

Trusted Solaris 2.5.1 Man Pages: 4TSOL File Formats

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

Copyright 1998 Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, California 94043-1100 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19.

Sun, Sun Microsystems, the Sun logo, SunSoft, Solaris, SunOS, OpenWindows, DeskSet, ONC, ONC+, and NFS are trademarks, or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

RESTRICTED RIGHTS : Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

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.

Copyright 1997 Sun Microsystems, Inc., 2550 Garcia Avenue, Mountain View, Californie 94043-1100 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, SunSoft, Solaris, SunOS, OpenWindows, DeskSet, ONC, ONC+, et NFS sont des marques de fabrique ou des marques déposées, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

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 REPOUDRE 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.

Portions © AT&T 1983-1990 and reproduced with permission from AT&T.

Preface

The Trusted Solaris operating environment is based on the SunOS operating system and other components of the Solaris operating environment and also bundles security-enhanced versions of the Common Desktop Environment (CDE), X windows, and Solstice AdminSuite tools. Therefore, the Trusted Solaris Reference Manual includes man pages not only for the operating system but also for the other products included in the Trusted Solaris product as well. In the Trusted Solaris Reference Manual, as in other UNIX reference manuals, each collection of information on a particular 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”? Man pages are not intended to be tutorials. Depending what you are trying to do, refer to the Trusted Solaris user, developer, or 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 typically delivered on a CD. Whoever installs a system usually decides whether or not all the packages are also installed. Installing the documentation packages is optional, because they are not required for operations. As a result, not everyone has access to every package. The packages that contain man pages in the Trusted Solaris operating environment are: SUNWman, plus SUNWaudmo , SUNWdtma , SUNWdtmad ,

SUNWkcsrt , SUNWkcspg , SUNWmfman , SUNWmfrun , SUNWolman , SUNWrtvcu , SUNWsadmm , SUNWtltkm , SUNWxwacx , SUNWxwman , SUNWxwplt , and SUNWxwpmn.

The first means of accessing the man pages is by using the **man(1)** command to view the man pages online. An account can use the **man** command when the man page package that contains the desired man page is available on the local system or mounted from a remote server, if, in addition, a terminal emulator (such as **dtterm(1)**) and the **man(1)** command are in one of the account's execution profiles. (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 list directory's contents, a user enters the command: .

The second way to read man pages is by looking them up in the printed Trusted Solaris Reference Manual, which is in the Trusted Solaris documentation set, part number: TS2DS-251-9999.

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 or mounted from a remote server, anyone with the **answerbook(1)** command and a terminal emulator in an execution profile can display any of the man pages in the Trusted Solaris reference manual. The Trusted Solaris AnswerBook CD is packaged with the Trusted Solaris software CD. After the AnswerBook tool is launched, clicking the AnswerBook Navigator Search button brings up a dialog box where the name of a man page or terms contained in a man page can be entered to locate a specific man page.

For access to all available man pages for the operating system and for the bundled CDE, X windows, and Solstice AdminSuite products, the following man directories should be set in the MANPATH environment variable: **/usr/man**, **/usr/openwin/man**, and **/usr/dt/man**. For more about the format and contents of the man pages, see also the information in the **Intro** man pages for each section.

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 added or modified from the base Solaris or bundled products.

- 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 compatibility, they will be removed in future releases. Also, the BSD print management commands are not useful for managing print jobs on remote printers.

- Section 1MTSOL describes Trusted Solaris system maintenance and administration commands.
- Section 1TSOL describes modified user commands from the base SunOS operating system, and new Trusted Solaris user commands.
- Section 2TSOL describes Trusted Solaris new or modified 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 documentation for Trusted Solaris macros.
- 7*TSOL subsections describe various special files that refer to specific hardware peripherals and device drivers.

Subsections include: 7DTSOL, 7MTSOL, and 7TSOL.

- 9*TSOL subsections provide reference information for writing device drivers in the kernel operating system environment.

Trusted Solaris subsections are: 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 3RTSOL 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 availability 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, the security administrator can use **pkgadd(1)** or **swmtool(1)** to install the missing package.

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 how to use a command or function. Wherever possible a complete example including command line entry and machine response is shown. When an example is given for a command entered by a normal user, the prompt is shown as

example%

If the user must be in an administrative role, the example uses either the profile shell prompt for the secadmin or admin roles:

\$

or the root role prompt: **#**

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 described throughout the man page in a single easy-to-find place.

