AUTH_AT_USER authorization, other users' at-jobs are also displayed.

When a username other than the invoking user's is specified, the named user's at-jobs are displayed under either of two conditions. The first condition is when the specified username is in the **/etc/cron.d/at.admin** file (which contains a list of administratives users for **at**) or is a role user; *and* the invoking user has the TSOL_AUTH_AT_ADMIN authoriztion. The second condition is when the specified username is neither in the **/etc/cron.d/at.admin** file, nor a role user; *and* the invoking user has the TSOL_AUTH_AT_USER authorization.

OPTIONS

–**c** Display the queued jobs in the order they were created (that is, the time that the **at** command was given).

–**n** Display only the total number of jobs currently in the queue.

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed, the **atq** command must have the **file_dac_read** privilege in its set of forced privileges.

FILES

/var/spool/cron/atjobs	spool area for at jobs.
/etc/cron.d/at.admin	names of administrative users for at; one per line. Do not put roles in this file.

SEE ALSO

at(1TSOL), **atrm**(1TSOL), **cron**(1MTSOL)

NAME

atrm – remove jobs spooled by at or batch

SYNOPSIS

atrm [**–afi**] [[*job #*] [*user*] . . .]

AVAILABILITY

SUNWcsu

DESCRIPTION

atrm removes any delayed-execution jobs specified by job number that were created with **at**(1TSOL) but have not yet executed—if the jobs are owned by the invoking account at the invoking processes' sensitivity label. The list of these jobs and associated job numbers can be displayed by using **atq**(1TSOL).

atrm removes jobs belonging to another user only if one of the two following conditions is met. The first condition is met when *user* either is one of the special system account names listed in the **/etc/cron.d/at.admin** file or *user* is the name of a role account, and when the invoking account has the **modify at admin** authorization. The second condition is met if the specified *user*'s name is not in the **at.admin** file and it is not the name of a role account, and when the invoking account has the **modify at user** authorization. **atrm** needs the **proc_audit_tcb** privilege to succeed.

OPTIONS

–a All. Remove all unexecuted jobs at the invoking processes' sensitivity label that were created by the invoking user. The at_jobs owned by another user are removed only when either of two conditions described in the **DESCRIPTION** setion are met.

–f Force. All information regarding the removal of the specified jobs is suppressed.

–i Interactive. **atrm** asks if a job should be removed. If you respond with a **y**, the job will be removed.

SUMMARY OF TRUSTED SOLARIS CHANGES

atrm needs the **proc_audit_tcb** privilege to succeed. **atrm** removes jobs only at the sensitivity label of the current process. New **at**-related authorizations are used to control the removal of jobs owned by other users. **atrm** removes jobs belonging to another user only if the account invoking **atrm** both has needed authorizations and the specified *user* name meets additional requirements described in the conditions in the **DESCRIPTION** section.

FILES

/var/spool/cron/atjobs	spool area for at jobs
/etc/cron.d/at.admin	List of default system account names, one per line. Seldom needs to be updated. Never add the names of role accounts to this file.

SEE ALSO

at(1TSOL), **atq**(1TSOL), **cron**(1MTSOL), *UNKNOWN TITLE ABBREVIATION: SYSADMIN2* and *Trusted Solaris Administrator's Procedures*

NAME

chgrp – change file group ownership

SYNOPSIS

chgrp [–**fhRM**] *group file* . . .

AVAILABILITY

SUNWcsu

DESCRIPTION

The **chgrp** utility will set the group ID of the file named by each *file* operand to the group ID specified by the *group* operand.

For each *file* operand, it will perform actions equivalent to the **chown**(2) function, called with the following arguments:

- The *file* operand will be used as the *path* argument.
- The user ID of the file will be used as the *owner* argument.
- The specified group ID will be used as the *group* argument.

Unless **chgrp** is invoked by a process with appropriate privileges, the set-user-ID and set-group-ID bits of a regular file will be cleared upon successful completion; the set-user-ID and set-group-ID bits of other file types may be cleared.

The operating system has a configuration option **{_POSIX_CHOWN_RESTRICTED}**, to restrict ownership changes. When this option is in effect, the owner of the file may change the group of the file only to a group to which the owner belongs. To arbitrarily change owner IDs, this command needs the **file_chown** privilege, whether or not this option is in effect.

OPTIONS

–**f** Force. Do not report errors.

–**h** If the file is a symbolic link, change the group of the symbolic link. Without this option, the group of the file referenced by the symbolic link is changed.

–**R** Recursive. **chgrp** descends through the directory, and any subdirectories, setting the specified group ID as it proceeds. When a symbolic link is encountered, the group of the target file is changed (unless the –**h** option is specified), but no recursion takes place.

–**M** **chgrp** processes all accessible SLDs in multilevel directories as it descends through the directory tree.

OPERANDS

The following operands are supported:

group A group name from the group database or a numeric group ID. Either specifies a group ID to be given to each file named by one of the *file* operands. If a numeric *group* operand exists in the group database as a group name, the group ID number associated with that group name is used as the group ID.

file A path name of a file whose group ID is to be modified.

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0 The utility executed successfully and all requested changes were made.

>0 An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**M** option processes all accessible single-level directories in multilevel directories. To arbitrarily change owner IDs, **chgrp** requires the **file_chown** privilege.

FILES

/etc/group	group file

SEE ALSO

chmod(1TSOL), **chown**(1TSOL), **id**(1M), **chown**(2TSOL), **group**(4), **passwd**(4), **environ**(5)

NAME

chmod – change the permissions mode of a file

SYNOPSIS

chmod [**–fRM**] <*absolute-mode*> *file*. . .
chmod [**–fRM**] <*symbolic-mode-list*> *file*. . .

AVAILABILITY

SUNWcsu

DESCRIPTION

chmod changes or assigns the mode of a file. The mode of a file specifies its permissions and other attributes. The mode may be absolute or symbolic.

Absolute *mode*

An absolute *mode* is specified using octal numbers:

chmod *nnnn file* . . .

where:

n a number from **0** to **7**. An absolute mode is constructed from the OR of any of the following modes:

4000	Set user ID on execution.
20#0	Set group ID on execution if # is **7**, **5**, **3**, or **1**. Enable mandatory locking if # is **6**, **4**, **2**, or **0**. For directories, files are created with BSD semantics for propagation of the group ID. With this option, files and subdirectories created in the directory inherit the group ID of the directory, rather than of the current process. It may be cleared only by using symbolic mode.
1000	Turn on sticky bit. See **chmod**(2).
0400	Allow read by owner.
0200	Allow write by owner.
0100	Allow execute (search in directory) by owner.
0700	Allow read, write, and execute (search) by owner.
0040	Allow read by group.
0020	Allow write by group.
0010	Allow execute (search in directory) by group.
0070	Allow read, write, and execute (search) by group.
0004	Allow read by others.
0002	Allow write by others.
0001	Allow execute (search in directory) by others.
0007	Allow read, write, and execute (search) by others.

Note that the **setgid** bit cannot be set (or cleared) in absolute mode; it must be set (or cleared) in symbolic mode using **g+s** (or **g-s**).

Symbolic *mode*

A symbolic *mode* specification has the following format:

chmod <*symbolic-mode-list*> *file* . . .

where: <*symbolic-mode-list*> is a comma-separated list (with no intervening whitespace) of symbolic mode expressions of the form:
[*who*] *operator* [*permissions*]

Operations are performed in the order given. Multiple *permissions* letters following a single operator cause the corresponding operations to be performed simultaneously.

who zero or more of the characters **u**, **g**, **o**, and **a** specifying whose permissions are to be changed or assigned:

u	user's permissions
g	group's permissions
o	others' permissions
a	all permissions (user, group, and other)

If *who* is omitted, it defaults to **a**, but the setting of the file mode creation mask (see **umask** in **sh**(1) or **csh**(1) for more information) is taken into account. When *who* is omitted, **chmod** will not override the restrictions of your user mask.

operator either +, –, or =, signifying how permissions are to be changed:

+ Add permissions.

If *permissions* is omitted, nothing is added.

If *who* is omitted, add the file mode bits represented by *permissions*, *except* for the those with corresponding bits in the file mode creation mask.

If *who* is present, add the file mode bits represented by the *permissions*.

– Take away permissions.

If *permissions* is omitted, do nothing.

If *who* is omitted, clear the file mode bits represented by *permissions*, *except* for those with corresponding bits in the file mode creation mask.

If *who* is present, clear the file mode bits represented by *permissions*.

= Assign permissions absolutely.

If *who* is omitted, clear all file mode bits; if *who* is present, clear the file mode bits represented by *who*.

If *permissions* is omitted, do nothing else.

If *who* is omitted, add the file mode bits represented by *permissions*, *except* for the those with corresponding bits in the file mode creation mask.

If *who* is present, add the file mode bits represented by *permissions*.

Unlike other symbolic operations, **=** has an absolute effect in that it resets all other bits represented by *who*. Omitting *permissions* is useful only with **=** to take away all permissions.

permission any compatible combination of the following letters:

r	read permission
w	write permission
x	execute permission
l	mandatory locking
s	user or group set-ID
t	sticky bit
u,**g**,**o**	indicate that *permission* is to be taken from the current user, group or other mode respectively.

Permissions to a file may vary depending on your user identification number (UID) or group identification number (GID). Permissions are described in three sequences each having three characters:

User	Group	Other
rwx	**rwx**	**rwx**

This example (user, group, and others all have permission to read, write, and execute a given file) demonstrates two categories for granting permissions: the access class and the permissions themselves.

The letter **s** is only meaningful with **u** or **g**, and **t** only works with **u**.

Mandatory file and record locking (**l**) refers to a file's ability to have its reading or writing permissions locked while a program is accessing that file.

In a directory which has the set-group-ID bit set (reflected as either **-----s---** or **-----l---** in the output of **'ls -ld'**), files and subdirectories are created with the group-ID of the parent directory—not that of current process.

It is not possible to permit group execution and enable a file to be locked on execution at the same time. In addition, it is not possible to turn on the set-group-ID bit and enable a file to be locked on execution at the same time. The following examples, therefore, are invalid and elicit error messages:

chmod g+x,+l *file*
chmod g+s,+l *file*

Only the owner of a file or directory (or a user running the command with the **file_chown** privilege) may change that file's or directory's mode. Only a user invoking the command with the **sys_config** privilege may set the sticky bit on a non-directory file. If the command is invoked without the **sys_config** privilege, **chmod** will mask the sticky-bit but will not return an error. In order to turn on a file's set-group-ID bit, your own group ID must correspond to the file's and group execution must be set.

OPTIONS

The following options are supported:

–f Force. **chmod** will not complain if it fails to change the mode of a file.

–R Recursively descend through directory arguments, setting the mode for each file as described above. When symbolic links are encountered, the mode of the target file is changed, but no recursion takes place.

–M **chmod** processes all single-level directories as it descends multilevel directories.

OPERANDS

The following operands are supported:

mode Represents the change to be made to the file mode bits of each file named by one of the *file* operands; see **DESCRIPTION**.

file A path name of a file whose file mode bits are to be modified.

EXAMPLES

Deny execute permission to everyone:

```
example% chmod a–x file
```

Allow only read permission to everyone:

```
example% chmod 444 file
```

Make a file readable and writable by the group and others:

```
example% chmod go+rw file
example% chmod 066 file
```

Cause a file to be locked during access:

```
example% chmod +l file
```

Allow everyone to read, write, and execute the file and turn on the set group-ID.

```
example% chmod a=rwx,g+s file
example% chmod 2777 file
```

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**M** option processes all accessible single-level directories in multilevel directories. Running the command by a user other than the owner of a file or directory requires the **file_chown** privilege. Setting the sticky bit on a non-directory file requires the **sys_config** privilege.

SEE ALSO

ls(1), **chmod**(2TSOL) **environ**(5)

NOTES

Absolute changes don't work for the set-group-ID bit of a directory. You must use **g+s** or **g-s**.

chmod permits you to produce useless modes so long as they are not illegal (for instance, making a text file executable). **chmod** does not check the file type to see if mandatory locking is meaningful.

If the filesystem is mounted with the *nosuid* option, *setuid* execution is not allowed.

NAME

chown – change file ownership

SYNOPSIS

chown [**–fhRM**] *owner*[:*group*] *file*. . .

AVAILABILITY

SUNWcsu

DESCRIPTION

The **chown** utility will set the user ID of the file named by each *file* to the user ID specified by *owner*, and, optionally, will set the group ID to that specified by *group*.

If **chown** is invoked, without the **file_setid privilege,** to change the ownership of a file, the set-user-ID bit is cleared.

Only the owner of a file (or a user invoking the command with the **file_chown** privilege) may change the owner of that file.

The operating system has a configuration option {**_POSIX_CHOWN_RESTRICTED**}, to restrict ownership changes. When this option is in effect the owner of the file is prevented from changing the owner ID of the file. The command requires the **file_chown** privilege to arbitrarily change owner IDs whether or not this option is in effect.

OPTIONS

The following options are supported:

–f Do not report errors.

–h If the file is a symbolic link, change the owner of the symbolic link. Without this option, the owner of the file referenced by the symbolic link is changed.

–R Recursive. **chown** descends through the directory, and any subdirectories, setting the ownership ID as it proceeds. When a symbolic link is encountered, the owner of the target file is changed (unless the **–h** option is specified), but no recursion takes place.

–M **chown** processes all accessible single-level directories as it descends multilevel directories.

OPERANDS

The following operands are supported:

owner[**:** *group*] A user ID and optional group ID to be assigned to *file*. The *owner* portion of this operand must be a user name from the user database or a numeric user ID. Either specifies a user ID to be given to each file named by *file*. If a numeric *owner* exists in the user database as a user name, the user ID number associated with that user name will be used as the user ID. Similarly, if the *group* portion of this operand is present, it must be a group name from the group database or a numeric group ID. Either specifies a group ID to be given to each file. If a numeric group operand exists in the group database as a group name, the group ID number associated with that group name will be used as the group ID.

file A path name of a file whose user ID is to be modified.

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0 The utility executed successfully and all requested changes were made.

>0 An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**M** option processes all accessible single-level directories in multilevel directories. If **chown** is invoked without the **file_setid** privilege to change the ownership of a file, **chown** clears the file's set-user-ID bit. To arbitrarily change owner IDs, **chown** requires the **file_chown** privilege.

FILES

/etc/passwd	system password file

SEE ALSO

chgrp(1TSOL), **chmod**(1TSOL), **chown**(2TSOL), **passwd**(4), **environ**(5)

NAME

crontab – user crontab file

SYNOPSIS

crontab [*filename*]
crontab [–elr] *username*

AVAILABILITY

SUNWcsu

DESCRIPTION

crontab manages a user's access with **cron** by copying, creating, listing, and removing a **crontab** file at the sensitivity label that matches the sensitivity label of the invoking process. If invoked without options, **crontab** copies the specified file, or the standard input if no file is specified, into a directory that holds all users' crontabs.

crontab Access Control

Users: Access to **crontab** is allowed:

- if the user's name appears in **/etc/cron.d/cron.allow**.
- if **/etc/cron.d/cron.allow** does not exist and the user's name is not in **/etc/cron.d/cron.deny**.

Users: Access to **crontab** is denied:

- if **/etc/cron.d/cron.allow** exists and the user's name is not in it.
- if **/etc/cron.d/cron.allow** does not exist and user's name is in **/etc/cron.d/cron.deny**.
- if neither file exists.

Note that the rules for **allow** and **deny** apply to **root** only if the **allow** / **deny** files exist.

The **allow** / **deny** files consist of one user name per line.

crontab Entry Format

A **crontab** file consists of lines of six fields each. The fields are separated by spaces or tabs. The first five are integer patterns that specify the following:

minute (**0**–**59**),
hour (**0**–**23**),
day of the month (**1**–**31**),
month of the year (**1**–**12**),
day of the week (**0**–**6** with **0**=Sunday).

Each of these patterns may be either an asterisk (meaning all legal values) or a list of elements separated by commas. An element is either a number or two numbers separated by a minus sign (meaning an inclusive range). Note that the specification of days may be made by two fields (day of the month and day of the week). Both are adhered to if specified as a list of elements. See **EXAMPLES**.

The sixth field of a line in a **crontab** file is a string that is executed by the shell at the specified times. A percent character in this field (unless escaped by \ ) is translated to a NEWLINE character.

Only the first line (up to a **'%'** or end of line) of the command field is executed by the shell. Other lines are made available to the command as standard input. Any line beginning with a **'#'** is a comment and will be ignored. The file should not contain blank lines.

The shell is invoked from your **$HOME** directory with an **arg0** of **pfsh,** if your login shell or **$SHELL** is the profile shell; otherwise, **sh** is invoked by default. Users who desire to have their **.profile** executed must explicitly do so in the **crontab** file. **cron** supplies a default environment for every shell, defining **HOME**, **LOGNAME**, **SHELL(=/bin/sh)**, **TZ**, and **PATH**. The default **PATH** for **user** cron jobs is **/usr/bin**; while **root** cron jobs default to **/usr/sbin:/usr/bin**. The default **PATH** can be set in **/etc/default/cron**; see **cron**(1M).

If you do not redirect the standard output and standard error of your commands, any generated output or errors will be mailed to you.

OPTIONS

–e edits a copy of the current user's **crontab** file, or creates an empty file to edit if **crontab** does not exist at the sensitivity label of the invoking process. When editing is complete, the file is installed as the user's **crontab** file.

If a *username* is given, the specified user's **crontab** file is edited, rather than the current user's **crontab** file; this may be done only under either of two conditions. The first condition is if the specified username is in the **/etc/cron.d/cron.admin** file (which contains a list of administrative users for **cron**), or is a role user; *and* the user has the TSOL_AUTH_CRON_ADMIN authorization. The second condition is if the specified username is neither in the **/etc/cron.d/cron.admin** file, nor a role user; *but* the user has the TSOL_AUTH_CRON_USER authorization. The environment variable **EDITOR** or **VISUAL** determines which editor is invoked with the –**e** option when the user is not assigned the profile shell. The default editor is **ed**(1). If the user is assigned the profile shell to run in a restricted environment, the –**e** option determines the editor as follows: if the environment variable is set to be **vi**, the **adminvi** editor is used; if it is set to **dtpad**, the **TSOLdtpad** editor is used; and if neither variable is set, the **adminvi** editor is used. Note that all crontab jobs should be submitted using **crontab**; you should not add jobs by just editing the **crontab** file because **cron** will not be aware of changes made this way.

–l lists the **crontab** file for the current user at the sensitivity label of the invoking process. A user can list another user's **crontab** file under either of two conditions. The first condition is when the specified username is in the **/etc/cron.d/cron.admin** file or is a role user; *and* the user has the TSOL_AUTH_CRON_ADMIN authorization. The second condition is when the specified username is neither in the **/etc/cron.d/cron.admin** file, nor a role user; *and* the user has the TSOL_AUTH_CRON_USER authorization.

–r removes a user's **crontab** (at the invoking process' senstivity label) from the **crontabs** directory. A user can remove another user's **crontab** file under either of two conditions. The first condition is when the specified username is in the **/etc/cron.d/cron.admin** file or is a role user; *and* the user has the TSOL_AUTH_CRON_ADMIN authorization. The second condition is when the specified username is neither in the **/etc/cron.d/cron.admin** file, nor a role user; *and* the user has the TSOL_AUTH_CRON_USER authorization.

EXAMPLES

1. Clean up **core** files every weekday morning at 3:15 am:

```
15 3 * * 1-5 find $HOME -name core 2>/dev/null | xargs rm -f
```

2. Mail a birthday greeting:

```
0 12 14 2 * mailx john%Happy Birthday!%Time for lunch.
```

3. As an example of specifying the two types of days:

```
0 0 1,15 * 1
```

would run a command on the first and fifteenth of each month, as well as on every Monday. To specify days by only one field, the other field should be set to *, for example:

```
0 0 * * 1
```

would run a command only on Mondays.

SUMMARY OF TRUSTED SOLARIS CHANGES

The **crontab** command requires the following forced privileges: **proc_audit_tcb**, **file_owner**, and **proc_setid**.

An ancillary file is created in the **/var/spool/cron/crontabs** directory for each **crontab** file. By convention, the file is named **username.ad**; and it is used by the **clock** daemon to set up the cron job to run.

cron jobs are run with the profile shell if the user's login shell (in the passwd entry) or $SHELL is the profile shell. Otherwise, **sh** is used.

The default Trusted Solaris system has an **/etc/cron.d/cron.deny** file, and an **/etc/cron.d/cron.admin** file.

ENVIRONMENT

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

EDITOR Determine the editor to be invoked when the −**e** option is specified. The default editor is **ed**(1).

EXIT STATUS

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

FILES

/etc/cron.d	main cron directory
/etc/cron.d/cron.allow	list of allowed users
/etc/default/cron	contains cron default settings
/etc/cron.d/cron.deny	list of denied users
/var/cron/log	accounting information
/var/spool/cron/crontabs	spool area for **crontab**.
/etc/cron.d/cron.admin	list of administrative users for crontab; one per line. Do not put roles in this file.

SEE ALSO

atq(1TSOL), **atrm**(1TSOL), **ed**(1), **sh**(1), **cron**(1MTSOL), **su**(1M), **environ**(5)

NOTES

If you inadvertently enter the **crontab** command with no argument(s), do not attempt to get out with CTRL-D. This removes all entries in your **crontab** file. Instead, exit with CTRL-C.

If an authorized user modifies another user's **crontab** file, resulting behavior may be unpredictable. Instead, the authorized user should first **su**(1M) to the other user's login before making any changes to the **crontab** file.

NAME

enable, disable – enable / disable LP printers

SYNOPSIS

/usr/bin/enable *printer ...*
/usr/bin/disable [**–c** **|** **–W**] [**–r** [*reason*]] *printer ...*

AVAILABILITY

SUNWlpu

DESCRIPTION

The **enable** command activates the named *printer*s, enabling them to print requests submitted by the **lp** command. If the printer is remote, the command will only enable the transfer of requests to the remote system; the **enable** command must be run again, on the remote system, to activate the printer. (Run **lpstat** **–p** to get the status of *printer*s.)

The **disable** command deactivates the named *printer*, disabling it from printing requests submitted by **lp**. By default, any requests that are currently printing on the designated printer(s) will be reprinted in their entirety either on the same printer or on another member of the same class of printers. If the printer is remote, this command will only stop the transmission of jobs to the remote system. The **disable** command must be run on the remote system to disable the printer. (Run **lpstat** **–p** to get the status of *printer*s.)

OPTIONS

Options for use with **disable** are:

–c Cancel any requests that are currently printing on the designated printer(s). This option cannot be used with the **–W** option. If the printer is remote, the **–c** option will be silently ignored.

–W Wait until the request currently being printed is finished before disabling the specified printer. This option cannot be used with the **–c** option. If the printer is remote, the **–W** option will be silently ignored.

–r *reason* Assign a *reason* for the disabling of the printer(s). This *reason* applies to all *printer*s specified. This *reason* is reported by **lpstat** **–p**. *reason* must be enclosed in quotes if it contains blanks. The default reason is **unknown reason** for the existing printer, and **new printer** for a printer just added to the system but not yet enabled.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **enable** and **disable** commands requires the **administer_printing** authorization.

FILES

/var/spool/lp/∗

SEE ALSO

lp(1TSOL), **lpstat**(1TSOL)

NAME

find, tfind – Find files

SYNOPSIS

find *path . . . expression*

tfind *path . . . expression*

AVAILABILITY

SUNWcsu

DESCRIPTION

The **find** command recursively descends the directory hierarchy for each *path* seeking files that match a Boolean *expression* written in the primaries given in **OPERANDS**.

find will be able to descend to arbitrary depths in a file hierarchy and will not fail because of path length limitations (unless a *path* operand specified by the application exceeds **PATH_MAX** requirements). The **tfind** command supports execution of commands in restricted environments defined by the profile-shell mechanism. **tfind** contains all the functionality of **find** except for expressions **-exec** *command* and **-ok** *command.* For these expressions **tfind** invokes *command* through the profile shell (**pfsh**).

OPERANDS

The following operands are supported:

path a path name of a starting point in the directory hierarchy.

The first argument that starts with a –, or is a **!** or a **(** and all subsequent arguments will be interpreted as an *expression* made up of the following primaries and operators. In the descriptions, wherever *n* is used as a primary argument, it will be interpreted as a decimal integer optionally preceded by a plus (**+**) or minus (–) sign:

+*n*	more than *n*
n	exactly *n*
–*n*	less than *n*

Expressions

These are valid expressions:

–**atime** *n* True if the file was accessed *n* days ago. The access time of directories in *path* is changed by **find** itself.

–**cpio** *device* Always true; write the current file on *device* in **cpio** format (5120-byte records).

–**ctime** *n* True if the file's status was changed *n* days ago

–**depth** Always true; causes descent into the directory hierarchy so that all entries in a directory are acted on before the directory itself. This option can be useful when **find** is used with **cpio**(1) to transfer files that are contained in directories without write permission.

–**exec** *command* True if the executed *command* returns a zero value as exit status. The end of *command* must be punctuated by an escaped semicolon. A command argument **{}** is replaced by the current path name. If issued from **tfind**, the command is invoked through the profile shell (**pfsh**).

–follow Always true; causes symbolic links to be followed. When following symbolic links, **find** keeps track of the directories visited so that it can detect infinite loops; for example, such a loop would occur if a symbolic link pointed to an ancestor. This expression should not be used with the **–type l** expression.

–fstype *type* True if the file system to which the file belongs is of type *type*

–group *gname* True if the file belongs to the group *gname*. If *gname* is numeric and does not appear in the **/etc/group** file, *gname* is taken as a group ID.

–inum *n* True if the file has inode number *n*

–links *n* True if the file has *n* links

–local True if the file system type is not a remote file-system type as defined in the **/etc/dfs/fstypes** file **nfs** is used as the default remote file-system type if the **/etc/dfs/fstypes** file is not present.

–ls Always true; prints current path name together with its associated statistics including (respectively):

- inode number
- size in kilobytes (1024 bytes)
- protection mode
- number of hard links
- user
- group
- size in bytes
- modification time.

If the file is a special file, the size field will instead contain the major and minor device numbers.

If the file is a symbolic link, the pathname of the linked-to file is printed preceded by '→'. The format is identical to that of **ls –gilds**. [See **ls**(1).]

NOTE: Formatting is done internally without executing the **ls** program.

–M In all multilevel directories (MLD) encountered, search single-level directories (SLDs) that are dominated by the sensitivity label of the process. However, if the effective privilege set of the process contains the **file_mac_read** and **file_mac_search** privileges, search all SLDs. The file system enforces all underlying DAC policies and privilege interpretations.

If –**M** is *not* specified and *path* points to an adorned MLD, traverse only this MLD's SLDs. For all other MLDs encountered, automatically translate to the SLD at the sensitivity label of the process even if **find** is run with all privileges.

If –**M** is *not* specified and *path* points to an unadorned MLD, for this and all other MLDs encountered, automatically translate to the SLD at the sensitivity label of the process even if **find** is run with all privileges.

If –**M** is *not* specified and *path* does not point to an MLD, for all MLDs encountered, automatically translate to the SLD at the sensitivity label of the process even if **find** is run with all privileges.

–**mount** Always true; restricts the search to the file system containing the directory specified; does not list mount points to other file systems

–**mtime** *n* True if the file's data was modified *n* days ago

–**name** *pattern* True if *pattern* matches the current file name. Normal shell file-name-generation characters [see **sh**(1)] may be used. A backslash (****) is used as an escape character within the pattern. The pattern should be escaped or quoted when **find** is invoked from the shell.

–**ncpio** *device* Always true; write the current file on *device* in **cpio** –**c** format (5120 byte records).

–**newer** *file* True if the current file has been modified more recently than the argument *file*

–**nogroup** True if the file belongs to a group not in the **/etc/group** file

–**nouser** True if the file belongs to a user not in the **/etc/passwd** file

–**ok** *command* Like –**exec** except that the generated command line is printed with a question mark first, and is executed only if the user responds by typing **y**

If issued from **tfind**, *command* is invoked through the profile shell (**pfsh**).

–**perm** [–]*mode* The *mode* argument is used to represent file-mode bits. *mode* will be identical in format to the <*symbolic*mode> operand described in **chmod**(1), and will be interpreted as follows. To start, a template will be assumed with all file mode bits cleared. An *op* symbol of

+ will set the appropriate mode bits in the template.

– will clear the appropriate bits.

= will set the appropriate mode bits without regard to the contents of the file-mode creation mask of the process.

The *op* symbol of – cannot be the first character of *mode*; this restriction avoids confusion with the optional leading hyphen. Because the initial mode is all bits off, there are not any symbolic modes that need to use – as the first character.

If the hyphen is omitted, the primary will evaluate as true when the file permission bits exactly match the value of the resulting template.

Otherwise, if *mode* is prefixed by a hyphen, the primary will evaluate as true if at least all the bits in the resulting template are set in the file-permission bits.

–**perm** [–]*onum* True if the file permission flags exactly match the octal number *onum*

[See **chmod**(1).] If *onum* is prefixed by a minus sign (–), only the bits that are set in *onum* are compared with the file-permission flags, and the expression evaluates true if they match.

–print Always true; causes the current path name to be printed

–prune Always yields true. Do not examine any directories or files in the directory structure below the *pattern* just matched. See **EXAMPLES**.

–size *n*[**c**] True if the file is *n* blocks long (512 bytes per block). If *n* is followed by a **c**, the size is in bytes.

–type *c* True if the type of the file is *c*, where *c* is **b**, **c**, **d**, **l**, **m**, **p**, or **f** for block special file, character special file, directory, symbolic link, MLD, FIFO (named pipe), or plain file, respectively.

–user *uname* True if the file belongs to the user *uname*. If *uname* is numeric and does not appear as a login name in the **/etc/passwd** file, *uname* is taken as a user ID.

–xdev Same as the **–mount** primary

Complex Expressions

The primaries may be combined using these operators, listed in order of decreasing precedence:

1) **(** *expression* **)** True if the parenthesized expression is true (Parentheses are special to the shell and must be escaped.)

2) **!** *expression* The negation of a primary (**!** is the unary *not* operator).

3) *expression* **[–a]** *expression* Concatenation of primaries (The AND operation is implied by the juxtaposition of two primaries.)

4) *expression* **–o** *expression* Alternation of primaries (**–o** is the OR operator.)

NOTE: When you use **find** in conjunction with **cpio**, if you use the –**L** option with **cpio**, then you must use the –**follow** expression with **find** and vice versa. Otherwise there will be undesirable results.

If no *expression* is present, –**print** will be used as the expression. Otherwise, if the given expression does not contain any of the primaries –**exec**, –**ok** or –**print**, the given expression will be effectively replaced by

(*given_expression*) –print

The –**user**, –**group**, and –**newer** primaries each will evaluate their respective arguments only once.

EXAMPLES

The following commands are equivalent:

```
example% find .
example% find . -print
```

They both write out the entire directory hierarchy from the current directory.

In your home directory, remove all files named **a.out** or ∗**.o** that have not been accessed for a week:

```
example% find $HOME \ ( −name a.out −o −name ′∗.o′ \
          −atime +7 \ −exec rm {} \ ;
```

Recursively print all file names in the current directory and below, but skip SCCS directories:

```
example% find . −name SCCS −prune −o −print
```

Recursively print all file names in the current directory and below, skipping the contents of SCCS directories but printing the SCCS directory name:

```
example% find . −print −name SCCS −prune
```

The following command is roughly equivalent to the −**nt** extension to **test**(1):

```
example$ if [ -n "$(find file1 -prune -newer file2)" ]; then
          printf %s\\n "file1 is newer than file2"
fi
```

The descriptions of −**atime**, −**ctime**, and −**mtime** use the terminology *n* “24-hour periods.” For example, a file accessed at 23:59 will be selected by

```
example% find . -atime -1 -print
```

at 00:01 the next day (less than 24 hours later, not more than one day ago); the midnight boundary between days has no effect on the 24-hour calculation.

Find files with “abc” in their names; search only those SLDs that have the same senisitivity label as the **find** process:

```
example%  find begin_path -type f -name ’∗abc∗’
```

Find files with “abc” in their names; search all SLDs dominated by the senisitivity label as the **find** process:

```
example%  find begin_path -M -type f -name ’∗abc∗’
```

Find MLDs with “xyz” in their names; search all SLDs dominated by the senisitivity label as the **find** process:

```
example%  find begin_path -M -type m -name ’∗xyz∗’
```

Remove files with “abc” in their names; begin at the current directory and perform the removal through the profile shell (**pfsh**):

```
example% tfind . -type f -name ’*abc*’ -exec rm {} \;
```

ENVIRONMENT

See **environron**(5) for descriptions of the following environment variables that affect the execution of **find**: **LC_COLLATE** , **1LC_CTYPE** , **LC_MESSAGES** , and

EXIT STATUS

If all *path* operands were traversed successfully, **find** returns **0**. If an error occurred, **find** returns **>0**.

SUMMARY OF TRUSTED SOLARIS CHANGES

Modifications to the **find** command deal with multilevel directories. A new –**M** option enables traversing MLDs. A new argument (**m**) for the –**type** option enables selecting the MLD type.

FILES

/etc/passwd	Password file
/etc/group	Group file
/etc/dfs/fstypes	File that registers distributed file-system packages

SEE ALSO

chmod(1), **ls**(1), **sh**(1), **test**(1), **stat**(2TSOL), **umask**(2), **environ**(5)

WARNINGS

These options are obsolete and will not be supported in future releases:

-cpio *device* Always true; write the current file on *device* in **cpio** format (5120-byte records).

-ncpio *device* Always true; write the current file on *device* in **cpio -c** format (5120 byte records).

NOTES

When using **find** to determine files modified within a range of time, one must use the **?time** argument *before* the –**print** argument to keep **find** from giving all files.

NAME

getfattrflag – gets the file's security attributes flag

SYNOPSIS

/usr/bin/getfattrflag *filename* ...
/usr/bin/getfattrflag [**–t**] **–m** [**–p**] *filename* ...
/usr/bin/getfattrflag [**–t**] [**–q** **–m**] | [**–q** **–p**]] | [**–q** **–s**] *filename* ...

AVAILABILITY

SUNWtsolu

DESCRIPTION

getfattrflag displays the security attributes flags of *filename*. To display a file's attributes flag information, you must have DAC read and execute permission to all directories in the path name leading to the file, and MAC read access to the file. If no option is specified, the **–m**, **–p**, and **–s** options are applied by default.

OPTIONS

–m Determine if *filename* is a multilevel directory.

–p Determine if *filename* is a public object. To display the true value of the flag, the process must have the **file_audit** privilege.

–q Quiet mode. This option must be used with one (and only one) of the other options. No verbose output is supplied.

–s Determine if *filename* is a single-level directory.

–t If *filename* is a multilevel directory, this option causes **getfattrflag** to return the flag values for the underlying SLD. Without this option, the flag values for the MLD are returned.

EXAMPLES

getfattrflag does not distinquish between directories and regular files. If no option is specified, **getfattrflag** returns the current value of all flags.

```
example% getfattrflag foo
foo:     is not a multilevel directory, is not a single-level directory, is a public object

example% getfattrflag –p foo
foo:     is a public object

example% getfattrflag –m foo
foo:     is not a multilevel directory
```

RETURN VALUES

getfattrflag exits with one of the following values:

0 True value returned for requested flag.

1 False value return for requested flag.

2 Occurrence of error.

NOTES

Using the **–m** and **–t** options together returns false unless *filename* is a fully adorned pathname to a multilevel directory.

NAME

getfpriv – gets the privileges assigned to files

SYNOPSIS

getfpriv *filename . . .*
getfpriv [–**s**] –**a** *filename . . .*
getfpriv [–**s**] –**f** *filename . . .*

AVAILABILITY

SUNWtsolu

DESCRIPTION

getfpriv gets the privileges associated with each *filename*. With no options, both the forced and allowed sets are displayed. The forced privileges are displayed first followed by the allowed set. The default output is as follows:

filename FORCED :" *p1,p2,p3...* ALLOWED :" *p1,p2,p3...*

The –**s** option is used when **getfpriv** is invoked within the command line of **setfpriv**(1TSOL). The output of the command with the –**s** option is as follows:

p1,p2,p3...

For example, if the allowed privileges need to be set on *file1*, exactly as they were set on *filename*, the command line of **setfpriv** would look like the following:

setfpriv –s –a '**getfpriv** –**s** –**a** *filename.B* ' *file1*

OPTIONS

–**a** Display the privileges in the allowed set only.

–**s** Print the list of privileges in a format suitable for use by **setfpriv**(1TSOL). This option is a modifier and must be used with either the –**a** or –**f** option.

–**f** Display privileges in the forced set only.

EXIT CODES

getfpriv exits with one of the following values:

0 Successful completion of **getfpriv**.

1 Unsuccessful completion of **getfpriv**.

SEE ALSO

setfpriv(1TSOL)

NAME

getlabel – Get the CMW label for files

SYNOPSIS

/usr/bin/getlabel [–**h**] [–**x**] *filename* . . .
/usr/bin/getlabel [–**h**] [–**x**] –**i** *filename* . . .
/usr/bin/getlabel [–**h**] [–**x**] –**s** *filename* . . .
/usr/bin/getlabel [**-h**] –**l** *filename* . . .
/usr/bin/getlabel [**-h**] –**I** *filename* . . .
/usr/bin/getlabel [**-h**] –**S** *filename* . . .
/usr/bin/getlabel [**-h**] –**L** *filename* . . .
/usr/bin/getlabel [**-h**] –**i** –**s** *filename* . . .
/usr/bin/getlabel [**-h**] –**I** –**S** *filename* . . .
/usr/bin/getlabel [**-h**] –**i** –**S** *filename* . . .
/usr/bin/getlabel [**-h**] –**I** –**s** *filename* . . .

AVAILABILITY

SUNWtsolu

DESCRIPTION

getlabel gets the CMW label associated with each *filename*. When options are not specified, the output format of the CMW label is displayed in default format. When the options specified conflict, **getlabel** terminates with an error.

OPTIONS

–**h** Get the label on the symbolic link.

–**i** Get the information label portion from the CMW label associated with the specified file, and display the information label in short form.

–**I** Get the information label portion from the CMW label associated with the specified file, and display the information label in long form.

–**l** Get the CMW label associated with the specified file, and display the CMW label in short form; equivalent to –**i** –**s**.

–**L** Get the CMW label associated with the specified file, and display the CMW label in long form; equivalent to –**I** –**S**.

–**s** Get the sensitivity label portion from the CMW label associated with the specified file, and display the sensitivity label in short form.

–**S** Get the sensitivity label portion from the CMW label associated with the specified file, and display the sensitivity label in long form.