NAME	Intro, intro – introduction to file formats
DESCRIPTION	<p>This section outlines the formats of various files. The C structure declarations for the file formats are given where applicable. Usually, the headers containing these structure declarations can be found in the directories <code>/usr/include</code> or <code>/usr/include/sys</code>. For inclusion in C language programs, however, the syntax <code>#include <filename.h></code> or <code>#include <sys/filename.h></code> should be used.</p> <p>Because the operating system now allows the existence of multiple file system types, there are several instances of multiple manual pages with the same name. These pages all display the name of the FSType to which they pertain in the form <code>name_<fstype></code> at the top of the page. For example, <code>fs_ufs(4)</code></p> <p>Section 4 of the <i>Trusted Solaris Reference Manual</i> describes, in alphabetical order, the configuration files available with the Trusted Solaris operating environment, which is based on the Solaris operating environment and includes security-enhanced versions of the Common Desktop Environment (CDE) window system, and the Solstice AdminSuite set of system administration tools. Configuration files are usually not edited on the command line. Solstice AdminSuite tools in the <code>Solstice_Apps</code> folder and actions in the <code>System_Admin</code> folder in the Application Manager are used by administrative roles instead, because these tools audit all changes and preserve the required owner, group, permissions and sensitivity labels of the files. Man pages whose section IDs end with the 4TSOL suffix describe files that are either new or modified to work within Trusted Solaris security policy. An example of a new Trusted Solaris file is <code>device_allocate</code>, which is described on the <code>device_allocate(4TSOL)</code> man page.</p> <p>Modified files are files from any of the base products that have been modified to work within the Trusted Solaris <i>security policy</i>, such as: <code>proc</code>. Man pages for modified files have been rewritten to remove information that is not accurate for how the file is used within the Trusted Solaris system. Modified man pages, such as <code>proc(4TSOL)</code>, also add descriptions for any new fields or entities added to the file.</p>
NOTE	<p>The <code>answerbook(1)</code> and printed versions of the <i>Trusted Solaris Reference Manual</i> include only the Trusted Solaris man pages, while the online man pages that are viewable with the <code>man(1)</code> command include all the base man pages along with the Trusted Solaris man pages. The printed versions of the base Solaris section 4 man pages are in the Solaris reference manual. The <code>man</code> 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 <code>man</code> command with the <code>-s</code> option to specify the base section ID of the man page. For example, to display the <code>proc(4)</code> man page instead of the modified <code>proc(4TSOL)</code> man page, you would enter: <code>man -s4 proc</code>. To find out all the sections that contain man pages with the same name, enter: <code>man -l</code></p>

RULES FOR INCLUDING LABELS IN A CONFIGURATION FILE

<man_page_name>.

Follow the rules described here when entering labels in configuration files. When entering labels in graphical user interfaces, see **Rules for the Display and Entering in Intro(1TSOL)**. When entering labels on the command line in a UNIX shell, follow the rules in **Intro(1MTSOL)**.

Make sure that a program reading a configuration file can tell where the label starts and ends. Where the label is imbedded, as it is in the **device_allocate(4TSOL)** file, the only valid character to begin the label and terminate it is a semicolon (;). Most configuration files do not support label incrementations using plus or minus signs.

Configuration files are generally maintained at a sensitivity label of ADMIN_LOW . However, each site can choose whether to store labels in configuration files as text or as hexadecimal numbers, depending on the site's security policy, and the form used affects the sensitivity label at which the file should be stored. When labels are stored in human-readable form, the files that contain them must be protected at ADMIN_HIGH, so only administrative roles that have the ADMIN_HIGH label in their clearance can view the files. Also, if a file contains a collection of data written by all processes in the system (like the system log, **/dev/kmem**, and **/dev/mem** files) that file should be protected at the ADMIN_HIGH sensitivity label. NOTE Labels entered in text form must be quoted.

POLICY FOR SECURITY ATTRIBUTES ON CONFIGURATION FILES

The default user and group for configuration files are **root** and **sys** and default permissions are 00644. However, the security administrator should ensure that files that contain sensitivity information other than labels, such as those files that specify which activities are being audited system-wide, are not be publically readable. These files should have other permissions, owner and group IDs, and possibly a protective label.

SEE ALSO

Trusted Solaris Administrator's Procedures

Trusted Solaris Developer's Guide

Name	Description
audit_class(4TSOL)	audit class definitions
audit_control(4TSOL)	control information for system audit daemon
audit_data(4TSOL)	current information on audit daemon
audit_event(4TSOL)	audit event definition and class mapping
audit_user(4TSOL)	per-user auditing data file
audit.log(4TSOL)	audit trail file
auth_desc(4TSOL)	descriptions of defined authorizations
auth_name(4TSOL)	Trusted Solaris authorizations database
config_data(4TSOL)	Trusted Solaris configuration data for network and Jumpstart installations
config.privs(4TSOL)	list of window privileges that override system checks

device_allocate (4TSOL)	device_allocate file
device_deallocate (4TSOL)	device_deallocate file
device_maps (4TSOL)	device_maps file
device_policy (4TSOL)	device_policy file
inetd.conf (4TSOL)	Internet servers database
inittab (4TSOL)	script for init
label_encodings (4TSOL)	label encodings file
mnttab (4TSOL)	mounted file system table
nsswitch.conf (4TSOL)	configuration file for the name service switch
priv_desc (4TSOL)	descriptions of defined privileges
priv_name (4TSOL)	privilege name database file
proc (4TSOL)	/proc, the process file system
resolv.conf (4TSOL)	configuration file for name server routines
rmtab (4TSOL)	remote mounted file system table
sel_config (4TSOL)	selection rules for copy, cut, paste, drag and drop operations
sharetab (4TSOL)	shared file system table
tndlog (4TSOL)	log of tnd debug information
tnidb (4TSOL)	trusted network interface control database
tnrhdb (4TSOL)	trusted network remote host database
tnrhtp (4TSOL)	trusted network remote host templates
tsolgateways (4TSOL)	static routing configuration file
tsolinfo (4TSOL)	package security-attribute description file
tsolprof (4TSOL)	Trusted Solaris user profiles database
tsoluser (4TSOL)	Trusted Solaris user security attributes database
vfstab (4TSOL)	table of file system defaults
vfstab_adjunct (4TSOL)	attribute data file for mounting a filesystem

NAME	audit.log – File for audit trail																								
SYNOPSIS	#include <bsm/audit.h> #include <bsm/audit_record.h>																								
DESCRIPTION	<p>audit.log files are the repository for audit records stored on a local machine or on an audit server. These files are kept in directories named in the file audit_control(4TSOL). File names reflect the time the files are created and are, when possible, renamed to reflect the time they are closed as well. The name takes the form</p> <p style="text-align: center;"><i>yyyymmddhhmmss.not_terminated.hostname</i></p> <p>when open. If the auditd(1MTSOL) terminated gracefully and the file closed properly, the form is</p> <p style="text-align: center;"><i>yyyymmddhhmmss.yyyyymmddhhmmss.hostname</i></p> <p>where <i>yyyy</i> is the year; <i>mm</i>, the month; <i>dd</i>, the day in the month; <i>hh</i>, the hour in the day; <i>mm</i>, the minute in the hour; and <i>ss</i>, the second in the minute. All fields have fixed width. The audit.log file begins with a stand-alone file token and typically ends with one also. The beginning file token records the pathname of the previous audit file; the ending file token records the pathname of the next audit file. If the file name is NULL, the appropriate path was unavailable.</p> <p>The audit.log files contain audit records. Each audit record is made up of <i>audit tokens</i>. Each record contains a header token followed by various data tokens. Depending on the audit policy in place by auditon(2TSOL), other optional tokens such as trailers or sequences may be included.</p> <p>The tokens are defined as follows:</p> <p>The file token consists of</p> <table border="0" style="margin-left: 40px;"> <tr> <td>token ID</td> <td>char</td> </tr> <tr> <td>seconds of time</td> <td>u_int</td> </tr> <tr> <td>microseconds of time</td> <td>u_int</td> </tr> <tr> <td>file name length</td> <td>short</td> </tr> <tr> <td>file pathname</td> <td>null-terminated string</td> </tr> </table> <p>The header token consists of</p> <table border="0" style="margin-left: 40px;"> <tr> <td>token ID</td> <td>char</td> </tr> <tr> <td>record byte count</td> <td>u_long</td> </tr> <tr> <td>version #</td> <td>char (1)</td> </tr> <tr> <td>event type</td> <td>u_short</td> </tr> <tr> <td>event modifier</td> <td>u_short</td> </tr> <tr> <td>seconds of time</td> <td>u_int</td> </tr> <tr> <td>nanoseconds of time</td> <td>u_int</td> </tr> </table>	token ID	char	seconds of time	u_int	microseconds of time	u_int	file name length	short	file pathname	null-terminated string	token ID	char	record byte count	u_long	version #	char (1)	event type	u_short	event modifier	u_short	seconds of time	u_int	nanoseconds of time	u_int
token ID	char																								
seconds of time	u_int																								
microseconds of time	u_int																								
file name length	short																								
file pathname	null-terminated string																								
token ID	char																								
record byte count	u_long																								
version #	char (1)																								
event type	u_short																								
event modifier	u_short																								
seconds of time	u_int																								
nanoseconds of time	u_int																								

The trailer token consists of	
token ID	char
trailer magic number	u_short
record byte count	u_long
The arbitrary data token is defined	
token ID	char
how to print	char
basic unit	char
unit count	char
data items	depends on basic unit
The in_addr token consists of	
token ID	char
internet address	long
The ip token consists of	
token ID	char
version and ihl	char
type of service	char
length	short
id	u_short
offset	u_short
ttl	char
protocol	char
checksum	u_short
source address	long
destination address	long
The ipport token consists of	
token ID	char
port address	short
The opaque token consists of	
token ID	char
size	short
data	char, <i>size</i> chars
The path token consists of	
token ID	char
path length	short
path	null-terminated string

The process token consists of	
token ID	char
auid	u_long
euid	u_long
egid	u_long
ruid	u_long
rgid	u_long
pid	u_long
sid	u_long
terminal ID	u_long (port ID) u_long (machine ID)
The return token consists of	
token ID	char
error number	char
return value	u_int
The subject token consists of	
token ID	char
auid	u_long
euid	u_long
egid	u_long
ruid	u_long
rgid	u_long
pid	u_long
sid	u_long
terminal ID	u_long (port ID) u_long (machine ID)
The System V IPC token consists of	
token ID	char
object ID type	char
object ID	long
The text token consists of	
token ID	char
text length	short
text	null-terminated string
The attribute token consists of	
token ID	char
mode	u_long
uid	u_long
gid	u_long
file system id	long
node id	long
device	u_long