–**x** Get the CMW label associated with the specified file, and display the label in hexadecimal form.

EXIT CODES

getlabel exits with one of the following values:

0 Successful completion of **getlabel**

1 Unsuccessful completion of **getlabel** due to usage error

2 Unable to translate label

3 Unable to allocate memory

NAME

getmldadorn – Display the multilevel directory adornment of the file system

SYNOPSIS

getmldadorn *pathname*

AVAILABILITY

SUNWtsolu

DESCRIPTION

getmldadorn displays the MLD adornment of the file system on which *pathname* resides.

RETURN VALUES

getmldadorn exits with one of these values:

0 Success

1 Usage error

2 Failure [Error message is the system error number from **getmldadorn**(2TSOL)C .]

SEE ALSO

getmldadorn(2TSOL)

NAME

getsldname – display file-system single-level directory name

SYNOPSIS

/usr/bin/getsldname [**–s** *sensitivity_label*] *pathname*

AVAILABILITY

SUNWtsolu

DESCRIPTION

getsldname displays the SLD name associated with the sensitivity label of the current process within the multilevel directory (MLD) referred to by the specified full *pathname*. The final component of *pathname* must be an MLD.

OPTIONS

–s Get the SLD name associated with the sensitivity label provided.

getsldname exits with one of the following values:

0 Success

1 Usage error

2 Failure; error message is the system error number from **getcmwplabel**(2TSOL)

3 Failure; error message is the system error number from **getsldname**(2TSOL)

SEE ALSO

getcmwplabel(2TSOL), **getsldname**(2TSOL)

NAME

ipcrm – Remove a message queue, semaphore set, or shared memory ID

SYNOPSIS

ipcrm [**-l** *slabel*] [**-m** *shmid*] [**-q** *msqid*] [**-s** *semid*] [**-M** *shmkey*] [**-Q** *msgkey*]
[**-S** *semkey*] . . .

AVAILABILITY

SUNWipc

DESCRIPTION

ipcrm removes one or more messages, semaphores, or shared-memory identifiers.

The invoking process must have both mandatory and discretionary access to the IPC or must be suitably privileged.

OPTIONS

The identifiers are specified by these options:

–**l** *label* Use the specified ASCII sensitivity *label* (instead of the current sensitivity label) of the process in conjunction with subsequent **-M**, **-Q**, and **-S** options.

NOTE: The process must still pass the appropriate discretionary and mandatory access checks or must be suitably privileged with the **ipc_owner** and **ipc_mac_write** privileges as necessary.

–**m** *shmid* Remove the shared memory identifier *shmid* from the system. The shared memory segment and data structure associated with it are destroyed after the last detach.

–**q** *msqid* Remove the message queue identifier *msqid* from the system and destroy the message queue and data structure associated with it.

–**s** *semid* Remove the semaphore identifier *semid* from the system and destroy the set of semaphores and data structure associated with it.

–**M** *shmkey* Remove from the system the shared memory identifier, created with key *shmkey* . The shared memory segment and data structure associated with it are destroyed after the last detach.

–**Q** *msgkey* Remove from the system the message-queue identifier, created with key *msgkey* , and destroy the message queue and data structure associated with it.

–**S** *semkey* Remove from the system the semaphore identifier, created with key *semkey* , and destroy the set of semaphores and data structure associated with it.

The details of the removes are described in **msgctl**(2TSOL), **shmctl**(2TSOL), and **semctl**(2TSOL). Use the **ipcs** command to find the identifiers and keys.

EXAMPLE

Remove two message queues using the same key but at different sensitivity labels:

```
ipcrm –l "ts a b" -Q 0x00001ef8 –l "c a" -Q 0x00001ef8
```

SUMMARY OF TRUSTED SOLARIS CHANGES

There is a new option, **-l**, for operating on keys at a specific sensitivity label.

Appropriate privilege is required to override failed access checks.

SEE ALSO

ipcs(1TSOL), **msgctl**(2TSOL), **msgget**(2TSOL), **msgop**(2TSOL), **semctl**(2TSOL), **semget**(2TSOL), **semop**(2TSOL), **shmctl**(2TSOL), **shmget**(2TSOL), **shmop**(2TSOL)

NAME

ipcs – Report status of interprocess communication facilities

SYNOPSIS

ipcs [**-abclmopqst**] [**-C** *corefile*] [**-N** *namelist*]

AVAILABILITY

SUNWipc

DESCRIPTION

ipcs prints information about active interprocess communication facilities. Without *options*, information is printed in short format for message queues, shared memory, and semaphores that are currently active in the system.

If the mandatory (sensitivity label) or discretionary access (**MODE**, see **Column Heads**) check on an in-use IPC object fail, the process must be suitably privileged to obtain the information. If the process lacks the necessary privilege, nothing is output for the object.

The information that is displayed is controlled by the options supplied.

OPTIONS

–m Print information about active shared-memory segments.

–q Print information about active message queues.

–s Print information about active semaphores.

If –**q**, –**m**, or –**s** is specified, information about only those indicated is printed. If none of these three is specified, information about all three is printed subject to these options:

–a Use all print options. (This is a shorthand notation for **-b**, **-c**, **-l**, **-o**, **-p**, and **-t**.)

–b Print information on biggest allowable size: maximum number of bytes in messages on queue for message queues, size of segments for shared memory, and number of semaphores in each set for semaphores. See **Column Headings** for meaning of columns in a listing.

–c Print creator's login name and group name. See below.

–l Print the sensitivity and information label (if enabled) associated with the object. Note that information labels are undefined for message queues.

–o Print information on outstanding usage: number of messages on queue and total number of bytes in messages on queue for message queues and number of processes attached to shared-memory segments.

–p Print process number information: process ID of last process to send a message, process ID of last process to receive a message on message queues, process ID of creating process, and process ID of last process to attach or detach on shared-memory segments. See later discussion.

–t Print time information: time of the last control operation that changed the access permissions for all facilities, time of last **msgsnd** and last **msgrcv** on message queues, time of last **shmat** and last **shmdt** on shared memory, time of last **semop** on semaphores. See later discussion.

–**C** *corefile* Use the file *corefile* in place of **/dev/mem** and **/dev/kmem**. Use a core dump obtained from **savecore**(1M) in place of **/dev/mem** and **/dev/kmem**. Without the –**C** option (default), the running system image is used.

–**N** *namelist* Use the file *namelist* in place of **/dev/ksyms**.

Column Headings

The column headings and the meaning of the columns in an **ipcs** listing follow; the letters in parentheses indicate the options that cause the corresponding heading to appear; “all” means that the heading always appears.

NOTE: These options determine only what information is provided for each facility; they do not determine which facilities are listed.

T (all) Type of the facility:

q Message queue
m Shared-memory segment
s Semaphore

ID (all) The identifier for the facility entry

KEY (all) The key used as an argument to **msgget**, **semget**, or **shmget** to create the facility entry

NOTE: The key of a shared-memory segment is changed to **IPC_PRIVATE** when the segment has been removed until all processes attached to the segment detach it.

MODE (all) The facility access modes and flags: the mode consists of 11 characters that are interpreted as follows. The first two characters are

R A process is waiting on a *msgrcv*.
S A process is waiting on a *msgsnd*.
D The associated shared-memory segment has been removed. It will disappear when the last process attached to the segment detaches it.
C The associated shared-memory segment is to be cleared when the first attach is executed.
– The corresponding special flag is not set.

The next nine characters are interpreted as three sets of three bits each. The first set refers to the owner’s permissions; the next to permissions of others in the user-group of the facility entry; and the last to all others. Within each set, the first character indicates permission to read, the second character indicates permission to write or alter the facility entry, and the last character is currently unused.

The permissions are indicated as follows:

r Read permission is granted.
w Write permission is granted.
a Alter permission is granted.

– The indicated permission is not granted.

OWNER (all) The login name of the owner of the facility entry

GROUP (all) The group name of the group of the owner of the facility entry

CREATOR (a,c) The login name of the creator of the facility entry

CGROUP (a,c) The group name of the group of the creator of the facility entry

CBYTES (a,o) The number of bytes in messages currently outstanding on the associated message queue

QNUM (a,o) The number of messages currently outstanding on the associated message queue

QBYTES (a,b) The maximum number of bytes allowed in messages outstanding on the associated message queue

LSPID (a,p) The process ID of the last process to send a message to the associated queue

LRPID (a,p) The process ID of the last process to receive a message from the associated queue

STIME (a,t) The time the last message was sent to the associated queue

RTIME (a,t) The time the last message was received from the associated queue

CTIME (a,t) The time when the associated entry was created or changed

NATTCH (a,o) The number of processes attached to the associated shared-memory segment

SEGSZ (a,b) The size of the associated shared-memory segment

CPID (a,p) The process ID of the creator of the shared-memory entry

LPID (a,p) The process ID of the last process to attach or detach the shared-memory segment

ATIME (a,t) The time the last attach was completed to the associated shared-memory segment

DTIME (a,t) The time the last detach was completed on the associated shared-memory segment

NSEMS (a,b) The number of semaphores in the set associated with the semaphore entry

OTIME (a,t) The time the last semaphore operation was completed on the set associated with the semaphore entry

LABEL (l) The sensitivity and information labels (if enabled) on the object

SUMMARY OF TRUSTED SOLARIS CHANGES

There is a new option, **–l**, for printing labels attached to an IPC object.

Appropriate privilege is required to override failed access checks.

FILES

/etc/group	Group names
/etc/passwd	User names
/dev/mem	Memory
/dev/ksyms	System namelist

SEE ALSO

msgop(2TSOL), **semop**(2TSOL), **shmop**(2TSOL) **msgop**(2), **semop**(2), **shmop**(2)

NOTES

If the user specifies either the −**C** or −**N** flag, the real and effective UID / GID is set to the real UID / GID of the user invoking **ipcs**.

Things can change while **ipcs** is running; the information it gives is guaranteed to be accurate only when it was retrieved.

NAME

ld – link editor for object files

SYNOPSIS

/usr/ccs/bin/ld [**–a | –r**] [**–b**] [**–G**] [**–i**] [**–m**] [**–s**] [**–t**] [**–V**]
[**–B dynamic | static**] [**–B local**] [**–B reduce**] [**–B symbolic**] [**–d y | n**]
[**–D** *token*] [**–e** *epsym*] [**–F** *name*] [**–f** *name*] [**–h** *name*] [**–I** *name*]
[**–L** *path*] [**–l** *x*] [**–M** *mapfile*] [**–o** *outfile*] [**–Q y | n**] [**–R** *path*]
[**–u** *symname*] [**–Y** P,*dirlist*] [**–z defs | nodefs**] [**–z muldefs**] [**–z noversion**]
[**–z text**] *filename . . .*

AVAILABILITY

SUNWtoo

DESCRIPTION

The **ld** command combines relocatable object files, performs relocation, and resolves external symbols. **ld** operates in two modes, static or dynamic, as governed by the **–d** option. In static mode, **–dn**, relocatable object files given as arguments are combined to produce an executable object file; if the **–r** option is specified, relocatable object files are combined to produce one relocatable object file. In dynamic mode, **–dy**, the default, relocatable object files given as arguments are combined to produce an executable object file that will be linked at execution with any shared object files given as arguments; if the **–G** option is specified, relocatable object files are combined to produce a shared object. In all cases, the output of **ld** is left in **a.out** by default.

If any argument is a library, it is searched exactly once at the point it is encountered in the argument list. The library may be either a relocatable archive or a shared object. For an archive library, only those routines defining an unresolved external reference are loaded. The archive library symbol table (see **ar**(4)) is searched sequentially with as many passes as are necessary to resolve external references that can be satisfied by library members. Thus, the ordering of members in the library is functionally unimportant, unless there exist multiple library members defining the same external symbol. A shared object consists of a single entity all of whose references must be resolved within the executable being built or within other shared objects with which it is linked.

OPTIONS

–a In static mode only, produce an executable object file; give errors for undefined references. This is the default behavior for static mode. **–a** may not be used with the **–r** option.

–r Combine relocatable object files to produce one relocatable object file. **ld** will not complain about unresolved references. This option cannot be used in dynamic mode or with **–a**.

–b In dynamic mode only, when creating an executable, do not do special processing for relocations that reference symbols in shared objects. Without the **–b** option, the link editor creates special position-independent relocations for references to functions defined in shared objects and arranges for data objects defined in shared objects to be copied into the memory image of the executable by the runtime linker. With the **–b** option, the output code may be more efficient, but it will be less sharable.

–G In dynamic mode only, produce a shared object. Undefined symbols are allowed.

–i Ignore **LD_LIBRARY_PATH** setting. This option is useful when an **LD_LIBRARY_PATH** setting is in effect to influence the runtime library search, which would interfere with the link editing being performed.

–m Produce a memory map or listing of the input / output sections, together with any non-fatal multiply defined symbols, on the standard output.

–s Strip symbolic information from the output file. Any debugging information, that is *.debug*, *.line*, and *.stab* sections, and their associated relocation entries will be removed. Except for relocatable files or shared objects, the symbol table and string table sections will also be removed from the output object file.

–t Turn off the warning about multiply defined symbols that are not the same size.

–V Output a message giving information about the version of **ld** being used.

–B dynamic | static
Options governing library inclusion. –**B dynamic** is valid in dynamic mode only. These options may be specified any number of times on the command line as toggles: if the –**B static** option is given, no shared objects will be accepted until –**B dynamic** is seen. See also the –**l** option.

–B local Cause any global symbols, not assigned to a version definition, to be reduced to local. Version definitions can be supplied via a *mapfile*, and indicate the global symbols that should remain visible in the generated object. This option achieves the same symbol reduction as the *auto-reduction* directive available as part of a *mapfile* version definition, and may be useful when combining versioned and non-versioned relocatable objects.

–B reduce When generating a relocatable object, cause the reduction of symbolic information as defined by any version definitions. Version definitions can be supplied via a *mapfile*, and indicate the global symbols that should remain visible in the generated object. By default, when generating a relocatable object, version definitions are only recorded in the output image. The actual reduction of symbolic information will be carried out when the object itself is used in the construction of a dynamic executable or shared object. When creating a dynamic executable or shared object, this option is applied automatically.

–B symbolic In dynamic mode only, when building a shared object, bind references to global symbols to their definitions within the object, if definitions are available. Normally, references to global symbols within shared objects are not bound until runtime, even if definitions are available, so that definitions of the same symbol in an executable or other shared objects can override the object's own definition. **ld** will issue warnings for undefined symbols unless –**z defs** overrides.

–D *token,token,. .*
Print debugging information, as specified by each *token*, to the standard error. The special token *help* indicates the full list of tokens available.

–e *epsym* Set the entry point address for the output file to be that of the symbol *epsym*.

–F *name* Useful only when building a shared object. Specifies that the symbol table of the shared object is used as a "filter" on the symbol table of the shared object specified by *name.*

–f *name* Useful only when building a shared object. Specifies that the symbol table of the shared object is used as an "auxiliary filter" on the symbol table of the shared object specified by *name*.

–h *name* In dynamic mode only, when building a shared object, record *name* in the object's dynamic section. *name* will be recorded in executables that are linked with this object rather than the object's UNIX System file name. Accordingly, *name* will be used by the runtime linker as the name of the shared object to search for at runtime.

–I *name* When building an executable, use *name* as the path name of the interpreter to be written into the program header. The default in static mode is no interpreter; in dynamic mode, the default is the name of the runtime linker, **/usr/lib/ld.so.1**. Either case may be overridden by –**I***name.* **exec** will load this interpreter when it loads the **a.out** and will pass control to the interpreter rather than to the **a.out** directly.

–L *path* Add *path* to the library search directories. **ld** searches for libraries first in any directories specified by the –**L** options, and then in the standard directories. This option is useful only if it precedes the –**l** options to which it applies on the command line. The environment variable **LD_LIBRARY_PATH** may be used to supplement the library search path (see **LD_LIBRARY_PATH** below).

–l *x* Search a library **lib***x***.so** or **lib***x***.a**, the conventional names for shared object and archive libraries, respectively. In dynamic mode, unless the –**B static** option is in effect, **ld** searches each directory specified in the library search path for a file **lib***x***.so** or **lib***x***.a**. The directory search stops at the first directory containing either. **ld** chooses the file ending in **.so** if –**l***x* expands to two files whose names are of the form **lib***x***.so** and **lib***x***.a**. If no **lib***x***.so** is found, then **ld** accepts **lib***x***.a**. In static mode, or when the –**B static** option is in effect, **ld** selects only the file ending in **.a**. A library is searched when its name is encountered, so the placement of –**l** is significant.

–M *mapfile* Read *mapfile* as a text file of directives to **ld**. This option may be specified multiple times. If *mapfile* is a directory then all regular files (as defined by **stat**(2)) within the directory will be processed. See *Linker and Libraries Guide* for description of mapfiles.

–o *outfile* Produce an output object file named *outfile*. The name of the default object file is **a.out**.

−Q y | n Under **−Qy**, an **ident** string is added to the *.comment* section of the output file to identify the version of the link editor used to create the file. This will result in multiple **ld idents** when there have been multiple linking steps, such as when using **ld −r**. This is identical with the default action of the **cc** command. **−Qn** suppresses version identification.

−R *path* A colon-separated list of directories used to specify library search directories to the runtime linker. If present and not null, it is recorded in the output object file and passed to the runtime linker. Multiple instances of this option are concatenated together with each *path* separated by a colon.

−u *symname* Enter *symname* as an undefined symbol in the symbol table. This is useful for loading entirely from an archive library, since initially the symbol table is empty and an unresolved reference is needed to force the loading of the first routine. The placement of this option on the command line is significant; it must be placed before the library that will define the symbol.

−Y P,*dirlist* Change the default directories used for finding libraries. *dirlist* is a colon-separated path list.

−z defs Force a fatal error if any undefined symbols remain at the end of the link. This is the default when building an executable. It is also useful when building a shared object to assure that the object is self-contained, that is, that all its symbolic references are resolved internally.

−z muldefs Allows multiple symbol definitions. By default, multiple symbol definitions occurring between relocatable objects will result in a fatal error condition. This option suppresses the error condition, and allows the first symbol definition to be taken.

−z nodefs Allow undefined symbols. This is the default when building a shared object. When used with executables, the behavior of references to such "undefined symbols" is unspecified.

−z noversion

Do not record any versioning sections. Any version sections or associated *.dynamic* section entries will not be generated in the output image.

−z text In dynamic mode only, force a fatal error if any relocations against non-writable, allocatable sections remain.

ENVIRONMENT

LD_LIBRARY_PATH

A list of directories in which to search for libraries specified with the **−l** option. Multiple directories are separated by a colon. In the most general case, it will contain two directory lists separated by a semicolon:

dirlist1;*dirlist2*

If **ld** is called with any number of occurrences of –**L**, as in:

ld . . . –**L***path1* . . . –**L***pathn* . . .

then the search path ordering is:

dirlist1 path1 . . . *pathn dirlist2* **LIBPATH**

When the list of directories does not contain a semicolon, it is interpreted as *dirlist2*.

LD_LIBRARY_PATH is also used to specify library search directories to the runtime linker. That is, if **LD_LIBRARY_PATH** exists in the environment, the runtime linker will search the directories named in it, before its default directory, for shared objects to be linked with the program at execution.