The groups token consists of	
token ID	char
number	short
group list	long, <i>size</i> chars
The System V IPC permission token consists of	
token ID	char
uid	u_long
gid	u_long
cuid	u_long
cgid	u_long
mode	u_long
seq	u_long
key	long
The arg token consists of	
token ID	char
argument #	char
argument value	long
string length	short
text	null-terminated string
The exec_args token consists of	
token ID	char
count	short
text	<i>count</i> null-terminated string(s)
The exec_env token consists of	
token ID	char
count	short
text	<i>count</i> null-terminated string(s)
The exit token consists of	
token ID	char
status	long
return value	long
The socket token consists of	
token ID	char
socket type	short
local port	short
local Internet address	long
remote port	short
remote Internet address	long
The seq token consists of	
token ID	char
sequence number	long

The acl token consists of	
token ID	char
num of entries	int
(following three fields repeated num times)	
object type	int
uid/gid	int
permissions	short
The clearance token consists of	
token ID	char
CLEARANCE	
label ID	char
pad character	char
classification	short
compartments	8 ints
The host token consists of	
token ID	char
local Internet address	long
The ilabel token consists of	
token ID	char
ILABEL	
label ID	char
pad character	char
classification	short
compartments	8 ints
markings	8 ints
The liaison token consists of	
token ID	char
liaison ID	int
The priv token consists of	
token ID	char
succ/fail	char
priv. used	int
The privilege token consists of	
token ID	char
type of set	char
priv. set	4 ints
The slabel token consists of	
token ID	char
SLABEL	
pad character	char
classification	short
compartments	8 ints

The xatom token consists of	
token ID	char
string length	short
atom string	string length bytes
The xcolormap token consists of	
token ID	char
XID	int
creator UID	int
The xcursor token consists of	
token ID	char
XID	int
creator UID	int
The xfont token consists of	
token ID	char
XID	int
creator UID	int
The xgc token consists of	
token ID	char
XID	int
creator UID	int
The xpixmap token consists of	
token ID	char
XID	int
creator UID	int
The xproperty token consists of	
token ID	char
XID	int
creator UID	int
string length	short
string	string length bytes
The xselect token consists of	
token ID	char
property length	short
property string	property length bytes
prop. type len.	short
prop type	prop. type len. bytes
data length	short
window data	data length bytes
The xwindow token consists of	
token ID	char
XID	int
creator UID	int

**SUMMARY OF
TRUSTED
SOLARIS
CHANGES**

These audit tokens have been added to the Trusted Solaris auditing module: **acl**, **clearance**, **host**, **ilabel**, **liaison**, **priv**, **privilege**, **slabel**, **xatom**, **xcolormap**, **xcursor**, **xfont**, **xgc**, **xpixmap**, **xproperty**, **xselect**, and **xwindow**. Trusted Solaris auditing also uses **auditwrite**(3TSOL) instead of **au_to_***(3) function calls to create audit tokens.

SEE ALSO

audit(1MTSOL), **auditd**(1MTSOL), **audit**(2TSOL), **auditon**(2TSOL), **audit_control**(4TSOL) *Trusted Solaris Audit Administration Manual*

NOTES

This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See *Trusted Solaris Audit Administration Manual* for how to disable and enable auditing.

The Trusted Solaris 2.x audit module primarily uses **auditwrite**(3TSOL) to create audit tokens. In the Solaris audit module, each attribute is generally written using the **au_to_***(3) family of function calls.

NAME	audit_class – File storing audit-class definitions				
SYNOPSIS	<code>/etc/security/audit_class</code>				
AVAILABILITY	SUNWcsu				
DESCRIPTION	<p><code>/etc/security/audit_class</code> is an ASCII system file that stores class definitions. Programs use the <code>getauclassent</code>(3TSOL) routines to access this information.</p> <p>The fields for each class entry are separated by colons. Each class entry is a bitmap and is separated from the next by a new line. Each entry in the audit_class file has this form:</p> <p style="padding-left: 40px;"><i>mask:name:description</i></p> <p>The fields are defined as follows:</p> <p><i>mask</i> The class mask</p> <p><i>name</i> The class name</p> <p><i>description</i> The description of the class</p> <p>The classes are now user-configurable. Each class is represented as a bit in the class mask, which is an unsigned integer. Thus, there are 32 different classes available, plus two metaclasses—all and no: all represents a conjunction of all allowed classes and provides a shorthand method of specifying all classes; no is the “invalid” class; any event mapped solely to this class will not be audited. Turning on auditing to the all metaclass will <i>not</i> cause events mapped solely to the no class to be written to the audit trail.</p>				
EXAMPLES	<p>Here is a sample of an audit_class file:</p> <pre> 0x00000000:no:invalid class 0x00000001:fr:file read 0x00000002:fw:file write 0x00000004:fa:file attribute access 0x00000008:fm:file attribute modify 0x00000010:fc:file create 0x00000020:fd:file delete 0x00000040:cl:file close 0xffffffff:all:all classes </pre>				
SUMMARY OF TRUSTED SOLARIS CHANGES	By default, the audit module has been enabled on Trusted Solaris systems.				
FILES	<table border="0"> <tr> <td style="padding-right: 20px;"><code>/etc/security/audit_class</code></td> <td>Class definitions</td> </tr> <tr> <td><code>/usr/include/bsm/audit.h</code></td> <td>Definitions corresponding to those in <code>/etc/security/audit_class</code></td> </tr> </table>	<code>/etc/security/audit_class</code>	Class definitions	<code>/usr/include/bsm/audit.h</code>	Definitions corresponding to those in <code>/etc/security/audit_class</code>
<code>/etc/security/audit_class</code>	Class definitions				
<code>/usr/include/bsm/audit.h</code>	Definitions corresponding to those in <code>/etc/security/audit_class</code>				

SEE ALSO**getauclassent(3TSOL), audit_event(4TSOL)****NOTES**

This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See *Trusted Solaris Audit Administration Manual* for how to disable and enable auditing.

It is possible to deliberately turn on the **no** class in the kernel; in that case, the audit trail will be flooded with records for the audit event AUE_NULL because if the user remaps the **no** class to something other than 0, every event that is currently mapped to **no** will be audited.

NAME	audit_control – File containing control information for system audit daemon
SYNOPSIS	/etc/security/audit_control
AVAILABILITY	SUNWcsu
DESCRIPTION	<p>The audit_control file contains audit-control information used by auditd(1MTSOL). Each line consists of a title and a string, separated by a colon. There are no restrictions on the order of lines in the file although some lines must appear only once. A line beginning with a pound sign (#) is a comment.</p> <p>Directory-definition lines list the directories to be used when creating audit files, in the order in which the directories are to be used. The format of a directory line is</p> <p style="padding-left: 40px;">dir: <i>directory-name</i></p> <p><i>directory-name</i> specifies the directory in which the audit files will be created. Any valid writable directory can be specified.</p> <p>Unless explicitly told to look elsewhere, the auditreduce(1MTSOL) command by default looks for the audit trail in all directories named according to the following convention on the server on which the command is run. Therefore, this naming convention is recommended for directories in which audit-trail files are stored:</p> <p style="padding-left: 40px;">/etc/security/audit/server[.number]/files</p> <p><i>server</i> is the name of the audit server on which the audit files are stored. The optional <i>.number</i> is used when an audit server exports two or more audit partitions. For example, the audit server trustworthy exports /etc/security/audit/trustworthy and /etc/security/audit/trustworthy.1. For the current host to use both of these partitions, these lines must be added to the local audit_control file:</p> <p style="padding-left: 40px;">dir: /etc/security/audit/trustworthy/files dir: /etc/security/audit/trustworthy.1/files</p> <p>Audit data may be stored in directories with other names at the discretion of the site. Some sites may want to store each host's audit data in a separate subdirectory. The audit structure used will depend on each individual site. If the defined audit structure differs from /etc/security/audit/*/files, auditreduce needs to be given the new location of the audit trail explicitly as described in auditreduce(4TSOL).</p> <p>The audit-threshold line specifies the percentage of free space that must be present in the file system containing the current audit file. The format of the threshold line is</p> <p style="padding-left: 40px;">minfree: <i>percentage</i></p> <p>where <i>percentage</i> indicates the amount of free space required. If free space falls below this threshold, the audit daemon auditd(1MTSOL) invokes the shell script audit_warn(1MTSOL). If no threshold is specified, the default is 0 percent.</p> <p>The audit-flags line specifies the default system audit value. This value is combined with the user audit value read from audit_user(4TSOL) to form the process audit state. The user audit value overrides the system audit value. The format of a flags line is</p>

flags: *audit-flags*

where *audit-flags* specifies which event classes are to be audited. The character string representation of *audit-flags* contains a series of flag names, each one identifying a single audit class, separated by commas. A name preceded by minus (-) means that the class should be audited for failure only; successful attempts are not audited. A name preceded by plus (+) means that the class should be audited for success only; failing attempts are not audited. Without a prefix, the name indicates that the class is to be audited for both successes and failures. The special string **all** indicates that all events should be audited: **-all** indicates that all failed attempts are to be audited; **+all**, all successful attempts. These prefixes turn off flags specified earlier in the string: caret (^), caret minus (^-), and caret plus (^+). Caret minus (^-) turns off audits for failed attempts; caret plus (^+) turns off audits for successful attempts; caret (^) turns off audits for both successful and failed attempts. These operators are typically used to reset flags.

The nonattributable-flags line is similar to the flags line but contains the audit flags that define which event classes are audited when an action cannot be attributed to a specific user. The format of a **naflags** line is

naflags: *audit-flags*

The flags are separated by commas, with no spaces.

These are some of the predefined audit classes:

short name	long name	Short description
no	no_class	Null value for turning off event preselection
fr	file_read	Read of data, open for reading, etc.
fw	file_write	Write of data, open for writing, etc.
fa	file_attr_acc	Access of object attributes: stat, pathconf, etc.
fm	file_attr_mod	Change of object attributes: chown, flock, etc.
fc	file_creation	Creation of object
fd	file_deletion	Deletion of object
cl	file_close	close(2) system call
pc	process	process operations
nt	network	Network events: bind, connect, accept, etc.
ip	ipc	System V IPC operations
na	non_attrib	Nonattributable events
ad	administrative	Administrative actions: mount, exportfs, etc.
lo	login_logout	Login and logout events
ap	application	Application auditing
io	ioctl	ioctl(2) system call
fn	fcntl	fcntl(2) system call
ot	other	Everything else
all	all	All flags set

Note that the classes are configurable; see **audit_class**(4TSOL).