Note: When running a privileged program, set-user-ID or set-group-ID program, the runtime linker will only search for libraries in any full pathname specified within the executable as a result of a runpath being specified when the executable was constructed, or in **/usr/lib**, **/etc/lib**, **/usr/dt/lib**, **/usr/openwin/lib**, and, if **/etc/security/tsol/rtld** exists, the colon-separated list of directories specified therein. Any library dependencies specified as relative pathnames will be silently ignored.

LD_OPTIONS

A default set of options to **ld**. **LD_OPTIONS** is interpreted by **ld** just as though its value had been placed on the command line, immediately following the name used to invoke **ld**, as in:

ld $LD_OPTIONS . . . *other-arguments* . . .

LD_PRELOAD

A list of shared objects that are to be interpreted by the runtime linker. The specified shared objects are linked in after the program being executed and before any other shared objects that the program references.

Note: When running a set-user-ID or set-group-ID program, this option has some restrictions. The runtime linker will only search for these shared objects in any full pathname specified within the executable as a result of a runpath being specified when the executable was constructed, or in **/usr/lib**. Any shared object specified as a relative, or full pathname, will be silently ignored.

LD_RUN_PATH

An alternative mechanism for specifying a runpath to the link editor (see **-R** option). If both **LD_RUN_PATH** and the **-R** option are specified, **-R** supersedes.

LD_DEBUG

Provide a list of tokens that will cause the runtime linker to print debugging information to the standard error. The special token *help* indicates the full list of tokens available. The environment variable **LD_DEBUG_OUTPUT** may also be supplied to specify a file to which the debugging information is sent. The filename will be suffixed with the process id of the application generating the debugging information.

LD_PROFILE

A shared object that will be profiled by the runtime linker. When profiling is enabled, a profiling buffer file is created and mapped. The name of the buffer file is the name of the shared object being profiled with a **`.profile´** extension. By default this buffer is placed under **/var/tmp**. The environment variable **LD_PROFILE_OUTPUT** may also be supplied to indicate an alternative directory in which to place the profiling buffer. This buffer contains **profil**(2) and call count information similar to the *gmon.out* information generated by programs that have been linked with the **-xpg** option of **cc**. Any applications that use the named shared object and run while this environment variable is set, will accumulate data in the profile buffer. The profile buffer information may be examined using **gprof**(1).

Note that environment variable-names beginning with the characters **`LD_´** are reserved for possible future enhancements to **ld**.

SUMMARY OF TRUSTED SOLARIS CHANGES

For a privileged program, the runtime linker will search these additional libraries: **/usr/lib**, **/etc/lib**, **/usr/dt/lib**, and **/usr/openwin/lib**, and, if **/etc/security/tsol/rtld** exists, the colon-separated list of directories specified therein.

FILES

lib*x***.so**	libraries
lib*x***.a**	libraries
a.out	output file
LIBPATH	usually **/usr/ccs/lib:/usr/lib**

SEE ALSO

as(1), **cc**(1B), **gprof**(1), **ld**(1B), **pvs**(1), **exec**(2), **exit**(2), **profil**(2), **elf**(3E), **end**(3C), **exit**(3C), **a.out**(4), **ar**(4)

Linker and Libraries Guide
Binary Compatibility Guide

NOTES

Options No Longer Supported

The following SunOS 4.*x.y* options do not have any replacement in this release: –**B nosymbolic** (this is now the default if –**B symbolic** is not used), –**d**, –**dc**, and –**dp**, (these are now the default, see –**b** above to override the default), –**M**, –**S**, –**t**, –**x**, –**X**, and –**y***sym*.

The following SunOS 4.*x.y* options are not supported: –**align** *datum*, –**A** *name*, –**D** , –**p**, –**T**[**text**] *hex*, –**T data***hex*. Much of the functionality of these options can be achieved using the –**M***mapfile* option.

Obsolete Options

The following SunOS 4.*x.y* options are obsolete in this release: –**n**, –**N**, and –**z**.

NAME

login – Sign on to the system

SYNOPSIS

login [–p] [–d *device* **] [–h** *hostname* **[** *terminal* **] | –r** *hostname* **[–T] [–U** *uid* **]]**
[*name* **[** *environ* **. . .]]**

AVAILABILITY

SUNWcsu

DESCRIPTION

You use the **login** command at the beginning of each terminal session to identify yourself to the system. **login** is invoked by the system when a connection is first established, after the previous user has terminated the login shell by issuing the **exit** command.

login asks for your user name if it is not supplied as an argument, and your password if appropriate. Where possible, echoing is turned off while you type your password, so it will not appear on the written record of the session.

If there are no lowercase characters in the first line of input processed, **login** assumes the connecting TTY is an uppercase-only terminal. **login** then sets the port's **termio**(7I) options to reflect this condition. This feature will be removed in a Solaris release shipped after January 1, 1997.

If you make any mistake in the login procedure, the message

Login incorrect

is printed and a new login prompt appears. If you make five incorrect login attempts, all five may be logged in **/var/adm/loginlog** if it exists. The TTY line will be dropped.

If password aging is turned on and the password has "aged" [see **passwd**(1) for more information], login is denied with a message to use the desktop to log in and change the password.

After a successful login, accounting files are updated. Device owner, group, and permissions are set according to the contents of the **/etc/logindevperm** file; and the time you last logged in is printed. [See **logindevperm**(4).]

Except for remote logins, **login** asks you to select the sensitivity label (SL) at which you will operate for this terminal session. You must enter a label that you are authorized to use and that is valid for the device.

The user-ID, group-ID, supplementary group list, and working directory are initialized; and the command interpreter (usually **ksh**) is started.

The basic environment (*environ*) is initialized:

HOME=*your-login-directory*
LOGNAME=*your-login-name*
PATH=/usr/bin:
SHELL=*last-field-of-passwd-entry*
MAIL=/var/mail/*your-login-name*
TZ=*timezone-specification*

For Bourne shell and Korn shell logins, the shell executes **/etc/profile** and **$HOME/.profile** if it exists. For C shell logins, the shell executes **/etc/.login**, **$HOME/.cshrc**, and **$HOME/.login**. The default **/etc/profile** and **/etc/.login** files check quotas [see **quota**(1M)], print **/etc/motd**, and check for mail. None of the messages is printed if the file **$HOME/.hushlogin** exists. The name of the command interpreter is set to minus (–), followed by the last component of the interpreter's path name; for example, –**sh**.

If the *login-shell* field in the password file [see **passwd**(4)] is empty, then the default command interpreter, **/usr/bin/sh**, is used. If this field is asterisk (*), the named directory becomes the root directory. At that point, **login** is re-executed at the new level, which must have its own root structure.

The environment may be expanded or modified by supplying additional arguments to **login**, either at execution time or at **login**'s request for your login name. The arguments may take either the form *xxx* or the form *xxx=yyy*. Arguments without an equal sign are placed in the environment as

L*n*=xxx

where *n* is a number starting at 0 and incremented each time a new variable name is required. Variables containing an equal sign (**=**) are placed in the environment without modification. If they already appear in the environment, they replace the older values.

There are two exceptions: the variables **PATH** and **SHELL** cannot be changed to prevent people logged into restricted shell environments from spawning secondary shells that are not restricted. **login** understands simple single-character quoting conventions. Typing a backslash (\ ) in front of a character quotes it and allows the inclusion of such characters as spaces and tabs.

Alternatively, you can pass the current environment by supplying the –**p** flag to **login**. This flag indicates that all currently defined environment variables should be passed, if possible, to the new environment. This option does not bypass any environment variable restrictions mentioned earlier. Environment variables specified on the login line take precedence if a variable is passed by both methods.

To enable remote logins by administrative users (that is, administrative roles), edit the **/etc/default/login** file by inserting a pound sign (#) before the **CONSOLE=/dev/console** entry. See **FILES**.

OPTIONS

–**d** *device*
: **login** accepts a device option, *device. device* is taken as the path name of the TTY port on which **login** is to operate. The use of the device option can be expected to improve **login** performance because **login** will not need to call **ttyname**(3C).

–**h** *hostname* [*terminal*]
: **in.telnetd**(1M) uses this option to pass information about the remote host and terminal type.

–**p**
: Passes environment variables to the login shell.

–**r** *hostname*
: **in.rlogind**(1M) uses this option to pass information about the remote host.

–T **in.rlogind**(1M) uses this option to indicate that the trusted path process attribute is set on the remote host for the process invoking **rlogin**.

–U*uid* **in.rlogind**(1M) uses this option to pass information about the UID of the invoker of **rlogin**. If **uid**and**name**are both passed by **in.rlogind**(1M), the UID of **name** must match the **uid** value or login is denied.

EXIT STATUS

Upon success, **login** returns **0**. Upon failure, **login** returns a nonzero value.

SUMMARY OF TRUSTED SOLARIS CHANGES

Restrictions on labels and UIDs apply. The **DESCRIPTION** section explains these restrictions. Trusted Solaris adds two options: –**T** and–**U**. (See **OPTIONS**.)

FILES

$HOME/.cshrc	Initial commands for each csh
$HOME/.hushlogin	Suppresses login messages
$HOME/.login	User’s login commands for **csh**
$HOME/.profile	User’s login commands for **sh** and **ksh**
$HOME/.rhosts	Private list of trusted hostname / username combinations
/etc/.login	Systemwide csh login commands
/etc/logindevperm	Login-based device permissions
/etc/motd	Message of the day
/etc/nologin	Message for users attempting to log in during machine shutdown
/etc/passwd	Password file
/etc/profile	Systemwide **sh** and **ksh** login commands
/etc/shadow	List of users’ encrypted passwords
/usr/bin/sh	User’s default command interpreter
/var/adm/lastlog	Time of last login
/var/adm/loginlog	Record of failed login attempts
/var/adm/utmp	Accounting
/var/adm/wtmp	Accounting
/var/mail/*your-name*	Mailbox for user *your-name*
/etc/default/login	Default value can be set for these flags in **/etc/default/login**. For example: **TIMEZONE=EST5EDT**.

TIMEZONE	Sets the **TZ** environment variable of the shell. [See **environ**(5).]
HZ	Sets the **HZ** environment variable of the shell.
ULIMIT	Sets the file-size limit for the login. Units are disk blocks. Default is zero (no limit).
CONSOLE	If this flag is set, administrative users can log in only on that device. This setting will not prevent execution of remote commands with **rsh**(1).

Comment out this line to allow login by administrative users.

PASSREQ	Determines if login requires a password.
ALTSHELL	Determines if login should set the **SHELL** environment variable.
PATH	Sets the initial shell **PATH** variable.
SUPATH	Sets the initial shell **PATH** variable for administrative users.
TIMEOUT	Sets the number of seconds (between **0** and **900**) to wait before abandoning a login session.
UMASK	Sets the initial shell file-creation mode mask. See **umask**(1).
SYSLOG	Determines whether to use the **syslog**(3) **LOG_AUTH** facility to log all root logins at level **LOG_NOTICE** and multiple failed login attempts at **LOG_CRIT**.
SLEEPTIME	Sets the number of seconds to wait before login failure is printed to the screen and another login attempt is allowed. Default is **4** seconds; minimum is **0** seconds; maximum is **5** seconds.

SEE ALSO

csh(1), **ksh**(1), **mail**(1), **mailx**(1), **newgrp**(1), **passwd**(1), **rlogin**(1), **rsh**(1), **sh**(1), **shell_builtins**(1), **telnet**(1), **umask**(1), **admintool**(1M), **in.rlogind**(1M), **in.telnetd**(1M), **logins**(1M), **quota**(1M), **su**(1M), **syslogd**(1M), **useradd**(1M), **userdel**(1M), **rcmd**(3N), **syslog**(3), **ttyname**(3C), **hosts.equiv**(4), **logindevperm**(4), **loginlog**(4), **nologin**(4), **nsswitch.conf**(4), **passwd**(4), **profile**(4), **shadow**(4), **environ**(5), **termio**(7I)

DIAGNOSTICS

Login incorrect
The user name or the password cannot be matched, the user account is locked, or the name of a role was given instead of the name of a user.
#else /* !TSOL */ The user name or the password cannot be matched.

New password needed; use the desktop
The user's password has expired; to log in and change the password, you must use the desktop.

Not on system console
Administrative user login denied. Check the **CONSOLE** setting in **/etc/default/login**.

No directory! Logging in with home=/
The user's home directory named in the **passwd**(4) database cannot be found or has the wrong permissions. Contact your system administrator.

No shell
Cannot execute the shell named in the **passwd**(4) database. Contact your system administrator.

NO LOGINS: System going down in *N* minutes
The machine is in the process of being shut down and logins have been disabled.

WARNINGS

Users with a UID greater than 76695844 are not subject to password aging, and the system does not record their last login time.

If you use the **CONSOLE** setting to disable administrative user logins, you should arrange that remote command execution by administrative users is also disabled. See **rsh**(1), **rcmd**(3N), and **hosts.equiv**(4) for further details.

NAME

lp, cancel – send / cancel requests to an LP print service

SYNOPSIS

lp [−**m**] [−**p**] [−**s**] [−**w**] [−**d** *dest*] [−**f** *form-name* [−**d** **any**]]
[−**H** *special-handling*] [−**n** *number*] [−**o** *option*] [−**P** *page-list*] [−**q** *priority-level*]
[−**S** *character-set* [−**d** **any**]] [−**S** *print-wheel* [−**d** **any**]] [−**t** *title*]
[−**T** *content-type* [−**r**]] [−**y** *mode-list*] [*file* . . .]

lp −**i** *request-ID* . . . [−**c**] [−**m**] [−**p**] [−**s**] [−**w**] [−**d** *dest*]
[−**f** *form-name* [−**d** **any**]] [−**H** *special-handling*] [−**n** *number*] [−**o** *option*]
[−**P** *page-list*] [−**q** *priority-level*] [−**S** *character-set* [−**d** **any**]]
[−**S** *print-wheel* [−**d** **any**]] [−**t** *title*] [−**T** *content-type* [−**r**]] [−**y** *mode-list*]

cancel [*request-ID* . . .] [*printer* . . .]
cancel −**u** *login-ID-list* [*printer* . . .]

AVAILABILITY

SUNWlpu

DESCRIPTION

The first form of the **lp** command arranges for the named *file(s)* and associated information (collectively called a *request*) to be printed. If no file names are specified on the command line, the standard input is assumed. The standard input may be specified along with a named *file*(s) on the command line by listing the file name(s) and specifying `−´ (dash) for the standard input. The *files* will be printed in the order in which they appear on the shell command line.

The LP print service associates a unique *request-ID* (with the −**i** option) with each request and displays it on the standard output. This *request-ID* can be used later with the −**i** option when canceling or changing a request, or when determining its status. (See the section on **cancel** for details about canceling a request, and for information about checking the status of a print request.)

The second form of **lp** is used to change the options for a request. The print request identified by the *request-ID* is changed according to the printing options specified with this shell command. The printing options available are the same as those with the first form of the **lp** shell command. If the request has finished printing, the change is rejected. If the request is already printing, it will be stopped and restarted from the beginning (unless the −**P** option has been given).

The **cancel** command allows users to cancel print requests previously sent with the **lp** command. The first form of **cancel** permits cancellation of requests based on their *request-ID*. The second form of cancel permits cancellation of requests based on the *login-ID* of their owner.

Sending a Print Request

The first form of the **lp** command is used to send a print request to a particular printer or group of printers.

OPTIONS

Options to **lp** always precede any file names, but may be specified in any order. The following options are available for **lp**:

–**d** *dest* Choose *dest* as the printer or class of printers that is to do the printing. If *dest* is a printer, then the request will be printed only on that specific printer. If *dest* is a class of printers, then the request will be printed on the first available printer that is a member of the class. If *dest* is **any**, then the request will be printed on any printer which can handle it. Under certain conditions, (unavailability of printers, file space limitations, and so on) requests for specific destinations may not be accepted (see **lpstat**(1TSOL)). By default, *dest* is taken from the environment variable **LPDEST** (if it is set). Otherwise, a default destination (if one exists) for the computer system is used. Destination names vary between systems (see **lpstat**(1TSOL)).

–**f** *form-name* [–**d any**]
Print the request on the form *form-name*. The LP print service ensures that the form is mounted on the printer. If *form-name* is requested with a printer destination that cannot support the form, the request is rejected. If *form-name* has not been defined for the system, or if the user is not allowed to use the form, the request is rejected (see **lpforms**(1MTSOL)). When the –**d any** option is given, the request is printed on any printer that has the requested form mounted and can handle all other needs of the print request.

–**H** *special-handling*
Print the request according to the value of *special-handling*. Acceptable values for *special-handling* are defined below:

hold Do not print the request until notified. If printing has already begun, stop it. Other print requests will go ahead of a held request until it is resumed.

resume Resume a held request. If the request had begun to print when held, it will be the next request printed, unless it is superseded by an **immediate** request.

immediate (Available only to LP administrators.) Print the request next. If more than one request is assigned the most recent request is printed next. If a request is currently printing on the desired printer, a hold request must be issued to allow the immediate request to print.

–**m** Send mail (see **mail**(1)) after the files have been printed. By default, no mail is sent upon normal completion of the print request.

–**n** *number* Print *number* copies (default is **1**) of the output.

−o *option* Specify printer-dependent *options*. Several such *options* may be collected by specifying the **−o** keyletter more than once (**−o** $option_1$ **−o** $option_2$ ... **−o** $option_n$), or by specifying the **−o** keyletter followed by a list of options enclosed in double quotes (that is, **−o** "$option_1$ $option_2$ ... $option_n$"). The standard interface recognizes the following options:

nobanner Do not print banner and trailer pages with this request. Use of this option requires the **print without banners** authorization. (The administrator can disallow this option at any time.)

nofilebreak
Do not insert a form feed between the files given, if submitting a job to print more than one file.

nolabels Do not print page header and footer labels with this request. Use of this option requires the **print without labels** authorization.

length=*scaled-decimal-number*
Print this request with pages *scaled-decimal-number* lines long. A *scaled-decimal-number* is an optionally scaled decimal number that gives a size in lines, columns, inches, or centimeters, as appropriate. The scale is indicated by appending the letter “i” for inches, or the letter “c” for centimeters. For length or width settings, an unscaled number indicates lines or columns; for line pitch or character pitch settings, an unscaled number indicates lines per inch or characters per inch (the same as a number scaled with “i”). For example, **length=66** indicates a page length of **66** lines, **length=11i** indicates a page length of **11** inches, and **length=27.94c** indicates a page length of **27.94** centimeters.

This option may not be used with the –**f** option.

width=*scaled-decimal-number*
Print this request with page-width set to *scaled-decimal-number* columns wide. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**, above.) This option may not be used with the –**f** option.

lpi=*scaled-decimal-number*
Print this request with the line pitch set to *scaled-decimal-number* lines per inch. This option may not be used with the –**f** option.

cpi=*scaled-decimal-number*
Print this request with the character pitch set to *scaled-decimal-number* characters per inch. Character pitch can also be set to **pica** (representing **10** characters per inch) or **elite** (representing **12** characters per inch), or it can be

compressed (representing as many characters as a printer can handle). There is no standard number of characters per inch for all printers; see the Terminfo database (see **terminfo**(4)) for the default character pitch for your printer.

This option may not be used with the −**f** option.

stty=`*stty-option-list*´
A list of options valid for the **stty** command; enclose the list with single quotes if it contains blanks.

−**P** *page-list*　Print the pages specified in *page-list*. This option can be used only if there is a filter available to handle it; otherwise, the print request will be rejected.

The *page-list* may consist of range(s) of numbers, single page numbers, or a combination of both. The pages will be printed in ascending order.

−**p**　Enable notification on completion of the print request. Delivery of the notification is dependent on additional software.