EXAMPLES

Here is a sample `/etc/security/audit_control` file for the machine `eggplant`:

```

dir: /etc/security/audit/jedgar/eggplant/files
dir: /etc/security/audit/jedgar.aux/eggplant/files

#
# Last-ditch audit file system when jedgar fills up.
#
dir: /etc/security/audit/eggplant/files
minfree: 20
flags: lo,ad,-all,^-fm
naflags: lo,ad

```

This example identifies server `jedgar` with two file systems normally used for audit data and another server `eggplant` used only when `jedgar` fills up or breaks, and specifies that the warning script is run when the file systems are 80 percent filled. All logins and administrative operations are to be audited (regardless of whether they succeed or are attributable); failures of all types except failures to access object attributes are to be audited.

SUMMARY OF TRUSTED SOLARIS CHANGES

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

FILES

```

/etc/security/audit_control
/etc/security/audit_warn
/etc/security/audit/*/*/*
/etc/security/audit_user

```

SEE ALSO

`audit(1MTSOL)`, `audit_warn(1MTSOL)`, `auditd(1MTSOL)`, `audit(2TSOL)`, `getfauditflags(3TSOL)`, `audit.log(4TSOL)`, `audit_class(4TSOL)`, `audit_user(4TSOL)`, *Trusted Solaris Audit Administration Manual*

NOTES

This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See *Trusted Solaris Audit Administration Manual* for how to disable and enable auditing.

NAME	audit_data – File containing current information on audit daemon
SYNOPSIS	/etc/security/audit_data
DESCRIPTION	<p>The audit_data file contains information about the audit daemon. The file contains the process ID of the audit daemon and the path name of the current audit log file. This is the format of the file:</p> <p style="text-align: center;"><i>pid:pathname</i></p> <p>where <i>pid</i> is the process ID for the audit daemon and <i>pathname</i> is the full path name for the current audit-log file.</p>
EXAMPLE	64:/etc/security/audit/server1/19930506081249.19930506230945.bongos
SUMMARY OF TRUSTED SOLARIS CHANGES	By default, the audit module is enabled on Trusted Solaris systems.
FILES	/etc/security/audit_data
SEE ALSO	audit(1MTSOL), auditd(1MTSOL), audit(2TSOL), audit.log(4TSOL) <i>Trusted Solaris Audit Administration Manual</i>
NOTES	This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See <i>Trusted Solaris Audit Administration Manual</i> for how to disable and enable auditing.

NAME	audit_event – File containing audit-event definition and class mapping
SYNOPSIS	<code>/etc/security/audit_event</code>
AVAILABILITY	SUNWcsu
DESCRIPTION	<p><code>/etc/security/audit_event</code> is an ASCII system file that stores event definitions and specifies the event-to-class mappings. Programs use the <code>getauevent(3TSOL)</code> routines to access this information.</p> <p>The fields for each event entry are separated by colons. Each event is separated from the next by a new line. Each entry in the <code>audit_event</code> file takes this form:</p> <p style="padding-left: 40px;"><i>number:name:description:flags</i></p> <p>The fields are defined as follows:</p> <p><i>number</i> The event number</p> <p><i>name</i> The event name</p> <p><i>description</i> The description of the event</p> <p><i>flags</i> Flags specifying classes to which the event is mapped</p>
EXAMPLES	<p>Here is a sample of the <code>audit_event</code> file entries:</p> <pre> 7:AUE_EXEC:exec(2):ps 79:AUE_OPEN_WTC:open(2) - write,creat,trunc:fc,fd,fw 6152:AUE_login:login - local:lo 6153:AUE_logout:logout:lo 6154:AUE_telnet:login - telnet:lo 6155:AUE_rlogin:login - rlogin:lo </pre>
SUMMARY OF TRUSTED SOLARIS CHANGES	Programs use the <code>getauevent(3TSOL)</code> routines rather than the <code>getauevent(3)</code> routines to access this information.
FILES	<code>/etc/security/audit_event</code>
SEE ALSO	<code>getauevent(3TSOL)</code> , <code>audit_control(4TSOL)</code> <i>Trusted Solaris Audit Administration Manual</i>
NOTES	This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See <i>Trusted Solaris Audit Administration Manual</i> for how to disable and enable auditing.