−**q** *priority-level*　Assign this request *priority-level* in the printing queue. The values of *priority-level* range from **0**, the highest priority, to **39**, the lowest priority. If a priority is not specified, the default for the print service is used, as assigned by the system administrator. A priority limit may be assigned to individual users by the system administrator.

−**s**　Suppress messages from **lp** such as those that begin with "**request id is** . . ."

−**S** *character-set* [−**d any**]
−**S** *print-wheel* [−**d any**]
Print this request using the specified *character-set* or *print-wheel*. If a form was requested and it requires a character set or print wheel other than the one specified with the −**S** option, the request is rejected.

For printers that take print wheels: if the print wheel specified is not one listed by the administrator as acceptable for the printer specified in this request, the request is rejected unless the print wheel is already mounted on the printer.

For printers that use selectable or programmable character sets: if the *character-set* specified is not one defined in the Terminfo database for the printer (see **terminfo**(4)), or is not an alias defined by the administrator, the request is rejected.

When the –**d any** option is used, the request is printed on any printer that has the print wheel mounted or any printer that can select the character set, and that can handle the needs of the request.

–**t** *title* Print *title* on the banner page of the output. If *title* is not supplied the name of the file is printed on the banner page. Enclose *title* in quotes if it contains blanks.

–**T** *content-type* [–**r**]
Print the request on a printer that can support the specified *content-type*. If no printer accepts this type directly, a filter will be used to convert the content into an acceptable type. If the –**r** option is specified, a filter will not be used. If –**r** is specified, and no printer accepts the *content-type* directly, the request is rejected. If the *content-type* is not acceptable to any printer, either directly or with a filter, the request is rejected. Submitting a request with the "postscript" type requires the **print a Postscript file** authorization, whether or not –**T** is used.

–**w** Write a message on the user's terminal after the *files* have been printed. If the user is not logged in, then mail will be sent instead.

–**y** *mode-list* Print this request according to the printing modes listed in *mode-list*. The allowed values for *mode-list* are locally defined. This option may be used only if there is a filter available to handle it; otherwise, the print request will be rejected.

Canceling a Print Request

The **cancel** command cancels requests for print jobs made with the **lp** command. The first form allows a user to specify one or more *request-ID* of print jobs to be canceled. Alternatively, the user can specify one or more *printer*, on which only the currently printing job will be canceled.

The second form of **cancel** permits a user to cancel all of his or her own jobs on all printers. In this form the *printer* option can be used to restrict the printer(s) on which the user's job(s) will be canceled. Note: In this form, when the *printer* option is used, all jobs queued for that printer will be canceled. A printer class is not a valid argument.

Normally a user can cancel only requests associated with his or her own login ID. If the **cancel** command is run with the **file_dac_write** privilege, it can cancel jobs submitted by any user. The login-ID-list must be enclosed in quotes if it contains blanks.

Normally a user can cancel only requests that are at the user's current sensitivity label. If the **cancel** command is run with the file_mac_write privilege, the command can cancel a job whose label dominates the user's label. If **cancel** is run with the **file_mac_read** privilege, the command can cancel a job whose label is dominated by the user's label. If **cancel** is run with both **file_mac_read** and **file_mac_write** privileges, the command can cancel jobs at any label.

For printers that take mountable print wheels or font cartridges, if you do not specify a particular print wheel or font with the –**S** option, the one mounted at the time your request is printed will be used. Use the **lpstat** –**p** *printer* –**l** command to see which print wheels are available on a particular printer, or the **lpstat** –**S** –**l** command to find out what print wheels are available and on which printers. For printers that have selectable character sets, you will get the standard character set if you don't use the –**S** option.

OPERANDS

The following operands are supported by **lp**:

file A path name of a file to be output. If no *file* operands are specified, or if a *file* operand is –, the standard input will be used.

The following operands are supported by **cancel**:

ID A request *ID*, as returned by **lp**. Specifying a request *ID* cancels the associated request even if it is currently printing.

printer A printer name (for a complete list of printer names, use **lpstat**). Specifying a printer cancels the request that is currently printing on that printer.

ENVIRONMENT

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

LPDEST Determine the output device or destination. If the **LPDEST** environment variable is not set, the **PRINTER** environment variable will be used. The –**d** *dest* option takes precedence over **LPDEST**. Results are undefined when –**d** is not specified and **LPDEST** contains a value that is not a valid device or destination name.

PRINTER Determine the output device or destination. If the **LPDEST** and **PRINTER** environment variables are not set, an unspecified output device is used. The –**d** *dest* option and the **LPDEST** environment variable takes precedence over **PRINTER**. Results are undefined when –**d** is not specified, **LPDEST** is unset, and **PRINTER** contains a value that is not a valid device or destination name.

EXIT STATUS

The following exit values are returned by **lp**:

0	All input files were processed successfully.
>0	No output device was available, or an error occurred.

The following exit values are returned by **cancel**:

0	Successful completion.
>0	An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**c** option is accepted but is ignored; a copy of the file is always made before printing. The –**o nobanners** option requires the **print without banners** authorization. The –**o nolabels** option is added. Submitting a request with the "postscript" type requires the **print a PostScript file** authorization.

To cancel other users' requests, **cancel** must be run with the **file_dac_write privilege. To cancel requests at** the command must be run with either the **file_dac_write** or the **file_dac_read** privilege, or both.

FILES

/var/spool/lp/∗ LP print queue

SEE ALSO

enable(1TSOL), **lpstat**(1TSOL), **mail**(1), **postprint**(1), **pr**(1), **accept**(1MTSOL), **lpadmin**(1MTSOL), **lpfilter**(1MTSOL), **lpforms**(1MTSOL), **lpsched**(1MTSOL), **lpsystem**(1MTSOL), **lpusers**(1MTSOL), **terminfo**(4), **environ**(5)

NOTES

Printers for which requests are not being accepted will not be considered when the **lp** command is run and the destination is **any**. (Use the **lpstat −a** command to see which printers are accepting requests.) On the other hand, if a request is destined for a class of printers and the class itself is accepting requests, then *all* printers in the class will be considered, regardless of their acceptance status.

NAME

lpc – line printer control program

SYNOPSIS

/usr/ucb/lpc [*command* [*parameter*. . .]]

AVAILABILITY

SUNWscpu

DESCRIPTION

lpc controls the operation of the printer, or of multiple printers. **lpc** commands can be used to start or stop a printer, disable or enable a printer's spooling queue, rearrange the order of jobs in a queue, or display the status of each printer—along with its spooling queue and printer daemon.

With no arguments, **lpc** runs interactively, prompting with '**lpc>**´. If arguments are supplied, **lpc** interprets the first as a *command* to execute; each subsequent argument is taken as a *parameter* for that command. The standard input can be redirected so that **lpc** reads commands from a file.

USAGE

Commands

Commands may be abbreviated to an unambiguous substring. Specify the *printer* parameter by the name of the printer (for example, as **lw**), not as you would specify it to **lpr**(1BTSOL) or **lpq**(1BTSOL) (not as –**Plw**).

? [*command*] . . .
help [*command*] . . .

Display a short description of each command specified in the argument list, or, if no arguments are given, a list of the recognized commands.

abort [**all** | [*printer* . . .]]

Terminate an active spooling daemon on the local host immediately and then disable printing (preventing new daemons from being started by **lpr**(1BTSOL)) for the specified printers. Use of the **abort** command requires the **administer printing** authorization.

clean [**all** | [*printer* . . .]]

Remove all files created in the spool directory by the daemon from the specified printer queue(s) on the local machine. Use of the **clean** command requires the **administer printing** authorization.

disable [**all** | [*printer* . . .]]

Turn the specified printer queues off. This prevents new printer jobs from being entered into the queue by **lpr**(1B). Use of the **disable** command requires the **administer printing** authorization.

down [**all** | [*printer* . . .]] [*message*]

Turn the specified printer queue off, disable printing and put *message* in the printer status file. The message does not need to be quoted, and the remaining arguments are treated like **echo**(1). This is normally used to take a printer down and let others know the reason (**lpq**(1BTSOL) indicates that the printer is down, as does the **status** command).

enable [**all** | [*printer* . . .]]
Enable spooling on the local queue for the listed printers, so that **lpr**(1B) can put new jobs in the spool queue. Use of the **enable** command requires the **administer printing** authorization.

exit
quit Exit from **lpc**.

restart [**all** | [*printer* . . .]]
Attempt to start a new printer daemon. This is useful when some abnormal condition causes the daemon to die unexpectedly leaving jobs in the queue. This command can be run by any user.

start [**all** | [*printer* . . .]]
Enable printing and start a spooling daemon for the listed printers. Use of the **start** command requires the **administer printing** authorization.

status [**all** | [*printer* . . .]]
Display the status of daemons and queues on the local machine. This command can be run by any user.

stop [**all** | [*printer* . . .]]
Stop a spooling daemon after the current job completes and disable printing. Use of the **stop** command requires the **administer printing** authorization.

topq *printer* [*job#* . . .] [*user* . . .]
Move the print job(s) specified by *job#* or those job(s) belonging to *user* to the top (head) of the printer queue. Use of the **topq** command requires the **administer printing** authorization.

up [**all** | [*printer* . . .]] Enable everything and start a new printer daemon. Undoes the effects of **down**.

SUMMARY OF TRUSTED SOLARIS CHANGES

Functions of this command that are restricted to the super-user in Solaris require the **administer printing** authorization in Trusted Solaris.

FILES

/var/spool/lp/∗	spooling directories
/var/spool/lp/system/pstatus	printer status information

SEE ALSO

echo(1), **lpq**(1B), **lpr**(1B), **lprm**(1B), **lpstat**(1), **lpsched**(1M)

DIAGNOSTICS

?Ambiguous command
The abbreviation you typed matches more than one command.

?Invalid command
You typed a command or abbreviation that was not recognized.

?Privileged command
You used a command that requires the **administer printing** authorization.

lpc: *printer* : **unknown printer to the print service**

The **printer** was not found in the LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **`lpstat −p´** (see **lpstat**(1TSOL)) or the **status** command (see **Commands** above) to discover the reason.

lpc: error on opening queue to spooler

The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check to see if the printer spooler daemon **/usr/lib/lp/lpsched** is running.

lpc: Can't send message to LP print service

lpc: Can't receive message from LP print service

These indicate that the LP print service has been stopped. Get help from the system administrator.

lpc: Received unexpected message from LP print service

It is likely there is an error in this software. Get help from system administrator.

NAME

lpq – display the queue of printer jobs

SYNOPSIS

/usr/ucb/lpq [–**P** *printer*] [–**l**] [–**M**] [**+** [*interval*]] [*job#* . . .] [*username* . . .]

AVAILABILITY

SUNWscpu

DESCRIPTION

lpq displays the contents of a printer queue. It reports the status of jobs specified by *job#*, or all jobs owned by the user specified by *username*. **lpq** reports on all jobs in the default printer queue when invoked with no arguments.

For each print job in the queue, **lpq** reports the user's name, current position, the names of input files comprising the job, the job number (by which it is referred to when using **lprm**(1BTSOL)C) and the total size in bytes. Normally, only as much information as will fit on one line is displayed. Jobs are normally queued on a first-in-first-out basis. Filenames comprising a job may be unavailable, such as when **lpr** is used at the end of a pipeline; in such cases the filename field indicates the standard input.

Normally, **lpq** displays only the user's own print jobs. If **lpq** is run with the **file_dac_read** privilege, it displays other users' print jobs as well.

If **lpq** warns that there is no daemon present (that is, due to some malfunction), the **lpc**(1BTSOL) command can be used to restart a printer daemon.

OPTIONS

–**P** *printer*
Display information about the queue for the specified *printer*. In the absence of the –**P** option, the queue to the printer specified by the **PRINTER** variable in the environment is used. If the **PRINTER** variable is not set, and the **LPDEST** environment variable is not set, the queue for the default printer is used.

–**l**
Display queue information in long format; includes the name of the host from which the job originated.

–**M**
Display multilabel queue information. Without this option, only jobs at the user's sensitivity label are displayed. If the –**M** option is used, all jobs at sensitivity labels dominated by the user's sensitivity label are displayed. If the –**M** option is used and **lpq** is run with the **file_mac_read** privilege, jobs at all sensitivity labels are displayed. The label names for jobs at sensitivity labels not dominated by the user's sensitivity label may not be displayed properly if **lpq** is run without the **sys_trans_label** privilege.

+[*interval*]
Display the spool queue periodically until it empties. This option clears the terminal screen before reporting on the queue. If an *interval* is supplied, **lpq** sleeps that number of seconds in between reports.

SUMMARY OF TRUSTED SOLARIS CHANGES

The −**M** option is added. To display other users' print jobs requires the **file_dac_read** privilege. To display print jobs at sensitivity labels not dominated by the user's sensitivity label requires the **file_mac_read** privilege.

FILES

/var/spool/lp	spooling directory
/var/spool/lp/tmp/*system_name***/**∗**-0**	request files specifying jobs

SEE ALSO

lp(1TSOL)C , **lpc**(1BTSOL)C , **lpr**(1BTSOL)C , **lprm**(1BTSOL)C , **lpstat**(1TSOL)C , **lpsched**(1MTSOL)

DIAGNOSTICS

printer **is printing**

The **lpq** program queries the spooler **LPSCHED** about the status of the printer. If the printer is disabled, the administrator can restart the spooler using **lpc**(1BTSOL)C .

printer **waiting for auto-retry (offline ?)**

The daemon could not open the printer device. The printer may be turned off-line. This message can also occur if a printer is out of paper, the paper is jammed, and so on. Another possible cause is that a process, such as an output filter, has exclusive use of the device. The only recourse in this case is to kill the offending process and restart the printer with **lpc**.

waiting for *host* **to come up**

A daemon is trying to connect to the remote machine named *host*, in order to send the files in the local queue. If the remote machine is up, **lpd** on the remote machine is probably dead or hung and should be restarted using **lpc**.

sending to *host*

The files are being transferred to the remote *host*, or else the local daemon has hung while trying to transfer the files.

printer disabled reason:

The printer has been marked as being unavailable with **lpc**.

lpq: The LP print service isn't running or can't be reached.

The **lpsched** process overseeing the spooling queue does not exist. This normally occurs only when the daemon has unexpectedly died. You can restart the printer daemon with **lpc**.

lpr: *printer* **: unknown printer**

The **printer** was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use '**lpstat −p**' (see **lpstat**(1TSOL)C) or '**lpc status**' (see **lpc**(1BTSOL)C) to discover the reason.

lpr: error on opening queue to spooler

The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lpsched** is running.

lpr: Can't send message to LP print service

lpr: Can't receive message from LP print service

These indicate that the LP print service has been stopped. Get help from the system administrator.

lpr: Received unexpected message from LP print service

It is likely there is an error in this software. Get help from system administrator.

NOTES

Output formatting is sensitive to the line length of the terminal; this can result in widely-spaced columns.

NAME

lpr – send a job to the printer

SYNOPSIS

/usr/ucb/lpr [**–P** *printer*] [**–#** *copies*] [**–C** *class*] [**–J** *job*] [**–T** *title*]
[**–i** [*indent*]] [**–w** *cols*] [**–m**] [**–h**] [**–s**] [*–filter_option*] [*filename* ...]

AVAILABILITY

SUNWscpu

DESCRIPTION

lpr forwards printer jobs to a spooling area for subsequent printing as facilities become available. Each printer job consists of copies of each file you specify. The spool area is managed by the line printer spooler, **lpsched**. **lpr** reads from the standard input if no files are specified.

OPTIONS

–P *printer* Send output to the named *printer*. In the absence of the **–P** option, the queue to the printer specified by the **PRINTER** variable in the environment is used. If the **PRINTER** variable is not set, and the **LPDEST** environment variable is not set, the queue for the default printer is used.

–# *copies* Produce the number of *copies* indicated for each named file. For example:

```
lpr –#3 index.c lookup.c
```

produces three copies of **index.c**, followed by three copies of **lookup.c**. On the other hand,

```
cat index.c lookup.c | lpr –#3
```

generates three copies of the concatenation of the files.

–C *class* Print *class* as the job classification on the burst page. For example,

```
lpr –C Operations new.index.c
```

replaces the system name (the name returned by *hostname*) with **Operations** on the burst page, and prints the file **new.index.c**.

–J *job* Print *job* as the job name on the burst page. Normally, **lpr** uses the first file's name.

–T *title* Use *title* instead of the file name for the title used by **pr**(1).

–i[*indent*] Indent output *indent* SPACE characters. Eight SPACE characters is the default.

–w *cols* Use *cols* as the page width for **pr**.

–m Send mail upon completion.

–h Suppress printing the burst page. Use of this option requires the **print without banners** authorization.

filter_option The following single letter options notify the line printer spooler that the files are not standard text files. The spooling daemon will use the appropriate filters to print the data accordingly.

–p Use **pr** to format the files (**lpr –p** is very much like **pr | lpr**).
–l Print control characters and suppress page breaks.
–t The files contain **troff**(1) (cat phototypesetter) binary data.
–n The files contain data from **ditroff** (device independent troff).
–d The files contain data from **tex** (DVI format from Stanford).
–g The files contain standard plot data as produced by the **plot**(1B) routines.
–v The files contain a raster image. The printer must support an appropriate imaging model such as PostScript® in order to print the image.
–c The files contain data produced by *cifplot*.
–f Interpret the first character of each line as a standard FORTRAN carriage control character.

If no *filter_option* is given (and the printer can interpret PostScript), the string `**%!**´ as the first two characters of a file indicates that it contains PostScript commands. To print a file containing PostScript commands requires the **print a PostScript file** authorization.

These filter options offer a standard user interface, and all options may not be available for, nor applicable to, all printers.

SUMMARY OF TRUSTED SOLARIS CHANGES

Use of the **–h** option requires the **print without banners** authorization. Printing a file that contains PostScript commands requires the **print a PostScript file** authorization. The **–s** option is accepted but ignored; a copy of the file is always made before printing.

FILES

/etc/passwd	personal identification
/usr/lib/lp/lpsched	System V line printer spooler
/var/spool/lp/tmp/∗	directories used for spooling
/var/spool/lp/tmp/*system***/**∗**-0**	spooler control files
/var/spool/lp/tmp/*system***/**∗**-***N*	(*N* is an integer and > 0) data files specified in `∗**-0**´ files

SEE ALSO

lp(1TSOL), **lpc**(1BTSOL), **lpq**(1BTSOL), **lprm**(1BTSOL), **lpstat**(1TSOL), **plot**(1B), **pr**(1), **troff**(1), **lpsched**(1MTSOL)

DIAGNOSTICS

lpr: *printer* **: unknown printer**

The **printer** was not found in the LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use `**lpstat –p**´ (see **lpstat**(1TSOL)) or `**lpc status**´ (see **lps**(1BTSOL)) to discover the reason.

lpr: error on opening queue to spooler

The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lpsched** is running.

lpr: *printer* **: printer queue is disabled**

This means the queue was turned off with

```
/usr/etc/lpc disable printer
```

to prevent **lpr** from putting files in the queue. This is normally done by the system manager when a printer is going to be down for a long time. The printer can be turned back on by an administrator with **lpc**.

lpr: Can't send message to the LP print service

lpr: Can't receive message from the LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lpr: Received unexpected message from LP print service
It is likely there is an error in this software. Get help from system administrator.

lpr: There is no filter to convert the file content
Use the `**lpstat –p –l**´ command to find a printer that can handle the file type directly, or consult with your system administrator.

lpr: cannot access the file
Make sure file names are valid.

NOTES

lp is the preferred interface.

Command-line options cannot be combined into a single argument as with some other commands. The command:

```
lpr –fs
```

is not equivalent to

```
lpr –f –s
```

Placing the **–s** flag first, or writing each option as a separate argument, makes a link as expected.

lpr –p is not precisely equivalent to **pr | lpr**. **lpr –p** puts the current date at the top of each page, rather than the date last modified.

Fonts for **troff**(1) and T_EX® reside on the printer host. It is currently not possible to use local font libraries.

lpr objects to printing binary files.

The **–s** option, intended to use symbolic links in SunOS, does not use symbolic links in the compatibility package. Instead, the complete path names are used. Also, the copying is avoided only for print jobs that are run from the printer host itself. Jobs added to the queue from a remote host are always copied into the spool area. That is, if the printer does not reside on the host that **lpr** is run from, the spooling system makes a copy the file to print, and places it in the spool area of the printer host, regardless of **–s**.

NAME

lprm – remove jobs from the printer queue

SYNOPSIS

/usr/ucb/lprm [–**P***printer*] [–] [*job #* . . .] [*username* . . .]

AVAILABILITY

SUNWscpu

DESCRIPTION

lprm removes a job or jobs from a printer's spooling queue. Since the spool directory is protected from users, using **lprm** is normally the only method by which a user can remove a job.

Without any arguments, **lprm** deletes the job that is currently active, provided that the user who invoked **lprm** owns that job.

When a *username* is specified, **lprm** removes all jobs belonging to that user, if **lprm** is running with the **file_dac_write** privilege.

You can remove a specific job by supplying its job number as an argument, which you can obtain using **lpq**(1BTSOL). For example:

```
example%  lpq  –Phost
host is ready and printing

Rank     Owner    Job          Files            Total Size
active   wendy    385     standard input        35501 bytes

example%  lprm –Phost 385
```

To remove a job belonging to another user, **lpq** must be run with the **file_dac_write** privilege.

lpq can normally cancel only requests that are at its sensitivity label. If it is run with privilege, **lpq** can do more: with the **file_mac_write** privilege, it can cancel jobs whose labels dominate the user's label; with the **file_mac_read** privilege, it can cancel jobs whose labels are dominated by the user's label; and with both the **file_mac_read** and **file_mac_write** privileges, it can cancel jobs at any label.

lprm reports the names of any files it removes, and is silent if there are no applicable jobs to remove.

lprm Sends the request to cancel a job to the print spooler, **LPSCHED.**

OPTIONS

–**P***printer* Specify the queue associated with a specific printer. Otherwise the value of the **PRINTER** variable in the environment is used. If the **PRINTER** variable is not set, and the **LPDEST** environment variable is not set, the queue for the default printer is used.

– Remove all jobs owned by you. If invoked by the user with the **administer printing** authorization, all jobs in the spool are removed. Job ownership is determined by the user's login name and host name on the machine where the **lpr** command was executed.

SUMMARY OF TRUSTED SOLARIS CHANGES

To cancel other users' requests, **lpq** must be run with the **file_dac_write** privilege. To cancel requests at other sensitivity labels, it must be run with either the **file_dac_write** or the **file_dac_read** privilege, or both.

FILES

/var/spool/lp/*	spooling directories

SEE ALSO

lp(1TSOL)C , **lpc**(1BTSOL)C , **lpq**(1BTSOL)C , **lpr**(1BTSOL)C , **lpstat**(1TSOL)C , **lpsched**(1MTSOL)

DIAGNOSTICS

lprm: *printer* **: unknown printer**

The **printer** was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **`lpstat –p´** (see **lpstat**(1TSOL)C ) or **`lpc status´** (see **lpc**(1BTSOL)C) to discover the reason.

lprm: error on opening queue to spooler

The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lpsched** is running.

lprm: Can't send message to the LP print service

lprm: Can't receive message from the LP print service

These indicate that the LP print service has been stopped. Get help from the system administrator.

lprm: Received unexpected message from the LP print service

It is likely there is an error in this software. Get help from system administrator.

lprm: Can't cancel request

You are not allowed to remove another's request.

NOTES

An active job may be incorrectly identified for removal by an **lprm** command issued with no arguments. During the interval between an **lpq** command and the execution of **lprm**, the next job in queue may have become active; that job may be removed unintentionally if it is owned by you. To avoid this, supply **lprm** with the job number to remove when a critical job that you own is next in line.

lp is the preferred interface.

NAME

lpstat – print information about the status of the LP print service

SYNOPSIS

lpstat [–**d**] [–**r**] [–**R**] [–**s**] [–**t**] [–**a** [*list*]] [–**c** [*list*]] [–**f** [*list*] [–**l**]]
[–**o** [*list*]] [–**p** [*list*] [–**D**] [–**l**]] [–**P**] [–**S** [*list*] [–**l**]] [–**u** [*login-ID-list*]]
[–**v** [*list*]] [–**M**]

AVAILABILITY

SUNWlpu

DESCRIPTION

The **lpstat** command prints information about the current status of the LP print service.

If no options are given, then **lpstat** prints the status of all the user's print requests made by **lp** (see **lp**(1TSOL)). Any arguments that are not *options* are assumed to be *request-IDs* as returned by **lp**. The **lpstat** command prints the status of such requests. The *options* may appear in any order and may be repeated and intermixed with other arguments. Some of the keyletters below may be followed by an optional *list* that can be in one of two forms: a list of items separated from one another by a comma, or a list of items separated from one another by spaces enclosed in quotes. For example:

```
example% lpstat –u "user1 user2 user3"
```

Specifying **all** after any keyletter that takes *list* as an argument causes all information relevant to the keyletter to be printed. For example, the command:

```
example% lpstat –o all
```

prints the status of all output requests.

The omission of a *list* following such key letters causes all information relevant to the key letter to be printed. For example, the command:

```
example% lpstat –o
```

prints the status of all output requests.

OPTIONS

The following options are supported:

–**a** [*list*] Reports whether print destinations are accepting requests. *list* is a list of intermixed printer names and class names.

–**c** [*list*] Print name of all classes and their members. *list* is a list of class names.

–**d** Print the system default destination for output requests.

–**f** [*list*] [–**l**] Print a verification that the forms in *list* are recognized by the LP print service. *list* is a list of forms; the default is **all**. The –**l** option will list the form descriptions.

–**M** Include multilabel queue information in the output for the –**o** option. If the –**M** option is not used, only jobs at the user's current sensitivity label are displayed. If the –**M** option is used, all jobs at sensitivity labels dominated by the the user's sensitivity label are displayed. If the –**M** option is used and **lpstat** is run with the **file_mac_read** privilege, jobs at all sensitivity labels are displayed. The label names for jobs at sensitivity labels not dominated by the user's sensitivity label may not be displayed properly if **lpq** is run without the **sys_trans_label** privilege.

–o [*list*] Print the status of output requests: *list* is a list of intermixed printer names, class names, and *request-IDs*. The keyletter **–o** may be omitted. Normally **lpstat** displays only the invoking user's output requests. If **lpstat** is run with the **file_dac_read** privilege, the command displays other users' print jobs as well.

–p [*list*] [**–D**] [**–l**]
Print the status of printers. *list* is a list of printer names. If the **–D** option is given, a brief description is printed for each printer in *list*. If the **–l** option is given, and the printer is on the local machine, a full description of each printer's configuration is given, including the form mounted, the acceptable content and printer types, a printer description, the interface used, and so on.

–P Print the paper types.

–r Print the status of the LP request scheduler.

–R Print a number showing the position of each job in the print queue.

–s Print a status summary, including the status of the LP scheduler, the system default destination, a list of class names and their members, a list of printers and their associated devices, a list of the machines sharing print services, a list of all forms currently mounted, and a list of all recognized character sets and print wheels.

–S [*list*] [**–l**] Print a verification that the character sets or the print wheels specified in *list* are recognized by the LP print service. Items in *list* can be character sets or print wheels; the default for the list is **all**. If the **–l** option is given, each line is appended by a list of printers that can handle the print wheel or character set. The list also shows whether the print wheel or character set is mounted, or specifies the built-in character set into which it maps.

–t Print all status information. This includes all the information obtained with the **–s** option, plus the acceptance and idle / busy status of all printers.

–u [*login-ID-list*]
Print the status of output requests for users. 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

–v [*list*] Print the names of printers and the path names of the devices associated with them or remote system names for network printers: *list* is a list of printer names.

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**M** option is added. To display other users' print jobs requires the **file_dac_read** privilege. To display print jobs at sensitivity labels not dominated by the user's sensitivity label requires the **file_mac_read** privilege.

FILES

/etc/lp/∗	printer configuration files
/var/spool/lp/∗	print queue

SEE ALSO

enable(1TSOL), **lp**(1TSOL)

NAME

mkdir – make directories

SYNOPSIS

mkdir [**–m** *mode*] [**–p**] [**–M**] *dir*. . .

AVAILABILITY

SUNWcsu

DESCRIPTION

The **mkdir** command creates the named directories in mode **777** (possibly altered by the file mode creation mask **umask**(1)).

Standard entries in a directory (for instance, the files "**.**", for the directory itself, and "**..**", for its parent) are made automatically. **mkdir** cannot create these entries by name. Creation of a directory requires write permission in the parent directory.

The owner-ID and group-ID of the new directories are set to the process's effective user-ID and group-ID, respectively. **mkdir** calls the **mkdir**(2) system call.

setgid and mkdir

To change the **setgid** bit on a newly created directory, you must use **chmod g+s** or **chmod g-s** after executing **mkdir**.

The **setgid** bit setting is inherited from the parent directory.

OPTIONS

The following options are supported:

–m *mode* This option allows users to specify the mode to be used for new directories. Choices for modes can be found in **chmod**(1).

–p With this option, **mkdir** creates *dir* by creating all the non-existing parent directories first. The mode given to intermediate directories will be the difference between **777** and the bits set in the file mode creation mask. The difference, however, must be at least **300** (write and execute permission for the user).

–M With this option, **mkdir** creates *dir* as a multilevel directory.

OPERANDS

The following operand is supported:

dir A path name of a directory to be created.

EXAMPLES

The following example:

```
example% mkdir -p ltr/jd/jan
```

creates the subdirectory structure **ltr/jd/jan**.

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0	All the specified directories were created successfully or the **–p** option was specified and all the specified directories now exist.
>0	An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The –**M** option creates a multilevel directory.

SEE ALSO

rm(1TSOL), **sh**(1), **umask**(1), **intro**(2TSOL), **mkdir**(2TSOL), **environ**(5)

NAME

mldpwd – display the pathname of the current working directory, including any MLD adornments and SLD names.

SYNOPSIS

mldpwd

AVAILABILITY

SUNWtsolu

DESCRIPTION

mldpwd prints the canonicalized pathname of the (current) working directory. MLD adornments and SLD names are displayed as encountered. The example below illustrates the differences between **mldpwd** and **pwd**.

```
example% cd  /usr/wendy/january/reports
example% mldpwd
/usr/wendy/january/.MLD.reports/.SLD.1
example% pwd
/usr/wendy/january/reports
example%
```

SEE ALSO

pwd(1)

NAME

mldrealpath – display the canonicalized absolute pathname, including any MLD adornments and SLD names.

SYNOPSIS

mldrealpath *pathname*

AVAILABILITY

SUNWtsolu

DESCRIPTION

mldrealpath expands all symbolic links and resolves references to '/./', '/../', extra '/' characters, and translations in *pathname*. The resulting path will have no symbolic link components, nor any '/./', '/../', nor any unadorned MLD s, nor any hidden SLD names.

DIAGNOSTICS

mldrealpath exits with one of the following values:

0 Success

1 Usage error

2 Failure; error message is the system error number from **mldrealpath**(3TSOL).

SEE ALSO

mldrealpath(3TSOL)

NAME

pattr – Get the viewable process attribute flags

SYNOPSIS

/usr/proc/bin/pattr [-x] [–p *pid* **. . .]**

AVAILABILITY

SUNWtsolu

DESCRIPTION

pattr, a proc tools command, displays the viewable process attribute flags of the current process or of a process specified by *pid*. Those flags that cannot be viewed normally can be viewed with privilege. The process attribute flags are a collection of security flags:

Trusted path flag
Privilege debugging flag
Network token Mapping Process flag
Label view flags (external view or internal view)
Label translation flags
Part of diskless boot flag
Part of cut and paste selection agent flag
Part of Trusted Printing system

When the **–x** option is not specified, the output displays pairs of *Name* (N bits):<tab>*Value* as shown in the **EXAMPLES** section.

OPTIONS

–p *pid* **...** Display flags for the specified processes. If used, this option must be the last option on the command line and must be followed by one or more *pid*’s. If this option is not specified, the **pattr** command process is used.

–x Print process attribute flags in a hex format.

EXAMPLES

When **pattr** is invoked, the display looks like this:

```
host% pattr
Trusted Path (1 bit):                 Enabled/Disabled
Privilege Debugging (1 bit):          Enabled/Disabled
Label Translation (15 bits):          Specific flag (Enabled/Disabled)
Label View (2 bits):                  Internal/External
Token Mapper (1 bit):                 Enabled/Disabled
Diskless Boot (1 bit):                Enabled/Disabled
Selection Agent (1 bit):              Enabled/Disabled
Printing System (1 bit):              Enabled/Disabled
```

When **pattr** is invoked with the **–x** option, the display looks like this:

```
host% pattr -x
8533:0x40003
```

RETURN VALUES

Upon success, **pattr** returns **0**. Upon failure, **pattr** returns –**1**.

SEE ALSO

proc(1), **getpattr**(2TSOL), **setpattr**(2TSOL)

NAME

pclear – Get process clearance

SYNOPSIS

/usr/proc/bin/pclear [–p *pid***]**
/usr/proc/bin/pclear –l [–p *pid***]**
/usr/proc/bin/pclear –L [–p *pid***]**

AVAILABILITY

SUNWtsolu

DESCRIPTION

pclear, a proc tools command, displays the process clearance, the clearance at which the process is running. If no *pid* is specified, the clearance of **pclear** is returned. The information is displayed in the form **pid: clearance**.

OPTIONS

–p *pid* Display the clearance of the process whose process ID is specified. If no process ID is specified, display the clearance of the **pclear** command.

–l Display the clearance in short form. This option is the default.

–L Display the clearance in long form.

RETURN VALUES

pclear exits with one of these values:

0 Successful completion of **pclear**

1 Unsuccessful completion of **pclear** because of usage error

2 Inability to translate clearance

3 Inability to allocate memory

EXAMPLE

```
host% pclear
4676: TS A B
```

SEE ALSO

proc(1), **getclearance**(2TSOL)

NAME

plabel – Get the CMW label of a process

SYNOPSIS

plabel [–p *pid***]**
plabel –i [–p *pid***]**
plabel –s [–p *pid***]**
plabel –l [–p *pid***]**
plabel –I [–p *pid***]**
plabel –S [–p *pid***]**
plabel –L [–p *pid***]**
plabel –i –s [–p *pid***]**
plabel –I –S [–p *pid***]**
plabel –i –S [–p *pid***]**
plabel –I –s [–p *pid***]**

AVAILABILITY

SUNWtsolu

DESCRIPTION

plabel, a proc tools command, gets the CMW label of a process. When output options are not specified, the format of the CMW label display follows standard guidelines. If the preceding output format specified conflicts with the current output format, the current format prevails in the conflicting portion. Conflicting options include **–i** and **–I**, **–s** and **–S**, and **–l** and **–L**.

OPTIONS

–p *pid* Display the CMW label of the process whose process ID is specified. When this option is not specified, the label displayed is that of the **plabel** command.

–i Get the information label associated with the process, and display that label in short form.

–I Get the information label associated with the process, and display that label in long form.

–l Get the CMW label associated with the process, and display that label in short form.

–L Get the CMW label associated with the process, and display that label in long form.

–s Get the sensitivity label associated with the process, and display that label in short form.

–S Get the sensitivity label associated with the process, and display that label in long form.

RETURN VALUES

plabel exits with one of these values:

0 Successful completion of **plabel**

1 Unsuccessful completion of **plabel** because of a usage error

2 Inability to translate label

3 Inability to allocate memory

SEE ALSO

proc(1), **getcmwplabel**(2TSOL)

NAME

ppriv – Get the effective privileges of a process

SYNOPSIS

ppriv [–p *pid***]**

AVAILABILITY

SUNWtsolu

DESCRIPTION

ppriv, a proc tools command, gets the effective privileges of the process specified by *pid*. If no *pid* has been specified, the effective privileges of **ppriv** are displayed. The privileges are displayed in this manner:

```
p1,p2,p3...
```

When all the privileges are effective, the display is simply

```
all
```

OPTIONS

–p *pid* Display the effective privilege set of the process whose process ID is specified. If no process ID is specified, the effective privilege set of the **ppriv** command is displayed

RETURN VALUES

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

SEE ALSO

proc(1), **pprivtest**(1TSOL), **getppriv**(2TSOL)

NAME

pprivtest – Test effective privilege set of the process

SYNOPSIS

pprivtest [–e] [–s] [–p *pid***]** *priv_names*

AVAILABILITY

SUNWtsolu

DESCRIPTION

pprivtest, a proc tools command, tests whether the *priv_names* privileges are a subset of the effective set of the process. *priv_names* is one of these:

- A comma-separated list of ASCII privilege names, as reported by **ppriv**
- A comma-separated list of numeric privilege IDs as found in **<sys/tsol/priv_names.h>**
- The keyword **all** to indicate all privileges

Without the –**e** (equal) option, the specified privileges are checked as a subset of the process privileges. **pprivtest** reports those privileges that are specified in *priv_names* but not found in the process. The –**e** option additionally reports privileges that the file has, but that were not specified in the **pprivtest** command.

OPTIONS

–**p** *pid* Test the privilege set of the process specified by the process ID. If no process ID is specified, test the privilege set of the **pprivtest** command.

–**e** Test whether the specified privileges are equal to the effective privileges of the process.

–**s** Use silent mode to suppress outputs. (This option is useful in shell scripts that need only the return value.)

RETURN VALUES

pprivtest exits with one of these values:

0 All of the specified privileges are in the effective set.

With the –**e** option, the specified privileges are equal to the effective set of the process.

1 At least one of the specified privileges is not in the effective set of the process.

With the –**e** option, the specified privileges are not equal to the effective set of the process.

EXAMPLE

Use this command to test if the current process' privileges are exactly equal to the specified privileges:

```
example% pprivtest -e p1,p2
```

If the process privileges did not match exactly, the output could be in this example format:

```
example% 1298:missing:p2:extra:p3
```

SEE ALSO

proc(1)C , **ppriv**(1TSOL)C , **priv_names**(4TSOL)

NAME

rm, rmdir – remove directory entries

SYNOPSIS

/usr/bin/rm [**–f**] [**–i**] *file*. . .
/usr/bin/rm **–rRM** [**–f**] [**–i**] *dirname*. . . [*file*. . .]

/usr/xpg4/bin/rm [**–fiRr**] *file* . . .

/usr/bin/rmdir [**–ps**] *dirname*. . .

AVAILABILITY

/usr/bin/rm SUNWcsu
/usr/bin/rmdir
/usr/xpg4/bin/rm SUNWxcu4

DESCRIPTION

rm

The **rm** command removes the directory entry specified by each *file* argument. If a *file* has no write permission and the standard input is a terminal, the full set of permissions (in octal) for the file are printed followed by a question mark. This is a prompt for confirmation. If the answer begins with **y** (for yes), the file is deleted, otherwise the file remains.