NAME	audit_user – File containing per-user auditing data						
SYNOPSIS	<code>/etc/security/audit_user</code>						
AVAILABILITY	SUNWcsu						
DESCRIPTION	<p>audit_user is an access-restricted ASCII system file that stores per-user auditing preselection data. Programs use the getauusernam(3TSOL) routines to access this information. The fields for each user entry are separated by colons. Each user is separated from the next by a new line. audit_user does not have general read permission. Each entry in the audit_user file takes this form:</p> <p style="padding-left: 40px;"><i>username:always-audit-flags:never-audit-flags</i></p> <p>The fields are defined as follows:</p> <table border="0" style="padding-left: 40px;"> <tr> <td><i>username</i></td> <td>The user's login name</td> </tr> <tr> <td><i>always-audit-flags</i></td> <td>Flags specifying event classes <i>always</i> to audit</td> </tr> <tr> <td><i>never-audit-flags</i></td> <td>Flags specifying event classes <i>never</i> to audit</td> </tr> </table>	<i>username</i>	The user's login name	<i>always-audit-flags</i>	Flags specifying event classes <i>always</i> to audit	<i>never-audit-flags</i>	Flags specifying event classes <i>never</i> to audit
<i>username</i>	The user's login name						
<i>always-audit-flags</i>	Flags specifying event classes <i>always</i> to audit						
<i>never-audit-flags</i>	Flags specifying event classes <i>never</i> to audit						
EXAMPLES	<p>Here is a sample audit_user file:</p> <pre style="padding-left: 40px;">other:lo,ad:io,cl fred:lo,ex,+fc,-fr,-fa:io,cl ethyl:lo,ex,nt:io,cl</pre>						
SUMMARY OF TRUSTED SOLARIS CHANGES	By default, the audit module is enabled on Trusted Solaris systems.						
FILES	<code>/etc/security/audit_user</code> <code>/etc/passwd</code>						
SEE ALSO	getauusernam (3TSOL), audit_control (4TSOL), passwd (4TSOL) <i>Trusted Solaris Audit Administration Manual</i>						
NOTES	This functionality is active only if the audit module has been enabled. By default, auditing is enabled in the Trusted Solaris environment. See <i>Trusted Solaris Audit Administration Manual</i> for how to disable and enable auditing.						

NAME	auth_desc – Descriptions of defined authorizations
SYNOPSIS	#include <tsol/auth.h>
AVAILABILITY	SUNWtsolu
DESCRIPTION	<p>Every defined authorization has a manifest constant for use in programs, a name for use in user interfaces, and a description displayed by certain administrative tools. A set of authorizations is assigned to an execution profile, and the execution profile is assigned to a user or role. The user or role to which the execution profile is assigned has the authority to perform the tasks allowed by the authorizations in the execution profile. Authorizations are interpreted by utility and desktop tools. The manifest constant, name, and description for each authorization defined on this system follow.</p> <p>Manifest Constant TSOL_AUTH_ENABLE_LOGIN</p> <p>Name enable logins</p> <p>Allows a user to enable logins on a machine that was just booted. Until logins are enabled there is no interactive use of the machine's resources.</p> <p>Manifest Constant TSOL_AUTH_REMOTE_LOGIN</p> <p>Name remote login</p> <p>Allows a user to remotely login, using programs such as TELNET, or FTP, in a way that requires entering identification and authentication information. Such a login is different from extending an existing login from one machine to another without re-authentication because the trusted path is not guaranteed for these methods.</p> <p>Manifest Constant TSOL_AUTH_TERMINAL_LOGIN</p> <p>Name terminal login</p> <p>Allows a user to login via a serial port. Such a login is different from extending an existing login from one machine to another without re-authentication because there is no trusted path presented for entering either the identification or authentication information.</p> <p>Manifest Constant TSOL_AUTH_FILE_AUDIT</p> <p>Name set/get file audit flags</p> <p>Allows a user to specify or view the auditing preselection information or public object flag to be associated with access to a file or directory. The auditing preselection information may override the preselection information associated with a user's access to a file or directory. The public object flag may override the successful read/search access preselection information associated with a user's access to a file or directory.</p> <p>Manifest Constant TSOL_AUTH_FILE_DOWNGRADE_SL</p>

Name downgrade file sensitivity label

Allows a user to specify the Sensitivity Label to set on a file that does not dominate the file's existing Sensitivity Label.

Manifest Constant TSOL_AUTH_FILE_UPGRADE_SL

Name upgrade file sensitivity label

Allows a user to specify the Sensitivity Label to set on a file that dominates the file's existing Sensitivity Label.

Manifest Constant TSOL_AUTH_FILE_OWNER

Name act as file owner

Allows a user to act as a file's owner. This includes the ability to change the permission bits and ACL, to downgrade the Sensitivity Label, to downgrade the Information Label, and set privileges (if further authorized by set file privileges) of files not owned. Also included is the ability to read and search directories, copy, move, and delete files not owned.

Manifest Constant TSOL_AUTH_FILE_CHOWN

Name change file owner

Allows a user to change the ownership and group of a file.

Manifest Constant TSOL_AUTH_FILE_SETPRIV

Name set file privileges

Allows a user to specify the allowed and forced privileges to be associated with the execution of a program file.

Manifest Constant TSOL_AUTH_ALLOCATE

Name allocate device

Allows a user to allocate a device and specify the CMW label to associate with information imported from it, or exported to it.

Manifest Constant TSOL_AUTH_CONFIG_DEVICE

Name configure device attributes

Allows an administrator to configure a device. Device configuration includes such things as setting the device name, type, label range, allocatable status, and allocation authorization list.