If *file* is a symbolic link, the link will be removed, but the file or directory to which it refers will not be deleted. Users do not need write permission to remove a symbolic link, provided they have write permissions in the directory.

If multiple *file*s are specified and removal of a *file* fails for any reason, **rm** will write a diagnostic message to standard error, do nothing more to the current *file*, and go on to any remaining *file*s.

If the standard input is not a terminal, the command will operate as if the **–f** option is in effect.

rmdir

The **rmdir** command will remove the directory entry specified by each *dirname* operand, which must refer to an empty directory.

Directories will be processed in the order specified. If a directory and a subdirectory of that directory are specified in a single invocation of **rmdir**, the subdirectory must be specified before the parent directory so that the parent directory will be empty when **rmdir** tries to remove it.

If a specified directory is a single-level directory, the directory is not removed. SLDs may be removed by first removing all files in the SLDs, then removing the multilevel directory containing the SLDs.

OPTIONS

The following options apply to **rm**:

/usr/bin/rm

–f Remove all files (whether write-protected or not) in a directory without prompting the user. In a write-protected directory, however, files are never removed (whatever their permissions are), but no messages are displayed. If the removal of a write-protected directory is attempted, this option will not suppress an error message.

/usr/xpg4/bin/rm **–f** Do not prompt for confirmation. Do not write diagnostic messages or modify the exit status in the case of non-existent operands. Any previous occurences of the **–i** option will be ignored.

/usr/xpg4/bin/rm **–i** Interactive. With this option, **rm** prompts for confirmation before removing any files. It overrides the **–f** option and remains in effect even if the standard input is not a terminal.

/usr/xpg4/bin/rm **–i** Prompt for confirmation. Any occurences of the **–f** option will be ignored.

–r Recursively remove directories and subdirectories in the argument list. The directory will be emptied of files and removed. The user is normally prompted for removal of any write-protected files which the directory contains. The write-protected files are removed without prompting, however, if the **–f** option is used, or if the standard input is not a terminal and the **–i** option is not used.

Symbolic links that are encountered with this option will not be traversed.

If the removal of a non-empty, write-protected directory is attempted, the command will always fail (even if the **–f** option is used), resulting in an error message.

–R Same as **–r** option.

/usr/bin/rm **–M** When this option is used with the recursive option (**–R**), **rm** processes all accessible SLDs as it descends multilevel directories.

The following options apply to **rmdir**:

–p Allow users to remove the directory *dirname* and its parent directories which become empty. A message is printed on the standard error about whether the whole path is removed or part of the path remains for some reason.

–s Suppress the message printed on the standard error when **–p** is in effect.

OPERANDS

The following operand is supported:

file A path name of a directory entry to be removed.

dirname A path name of an empty directory to be removed.

EXAMPLES

rm

The following command:

```
example% rm a.out core
```

removes the directory entries: **a.out** and **core**.

The following command:

```
example% rm -rf junk
```

removes the directory **junk** and all its contents, without prompting.

rmdir

If a directory **a** in the current directory is empty except it contains a directory **b** and **a/b** is empty except it contains a directory **c**,

```
example% rmdir -p a/b/c
```

will remove all three directories.

ENVIRONMENT

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

EXIT STATUS

The following exit values are returned:

0 If the −**f** option was not specified, all the named directory entries were removed; otherwise, all the existing named directory entries were removed.

>0 An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

The −**M** option for **rm** processes all accessible SLDs in multilevel directories. If a directory specified for **rmdir** is an SLD, it is not removed.

SEE ALSO

rmdir(2), **unlink**(2), **environ**(5)

DIAGNOSTICS

All messages are generally self-explanatory.

It is forbidden to remove the files "**.**" and "**..**" in order to avoid the consequences of inadvertently doing something like the following:

```
rm −r .*
```

NOTES

A −− permits the user to mark explicitly the end of any command line options, allowing **rm** to recognize file arguments that begin with a −. As an aid to BSD migration, **rm** will accept − as a synonym for −−. This migration aid may disappear in a future release. If a −− and a − both appear on the same command line, the second will be interpreted as a file.

NAME

setfattrflag – sets the file's security attribute flags

SYNOPSIS

setfattrflag [**–m**] [**–t**] [**–p** *public*] *filename ...*

AVAILABILITY

SUNWtsolu

DESCRIPTION

setfattrflag sets the security attributes flags of *filename*. For **setfattrflag** to successfully set directory flags, *filename* must be a directory. For **setfattrflag** to successfully set file-related flags, *filename* must be a file. Setting a file's public object security attribute flag requires the FILE_AUDIT privilege, and additionally the FILE_OWNER privilege when the owner of the invoking process is not owner of the file. At least one option is required, and *public* must be either 0 (zero) to clear the flag or 1 (one) to set the flag.

OPTIONS

–m To set the MLD flag on the directory. Once set, this cannot be cleared.

–p To set the public object flag.

–t If *filename* is an mld, translate to the underlying single-level directory. By default, **setfattrflag** does not translate multi-level directories to underlying single-level directories. This options is not allowed with the **–m** option.

RETURN VALUES

setfattrflag exits with one of the following values:

0 The setting of the security attribute flag was successful.

1 The setting of the security attribute flag was unsuccessful.

NAME

setfpriv – Change the privilege sets associated with a file

SYNOPSIS

/usr/bin/setfpriv { –s | –m | –d } –a *privseta* **–f** *privsetf file* . . .

AVAILABILITY

SUNWtsolu

DESCRIPTION

Change the privilege sets of a file or files. Only a process with the **file_setpriv** privilege in its inheritable set may successfully execute this command. Only the owner of a file may change the privilege sets associated with that file unless the invoking process has the **file_owner** privilege. The invoking process must have MAC write permission. DAC write permission is not required.

The –**s** option sets the privileges to the entries specified on the command line. The –**d** option deletes one or more specified privileges from the file's privilege set. The –**m** option adds one or more specified privileges to the file's privilege set. One and only one of the options –**s**, –**m**, or –**d** must be specified.

The –**a** option specifies that a set of allowed privileges is to be set. The –**f** option specifies that a set of forced privileges is to be set. *privseta* and *privsetf* are one of these:

- A comma-separated list of ASCII privilege names as reported by **getfpriv**
- A comma-separated list of numeric privilege IDs as found in **<sys/tsol/priv_names.h>**
- The keyword **all** to indicate all privileges
- The keyword **none** to indicate an empty privilege set

One or both of the options –**a** and –**f** must be specified, each followed by a privilege set. No white space may exist in a privilege-set list.

An attempt to assert a privilege in a file's forced set is denied unless that privilege is also asserted in the file's allowed set. All privileges cleared in a file's allowed set are automatically cleared from the file's forced set. It is not an error to attempt to clear a privilege from a set in which it is already cleared.

EXAMPLE

Setting privileges in the forced set requires that those privileges be set in the file's allowed set. Both the file's allowed and forced privilege sets can be set at the same time. To set all allowed privileges and a set of forced privileges on a file, use this command:

```
example% setfpriv –s –a all –f p1,p2,p3 foo
```

To set a set of privileges in the allowed set of a file, use this command:

```
example % setfpriv –s –a p1,p2,p3 foo
```

To modify a set of privileges in the forced set of a file, use this command:

```
example% setfpriv –m –f p1,p2,p3 foo
```

To delete privileges in the forced set of a file, use this command:

```
example% setfpriv –d –f p1,p2,p3 foo
```

DIAGNOSTICS

setfpriv exits with one of the following values:

0 Successful completion.

1 Unsuccessful completion.

SEE ALSO

getfpriv(1TSOL), **testfpriv**(1TSOL), **getfpriv**(2TSOL), **setfpriv**(2TSOL)

NAME

setlabel – sets the CMW label for files

SYNOPSIS

setlabel [–**i** | –**s**] [–**h**] *newlabel filename* . . .

AVAILABILITY

SUNWtsolu

DESCRIPTION

setlabel sets the CMW label associated with each *filename.* Unless *newlabel* and *filename* have been specified, no labels will be set. Incremental changes to labels are supported.

Refer to **setcmwlabel**(2TSOL) for a complete description of conditions to satisfy and privileges needed to execute this command.

Users may enter a sensitivity label (SL), information label (IL), 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)C . 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.

NOTES

By convention, words appear in SLs in reverse order to the way they appear in ILs. Order does not matter on input. TS A B in an SL is displayed as TS B A in an IL.

Plus and minus signs may be used when modifying an existing label. They 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] }
or
*

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

The special case where the CMW label is an asterisk (∗) represents a CMW label whose SL is to be set equal to its IL.

OPTIONS

No options
: Set the information label portion and the sensitivity label portion of the CMW label.

–h Set the label of the symbolic link.

–i Set the information label portion of the CMW label.

–s Set the sensitivity label portion of the CMW label.

RETURN VALUES

setlabel exits with one of the following values:

0 Successful completion of **setlabel**.

1 Returned by **setlabel** due to usage error.

2 Error caused when getting, setting or translating the label.

USAGE

On the command line, enclose the label in double quotes unless the label you are entering is only one word. Without quotes, a second word or letter separated by a space is interpreted as a second argument. Labels containing **[** and **]** characters should be in quotes to suppress the shell's use of those characters in filename substitution.

```
setlabel -i "C A B" somefile
setlabel -s SECRET somefile
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
```

EXAMPLES

To set *somefile*'s IL to CONFIDENTIAL.

```
example% setlabel confidential somefile
```

To set *somefile*'s IL to ADMIN_LOW and SL to CONFIDENTIAL.

```
example% setlabel admin_low[confidential] somefile
```

To set *somefile*'s SL to SECRET A.

```
example% setlabel [Secret a] somefile
```

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

```
example% setlabel -s +b somefile
```

To turn off compartment A in somefile's SL.

```
example% 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.)

```
example% setlabel secret,b/a somefile
```

When the IL is SECRET B A, this example resets the IL to CONFIDENTIAL.

```
example% setlabel -i +confidential somefile
```

To set *somefile*'s IL to SECRET B.

```
example% setlabel secret a B -A somefile
```

NOTES

If an incremental change is being made to an existing label and the first character of the label is a hyphen (–), a preceding double-hyphen (––) is required; the double-hyphen must follow any of the –**i**, –**s**, and –**h** options. (See the examples.)

SEE ALSO

setcmwlabel(2TSOL)

NAME

tar – create tape archives and add or extract files

SYNOPSIS

tar c [**bBefFhiloPpTvwX** [**0-7**]] [*block*] [*tarfile*] [*exclude-file*]
    { –**I** *include-file* | –**C** "directory file" | *file* } . . .

tar r [**bBefFhilPpTvw** [**0-7**]] [*block*]
    { –**I** *include-file* | **-C** *directory file* | file } . . .

tar t [**BedfFhilTvX** [**0-7**]] [*tarfile*] [*exclude-file*] { –**I** *include-file* | *file* } . . .

tar u [**bBefFhilPpTvw** [**0-7**]] [*block*] [*tarfile*] *file* . . .

tar x [**BedfFhilmopTvwX** [**0-7**]] [*tarfile*] [*exclude-file*] [*file* . . .]

AVAILABILITY

SUNWcsu

DESCRIPTION

The **tar** command archives and extracts files to and from a single file called a *tarfile*. A tarfile is usually a magnetic tape, but it can be any file. **tar**'s actions are controlled by the *key* argument. The *key* is a string of characters containing exactly one function letter (**c**, **r**, **t** , **u**, or **x**) and zero or more function modifiers (letters or digits), depending on the function letter used. The *key* string contains no SPACE characters. Function modifier arguments are listed on the command line in the same order as their corresponding function modifiers appear in the *key* string.

The –**I** *include-file*, –**C** *directory file*, and *file* arguments specify which files or directories are to be archived or extracted. In all cases, appearance of a directory name refers to the files and (recursively) subdirectories of that directory. Arguments appearing within braces (**{ }**) indicate that one of the arguments must be specified.

The **tar** command provides the functionality to create, update, list the table of contents, and extract a tarfile that contains extended Trusted Solaris security attributes, MLD and SLD information. The **tar** command also provides the compatibility support to list the table of contents and extract a Trusted Solaris 1.x tarfile onto a Trusted Solaris 2.5 system. Two new function modifiers **T** and **d** are added to support these functions, and see below for their descriptions.

The **tar** command operates on a single file called the tarfile. The tarfile is essentially a sequence of the archived files. Each archived file contains the information that is needed to restore a file. When the tarfile contains Trusted Solaris extended security attributes, MLD and SLD information, each archived file is preceeded by its own ancillary file, which holds the extended security attributes, MLD and SLD information.

Without privileges, the **tar** command works within the Trusted Solaris security policy, which is enforced by the file system. When invoked by an ordinary user without privileges, **tar** works at a single sensitivity label and can be used only to create a tarfile at the sensitivity label of the current workspace.

OPTIONS

The following options are supported:

–**I** *include-file*

Open *include-file* containing a list of files, one per line, and treat as if each file appeared separately on the command line. Be careful of trailing white spaces. In the case where excluded files (see **X** function modifier) are also specified, they take precedence over all included files. If a file is specified in both the *exclude-file* and the *include-file* (or on the command line), it will be excluded.

–**C** *directory file*
Perform a **chdir** (see **cd**(1)) operation on *directory* and perform the **c** (create) or **r** (replace) operation on *file*. Use short relative path names for *file*. If *file* is '**.**', archive all files in *directory*. This option enables archiving files from multiple directories not related by a close common parent.

OPERANDS

The following operands are supported:

file A path name of a regular file or directory to be archived (when the **c**, **r** or **u** functions are specified), extracted (**x**) or listed (**t**). When *file* is the path name of a directory, the action applies to all of the files and (recursively) subdirectories of that directory. The directory portion of *file* (see **dirname**(1)) cannot exceed 155 characters. The file name portion (see **basename**(1)) cannot exceed 100 characters.

Function Letters

The function portion of the key is specified by one of the following letters:

c Create. Writing begins at the beginning of the tarfile, instead of at the end.

r Replace. The named *file*s are written at the end of the tarfile.

t Table of Contents. The names of the specified files are listed each time they occur in the tarfile. If no *file* argument is given, the names of all files in the tarfile are listed. With the **v** function modifier, additional information for the specified files is displayed.

u Update. The named *file*s are written at the end of the tarfile if they are not already in the tarfile, or if they have been modified since last written to that tarfile. An update can be rather slow. A tarfile created on a 5.x system cannot be updated on a 4.x system.

x Extract or restore. The named *file*s are extracted from the tarfile and written to the directory specified in the tarfile, relative to the current directory. Use the relative path names of files and directories to be extracted. If a named file matches a directory whose contents has been written to the tarfile, this directory is recursively extracted. The owner, modification time, and mode are restored (if possible); otherwise, to restore owner, **tar** must be run with user ID of 0. Character-special and block-special devices (created by **mknod**(1M)) can only be extracted when the tar program has asserted the **sys_devices** privilege. If no *file* argument is given, the entire content of the tarfile is extracted. If the tarfile contains several files with the same name, each file is written to the appropriate directory, overwriting the previous one. Filename substitution wildcards cannot be used for extracting files from the archive; rather, use a command of the form:

tar xvf... /dev/rmt/0 `tar tf... /dev/rmt/0 | grep '*pattern*'`

When extracting tapes created with the **r** or **u** functions, directory modification times may not be set correctly. These same functions cannot be used with many tape drives due to tape drive limitations such as the absence of backspace or append capabilities.

When using the **r**, **u**, or **x** functions or the **X** function modifier, the named files must match exactly the corresponding files in the *tarfile*. For example, to extract **./***thisfile*, you must specify **./***thisfile*, and not *thisfile*. The **t** function displays how each file was archived.

Function Modifiers

The characters below may be used in conjunction with the letter that selects the desired function.

b Blocking Factor. Use when reading or writing to raw magnetic archives (see **f** below). The *block* argument specifies the number of 512-byte tape blocks to be included in each read or write operation performed on the tarfile. The minimum is **1**, the default is **20**. The maximum value is a function of the amount of memory available and the blocking requirements of the specific tape device involved (see **mtio**(7I) for details.)

When a tape archive is being read, its actual blocking factor will be automatically detected, provided that it is less than or equal to the nominal blocking factor (the value of the *block* argument, or the default value if the **b** modifier is not specified). If the actual blocking factor is greater than the nominal blocking factor, a read error will result. See Example 5 in **EXAMPLES** below.

The automatic determination of the actual blocking factor may be fooled when reading from a pipe or a socket (see the **B** function modifier below).

1/4" streaming tape has an inherent blocking factor of one 512-byte block. It can be read or written using any blocking factor.

This function modifier works for archives on disk files and block special devices, among others, but is intended principally for tape devices.

B Block. Force **tar** to perform multiple reads (if necessary) to read exactly enough bytes to fill a block. This function modifier enables **tar** to work across the Ethernet, since pipes and sockets return partial blocks even when more data is coming. When reading from standard input, '–', this function modifier is selected by default to ensure that **tar** can recover from short reads.

d The function modifier **d** indicates the tarfile is in Trusted Solaris 1.x format. This function letter is not valid for the function letters **c**, **r**, or **u**. When this function modifier is used with the function letter **t** to display tarfile's contents, the **tar** program processes the input tarfile according to the Trusted Solaris 1.x format. If the function modifier **T** is also specified, then the contents of the Trusted Solaris 1.x **tarfile** is displayed with a line for each ancillary file and a line for each archived file. The line for an ancillary file has the same filename as its corresponding archived file, but it is suffixed by the string "(A)".

When this function modifier is used with the function letter **x** to extract a tarfile, the tar program processes the input tarfile according to the Trusted Solaris 1.x format. If the function modifier **T** is also specified, the appropriate MLD , SLD information and extended security attributes (which are valid on the Trusted Solaris 2.5 system) are used to restore each archived file.

e Error. Exit immediately with a positive exit status if any unexpected errors occur.

f File. Use the tarfile argument as the name of the tarfile. If **f** is specified, **/etc/default/tar** is not searched. If **f** is omitted, **tar** will use the device indicated by the **TAPE** environment variable, if set; otherwise, it will use the default values defined in **/etc/default/tar**. If the name of the tarfile is '–', **tar** writes to the standard output or reads from the standard input, whichever is appropriate. **tar** can be used as the head or tail of a pipeline. **tar** can also be used to move hierarchies with the command:

```
example% cd fromdir; tar cf – . | (cd todir; tar xfBp –)
```

F With one **F** argument, **tar** excludes all directories named SCCS and RCS from the tarfile. With two arguments, **FF**, **tar** excludes all directories named SCCS and RCS, all files with **.o** as their suffix, and all files named **errs**, **core**, and **a.out**.

h Follow symbolic links as if they were normal files or directories. Normally, **tar** does not follow symbolic links.

i Ignore directory checksum errors.

l Link. Output error message if unable to resolve all links to the files being archived. If **l** is not specified, no error messages are printed.

m Modify. The modification time of the file is the time of extraction. This function modifier is valid only with the **x** function.

o Ownership. Assign to extracted files the user and group identifiers of the user running the program, rather than those on tarfile. This is the default behavior for users when **tar** is not being run with the user ID of 0. If the **o** function modifier is not set and the **tar** command's user ID is 0, the extracted files will take on the group and user identifiers of the files on tarfile (see **chown**(1) for more information). The **o** function modifier is only valid with the **x** function.

p Restore the named files to their original modes, and ACLs if applicable, ignoring the present **umask**(1). SETUID and sticky information are also extracted for **tar** when the user ID is 0. When this function modifier is used with the **c** function, ACLs are created in the tarfile along with other information. Errors will occur when a tarfile with ACLs is extracted by previous versions of **tar**.

P Suppress the addition of a trailing "/" on directory entries in the archive.

T When this modifier is used with the function letter **c**, **r**, or **u** for creating, replacing or updating a tarfile, the extended security attributes, MLD and SLD information associated with each archived file are stored in the tarfile. The **tar** command also traverses any MLD it encounters. Hence, SLDs dominated by the **tar** process's sensitivity label are walked, or all SLDs are walked with certain privileges. Specifying **T** implies the function modifier **p.**

When used with the function letter **t**, the **tarfile** content is displayed with a line for each ancillary file and a line for each archived file. The line for an ancillary file has the same filename as its corresponding archived file, but it is suffixed by the string "(A)". Specifying **T** implies the function modifier **p.**

When used with the function letter **x** for extracting a **tarfile**, the tar program attempts to restore each archived file using the MLD and SLD information, and the extended security attributes. Specifying **T** implies the function modifier **p.**

v Verbose. Output the name of each file preceded by the function letter. With the **t** function, **v** provides additional information about the tarfile entries. The listing is similar to the format produced by the **–l** option of the **ls**(1) command.

w What. Output the action to be taken and the name of the file, then await the user's confirmation. If the first keystroke is **y**, the action is performed; otherwise, the action is not performed. This function modifier cannot be used with the **t** function.

X Exclude. Use the *exclude-file* argument as a file containing a list of relative path names for files (or directories) to be excluded from the tarfile when using the functions **c**, **x**, or **t**. Be careful of trailing white spaces. Multiple **X** arguments may be used, with one *exclude-file* per argument. In the case where included files (see **–I** *include-file* option) are also specified, the excluded files take precedence over all included files. If a file is specified in both the *exclude-file* and the *include-file* (or on the command line), it will be excluded.

[0-7] Select an alternative drive on which the tape is mounted. The default entries are specified in **/etc/default/tar**. If no digit or **f** function modifier is specified, the entry in **/etc/default/tar** with digit "**0**" is the default.

EXAMPLES

1. The following is an example using **tar** to create an archive of your home directory on a tape mounted on drive **/dev/rmt/0**:

```
example% cd
example% tar cvf /dev/rmt/0 .
messages from tar
```

The **c** function letter means create the archive; the **v** function modifier outputs messages explaining what **tar** is doing; the **f** function modifier indicates that the tarfile is being specified (**/dev/rmt/0** in this example). The dot (**.**) at the end of the command line indicates the current directory and is the argument of the **f** function modifier.

Display the table of contents of the tarfile with the following command:

```
example% tar tvf /dev/rmt/0
```

The output will be similar to the following:

```
rw-r--r--    1677/40    2123    Nov  7 18:15 1985       ./test.c
. . .
example%
```

The columns have the following meanings:

- column 1 is the access permissions to **./test.c**
- column 2 is the *user-id* / *group-id* of **./test.c**
- column 3 is the size of **./test.c** in bytes
- column 4 is the modification date of **./test.c**
- column 5 is the name of **./test.c**

To extract files from the archive:

```
example% tar xvf /dev/rmt/0
messages from tar
example%
```

If there are multiple archive files on a tape, each is separated from the following one by an EOF marker. To have **tar** read the first and second archives from a tape with multiple archives on it, the *non-rewinding* version of the tape device name must be used with the **f** function modifier, as follows:

```
example% tar xvfp /dev/rmt/0n       read first archive from tape
messages from tar
example% tar xvfp /dev/rmt/0n       read second archive from tape
messages from tar
example%
```

Note that in some earlier releases, the above scenario did not work correctly, and intervention with **mt**(1) between **tar** invocations was necessary. To emulate the old behavior, use the non-rewind device name containing the letter **b** for BSD behavior. See the **Close Operations** section of the **mtio**(7I) manual page.

2. To archive files from **/usr/include** and from **/etc** to default tape drive **0**:

```
example% tar c –C /usr  include –C /etc .
```

The table of contents from the resulting tarfile would produce output like the following:

```
include/
include/a.out.h
and all the other files in /usr/include . . .
./chown
and all the other files in /etc
```

To extract all files in the **include** directory:

```
example% tar xv include
x include/, 0 bytes, 0 tape blocks
and all files under include. . .
```

3. The following is an example using **tar** to transfer files across the Ethernet. First, here is how to archive files from the local machine (**example**) to a tape on a remote system (**host**):

```
example% tar cvfb – 20 files | rsh host dd of=/dev/rmt/0 obs=20b
messages from tar
example%
```

   In the example above, we are *creating* a tarfile with the **c** key letter, asking for *verbose* output from **tar** with the **v** function modifier, specifying the name of the output tarfile using the **f** function modifier (the standard output is where the tarfile appears, as indicated by the '–' sign), and specifying the blocksize (**20**) with the **b** function modifier. If you want to change the blocksize, you must change the blocksize arguments both on the **tar** command *and* on the **dd** command.

4. The following is an example that uses **tar** to retrieve files from a tape on the remote system back to the local system:

```
example% rsh –n host dd if=/dev/rmt/0 bs=20b | tar xvBfb – 20 files
messages from tar
example%
```

   In the example above, we are *extracting* from the tarfile with the **x** key letter, asking for *verbose output from* **tar** with the **v** function modifier, telling **tar** it is reading from a pipe with the **B** function modifier, specifying the name of the input tarfile using the **f** function modifier (the standard input is where the tarfile appears, as indicated by the '–' sign), and specifying the blocksize (**20**) with the **b** function modifier.

5. The following example creates an archive of the home directory on **/dev/rmt/0** with an actual blocking factor of 19.

```
example% tar cvfb /dev/rmt/0 19 $HOME
```

   To recognize this archive's actual blocking factor without using the **b** function modifier:

```
example% tar tvf /dev/rmt/0
tar: blocksize = 19
...
```

   To recognize this archive's actual blocking factor using a larger nominal blocking factor:

```
example% tar tvf /dev/rmt/0 30
tar: blocksize = 19
...
```

   Attempt to recognize this archive's actual blocking factor using a nominal blocking factor that is too small:

```
example% tar tvf /dev/rmt/0 10
tar: tape read error
```

6. The following is an example using **tar** to create a tarfile of the tartest directory and

save the extended security attributes, MLD and SLD information.

```
example% cd
example% tar cvfT onetarfile tartest
```

The output will be similar to the following:

```
a tartest/(A) 1K
a tartest/ 0K
a tartest/file1(A) 1K
a tartest/file1 0K
a tartest/mld1/(A) 1K
a tartest/mld1/ 0K
a tartest/mld1/(A) 1K
a tartest/mld1/ 0K
a tartest/mld1/file50(A) 1K
a tartest/mld1/file50 1K
...
```

The **c** function letter means create the archive; the **v** function modifier outputs messages explaining what **tar** is doing; the **f** function modifier indicates that the name of the **tarfile** to be created (**onetarfile** in this example). The **T** function modifier indicates that the extended security attributes, MLD and SLD information for each archived file are stored in the **tarfile**. The **tartest** is the name of the directory from which to create the **tarfile**.

The lines that end with (A) are the ancillary files for each archived file.

Display the table of contents of the tarfile (onetarfile in this example) with the following command:

```
example% tar tvfT onetarfile
```

The output will be similar to the following:

```
drwxr-xr-x 35436/10 54 Nov 11 17:07 1996 tartest/(A)
drwxr-xr-x+35436/10   0 Nov 11 17:07 1996 tartest/
-rw-r--r-- 35436/10  64 Nov 11 10:40 1996 tartest/file1(A)
-rw-r--r--+35436/10   0 Nov 11 10:40 1996 tartest/file1
drwxr-xr-x 35436/10 82 Nov 11 11:44 1996 tartest/mld1/(A)
drwxr-xr-x+35436/10 0 Nov 11 11:44 1996 tartest/mld1/
drwxr-xr-x 35436/10  87 Nov 11 11:33 1996 tartest/mld1/(A)
drwxr-xr-x+35436/10   0 Nov 11 11:33 1996 tartest/mld1/
-rw-r--r-- 35436/10 106 Nov 11 11:06 1996 tartest/mld1/file50(A)
-rw-r--r--+35436/10  17 Nov 11 11:06 1996 tartest/mld1/file50
...
```

The lines that end with (A) are ancillary files for each archived file.

Extract files from the tarfile (onetarfile in this example) with the following command:

```
example% tar xvfT onetarfile
```

The output will be similar to the following:

```
x tartest/(A), 54 bytes, 1 tape blocks
x tartest/, 0 bytes, 0 tape blocks
x tartest/file1(A), 64 bytes, 1 tape blocks
x tartest/file1, 0 bytes, 0 tape blocks
x tartest/mld1/(A), 82 bytes, 1 tape blocks
x tartest/.MLD.mld1/, 0 bytes, 0 tape blocks
x tartest/mld1/(A), 87 bytes, 1 tape blocks
x tartest/.MLD.mld1/.SLD.0/, 0 bytes, 0 tape blocks
x tartest/mld1/file50(A), 106 bytes, 1 tape blocks
x tartest/.MLD.mld1/.SLD.0/file50, 17 bytes, 1 tape blocks
. . .
```

The lines that end with (A) are ancillary files for each archived file.

ENVIRONMENT

See **environ**(5) for descriptions of the following environment variables that affect the execution of **tar**: **LC_COLLATE**, **LC_CTYPE**, **LC_MESSAGES**, **LC_TIME**, **TZ**, and **NLSPATH**.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

SUMMARY OF TRUSTED SOLARIS CHANGES

tar provides a function modifier **T** for creating, processing, and extracting a tarfile containing the extended security attributes, and MLD and SLD information. When an MLD is encountered in creating or updating a tarfile, the MLD is traversed according to the **tar** process's sensitivity label and privileges.

In addition, **tar** provides another function modifier for processing and extracting a tarfile created on a Trusted Solaris 1.x system. The function modifier **d** can be used only with the function letters **t** and **x**.

MAC restrictions apply when **tar** is used. Appropriate privileges may be required to override access checks that are enforced for the create, update and extract operations.

For creating or updating a tarfile, one or more of the following privileges may be required: **file_mac_read**, **file_mac_write**, **file_mac_search**, **file_dac_read**, **file_dac_write**, **file_dac_search**, or **sys_trans_label**.

The extended security attributes that require privileges to restore, are restored when the appropriate privileges are present. Hence, to successfully extract files from a tarfile and restore the extended security attributes, one or more of the following privileges may be required: **file_mac_read**, **file_mac_write**, **file_dac_read**, **file_dac_write**, **file_setdac**, **file_setid**, **file_chown**, **file_owner**, **file_downgrade_sl**, **file_downgrade_il**, **file_upgrade_sl**, **file_upgrade_il**, **file_setpriv**, **file_audit**, **sys_devices**, or **sys_trans_label**.

FILES

/dev/rmt/[0-7][b][n]
/dev/rmt/[0-7]l[b][n]
/dev/rmt/[0-7]m[b][n]
/dev/rmt/[0-7]h[b][n]
/dev/rmt/[0-7]u[b][n]
/dev/rmt/[0-7]c[b][n]

/etc/default/tar Settings may look like this:
archive0=/dev/rmt/0
archive1=/dev/rmt/0n
archive2=/dev/rmt/1
archive3=/dev/rmt/1n
archive4=/dev/rmt/0
archive5=/dev/rmt/0n
archive6=/dev/rmt/1
archive7=/dev/rmt/1n

/tmp/tar*

SEE ALSO

ar(1), **basename**(1), **cd**(1), **chown**(1), **cpio**(1), **csh**(1), **dirname**(1), **ls**(1), **mt**(1), **pax**(1), **setfacl**(1), **umask**(1), **mknod**(1M), **vold**(1M), **environ**(5), **mtio**(7I)

DIAGNOSTICS

Diagnostic messages are output for bad key characters and tape read / write errors, and for insufficient memory to hold the link tables.

NOTES

There is no way to access for the *n*-th occurrence of a file.

Tape errors are handled ungracefully.

When the Volume Management daemon is running, accesses to floppy devices through the conventional device names (for example, **/dev/rdiskette**) may not succeed. See **vold**(1M) for further details.

The **tar** archive format allows UIDs and GIDs up to 2097151 to be stored in the archive header. Files with UIDs and GIDs greater than this value will be archived with the UID and GID of **60001**.

Notes for function modifier **T** and **d :**

For both Trusted Solaris 1.x and Trusted Solaris 2.5 tarfiles, a compatible **label_encodings**(4TSOL) file is expected between the time the tarfile is created or updated and the time the tarfile is extracted.

When a Trusted Solaris 1.x tarfile is restored on a Trusted Solaris 2.5 system, the label SYSTEM_HIGH is mapped to the label ADMIN_HIGH, and the label SYSTEM_LOW is mapped to label ADMIN_LOW. In addition, the privileges and file audit mask are not used for the restored files because their formats are not compatible with Trusted Solaris 2.x equivalent security attributes.

If the name of the linked file in a symbolic link contains explicitly adorned MLD names and / or SLD names, it may no longer be a valid pathname after extraction. The reason is that the MLD adornment and SLD name at the time the tarfile is created or updated might be different than they are at the time the tarfile is extracted. At extraction time, **tar** attempts to update the link pathname of the symbolic link with the proper MLD adornment and SLD name. If **tar** fails, an error message is issued. Users need to perform any corrections themselves after the extraction is done.

Extracting a Trusted Solaris 2.5 tarfile on a standard Solaris 2.5 system may cause directory-checksum errors. Use the **–i** option, which ignores directory-checksum errors, to get around this problem.

NAME

testfpriv – Check or test the privilege sets associated with a file

SYNOPSIS

/usr/bin/testfpriv [–**s**] [[–**e**] –**a** *privseta*] [[–**e**] –**f** *privsetf*] *filename*

AVAILABILITY

SUNWtsolu

DESCRIPTION

Check or test the privilege sets of a file or files. The invoking process must have MAC read permission.

privseta and *privsetf* are one of these:

- A comma-separated list of ASCII privilege names as reported by **getfpriv**
- A comma-separated list of numeric privilege IDs as found in **<sys/tsol/priv_names.h>**
- The keyword **all** to indicate all privileges

No white space may exist in either list.

Without the –**e** (equal) option, the specified set of privileges is checked as a subset of the forced or the allowed privileges specified on the command line. The **testfpriv** function reports those privileges that are specified in *privseta* and *privsetf* but not found in the allowed or forced sets of the file. The –**e** option also reports privileges that the file has but that were not specified in the **testfpriv** command.

The privilege sets of each named file are checked according to options described in the next section.

OPTIONS

–**a** Test whether *privseta* is either equal to or a subset of the allowed set of *filename.*

–**e** Test the equality of *privset* and the privilege set of *filename.*

–**f** Test whether *privsetf* is either equal to or a subset of the forced set of *filename.*

–**s** Use silent mode to suppress output. (This option is useful in shell scripts that need only the return value.)

RETURN VALUES

testfpriv exits with one of these values:

0 Specified privileges are in the allowed or the forced set of the file. With the –**e** option, the specified privileges are equal to the allowed set or the forced set of the file.

1 The specified privileges are not in the allowed set of the file, or (with –**e**) the allowed set of the file contains privileges not specified in this command.

2 The specified privileges are not in the forced set of the file, or (with –**e**) the forced set of the file contains privileges not specified in this command.

3 Both the allowed and forced sets have mismatches as described for return values **1** and **2**.

4 **testfpriv** completed unsuccessfully.

EXAMPLE

To determine if a set of privileges are in the forced set of a file, use this command:

```
example% testfpriv –f p1,p2,p3 foo
```

If all the specified privileges are in the forced set of file **foo**, no output is returned. If any of the privileges is not in the forced set of file **foo**, the function displays the missing privilege(s). For example,

```
example% foo:missing:p2
```

To test if a file's forced and allowed sets are exactly equal to the specified privileges, use this command:

```
example% testfpriv -e -f p1 -e -a p2 foo
```

If the file's privileges did not match the specified privileges exactly, the output could be of this example format:

```
example% foo:forced:extra:p3:allowed:missing:p2:extra:p4
```

You can test both the allowed and the forced sets at the same time. For example, use this command to test for all bits on in the allowed set, and whether only **p1** and **p2** are present in the forced set:

```
example% testfpriv  -s  -e  -a all  -f p1,p2 foo
```

Because this example uses the silent mode, no output is returned. The returned exit value demonstrates the result.

SEE ALSO

getfpriv(1TSOL), **setfpriv**(1TSOL), **getfpriv**(2TSOL), **setfpriv**(2TSOL)

NAME

uname – Print name of current system

SYNOPSIS

uname [–**aimnprsv**]
uname [–**S** *system_name*]

AVAILABILITY

SUNWcsu

DESCRIPTION

The **uname** utility prints information about the current system on the standard output. When options are specified, symbols representing one or more system characteristics will be written to the standard output. If no options are specified, **uname** prints the current operating system's name. The options print selected information returned by **uname**(2), **sysinfo**(2), or both.

OPTIONS

These options are supported:

–**a** Print all information.

–**i** Print the name of the hardware implementation (platform).

–**m** Print the machine hardware name (class). Use of this option is discouraged; use **uname** –**p** instead. See **NOTES**.

–**n** Print the nodename. (The nodename is the name by which the system is known to a communications network.)

–**p** Print the current host's ISA or processor type.

–**r** Print the operating system release level.

–**s** Print the name of the operating system. This is the default.

–**v** Print the operating system version.

–**S** *system_name* The nodename may be changed by specifying a *system_name* argument. The *system_name* argument is restricted to **SYS_NMLN** characters. **SYS_NMLN** is an implementation-specific value defined in **<sys/utsname.h>**.

EXAMPLES

This command prints the operating system name and release level, separated by one SPACE character:

```
example% uname –sr
```

ENVIRONMENT

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

EXIT STATUS

Upon successful completion, **uname** returns **0**. If an error occurred, **uname** returns a value **>0**.

SUMMARY OF TRUSTED SOLARIS CHANGES

To succeed with the **-S** option, this command needs the **sys_net_config** privilege. If a user other than root attempts this option, the command also needs the **file_dac_read**, **file_dac_write**, **file_mac_read**, and **file_mac_write** privileges to update the **/etc/nodename** file.

SEE ALSO

sysinfo(2), **uname**(2), **environ**(5)

NOTES

Independent software vendors (ISVs) and others who need to determine detailed characteristics of the platform on which their software is being either installed or executed should use the **uname** command.

To determine the operating system name and release level, use **uname –sr**. To determine only the operating system release level, use **uname –r**. Note that operating system release levels are not guaranteed to be in *x.y* format (such as 5.3, 5.4, 5.5, and so forth); future releases could be in the *x.y.z* format (such as 5.3.1, 5.3.2, 5.4.1, and so forth).

In SunOS 4.*x* releases, the **arch** command was often used to obtain information similar to that obtained by using the **uname** command. The **arch** command output "sun4" was often incorrectly interpreted as a SunOS SPARC system. If hardware platform information is desired, use **uname –sp**.

The **arch –k** and **uname –m** commands return equivalent values; however, the use of either of these commands by third-party programs is discouraged, as is the use of the **arch** command in general. To determine the machine's Instruction Set Architecture (ISA or processor type), use **uname** with the **–p** option.

Index