Manifest Constant TSOL_AUTH_REVOKE_DEVICE

Name	revoke or reclaim device
Allows an administrator to deallocate a currently allocated device or reset the allocate error state to make a device allocatable again.	
Manifest Constant	TSOL_AUTH_WIN_DOWNGRADE_SL
Name	paste to a downgraded window
Allows a user to paste selected information to a window whose Sensitivity Label does not dominate the selected information's Sensitivity Label.	
Manifest Constant	TSOL_AUTH_WIN_UPGRADE_SL
Name	paste to an upgraded window
Allows a user to paste selected information to a window whose Sensitivity Label dominates the the selected information's Sensitivity Label.	
Manifest Constant	TSOL_AUTH_SYS_ACCRED_SET
Name	use all defined labels
Allows a user to use all the available labels on the system rather than to be restricted to just the labels in the label encodings defined user accreditation range. Using a label implies the ability to specify that label for any of the label building interfaces which include those to re-label files and create workspaces.	
Manifest Constant	TSOL_AUTH_BYPASS_FILE_VIEW
Name	bypass file view
Allows a user to drag and drop a file without viewing that files contents.	
Manifest Constant	TSOL_AUTH_SHUTDOWN
Name	shutdown the system
Allows a user to shutdown the system via the Trusted Path menu. When the system is a CDE X Terminal, the CDE X Terminal is shutdown not the server. Unless the "abort_enable" system variable (see /etc/system) is set to 0, this authorization can be bypassed by entering the keyboard abort sequence, gaining entry to the PROM, and rebooting from the PROM.	
Manifest Constant	TSOL_AUTH_USER_IDENT
Name	set user identity

Allows an administrator to set the security information related to the user's identity. The user name, primary group, secondary groups, comment, and login shell may all be set via the User Manager. This authorization is needed to add, copy, or delete a user.

Manifest Constant TSOL_AUTH_USER_PASSWORD

Name set user password

Allows an administrator to set password information pertaining to a user. The password, type of password, life time, expiration date, warning days, and the permission to set up the credentials table may all be set via the User Manager.

Manifest Constant TSOL_AUTH_USER_SELF

Name permit self-modification

Allows an administrator to modify his or her own user attributes.

Manifest Constant TSOL_AUTH_USER_LABELS

Name set user labels

Allows an administrator to set various label-related pieces information associated with a particular user. A user's minimum login label, clearance, label view, and label translation attributes may be set via the User Manager.

Manifest Constant TSOL_AUTH_USER_AUDIT

Name set user audit flags

Allows an administrator to set the per user audit flags.

Manifest Constant TSOL_AUTH_USER_PROFILES

Name set user profiles

Allows an administrator to assign profiles to a user.

Manifest Constant TSOL_AUTH_USER_IDLE

Name set idle time

Allows an administrator to set the idle time and determine which action to take when a workstation has been idle for too long. The idle time and idle command can be set via the User Manager.

Manifest Constant TSOL_AUTH_USER_ROLES

Name set roles list

Allows an administrator to select which roles a user may assume. When a user assumes a role he or she may use all commands and actions granted to that role.

Manifest Constant	TSOL_AUTH_USER_HOME
Name	set home directory attributes
Allows an administrator to determine such things as location, permissions, and initial contents of a user's home directory.	
Manifest Constant	TSOL_AUTH_PRINT_ADMIN
Name	administer printing
Allows a user to perform Trusted Printing System administration. Allows a user to start and stop printing daemons, list and cancel other users' print jobs, etc.	
Manifest Constant	TSOL_AUTH_PRINT_NOBANNER
Name	print without banners
Allows a user to submit a print request to the Trusted Printing System that specifies (by means of the 'lp -o nobanner' option) that the print job's banner and trailer pages should be suppressed.	
Manifest Constant	TSOL_AUTH_PRINT_POSTSCRIPT
Name	print a PostScript file
Allows a user to print a PostScript file with the Trusted Printing System.	
Manifest Constant	TSOL_AUTH_PRINT_UNLABELED
Name	print without labels
Allows a user to submit a print request to the Trusted Printing System that specifies (by means of the 'lp -o nolabels' option) that the body pages of the print job should have the top and bottom labels suppressed.	
Manifest Constant	TSOL_AUTH_DB_ALIASES
Name	modify aliases
Allows a user to edit the aliases databases via the Database Manager.	
Manifest Constant	TSOL_AUTH_DB_AUTO_HOME
Name	modify auto_home
Allows a user to edit the auto_home databases via the Database Manager.	
Manifest Constant	TSOL_AUTH_DB_BOOTPARAMS
Name	modify bootparams

Allows a user to edit the bootparams databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_ETHERS

Name modify ethers

Allows a user to edit the ethers databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_GROUP

Name modify group

Allows a user to edit the group databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_HOSTS

Name modify hosts

Allows a user to edit the hosts databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_LOCALE

Name modify locale

Allows a user to edit the locale databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_NETGROUP

Name modify netgroup

Allows a user to edit the netgroup databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_NETMASKS

Name modify netmasks

Allows a user to edit the netmasks databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_NETWORKS

Name modify networks

Allows a user to edit the networks databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_PASSWD

Name modify password

Allows a user to edit the password databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_PROTOCOLS

Name modify protocols

Allows a user to edit the protocols databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_RPC

Name modify rpc

Allows a user to edit the rpc databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_SERVICES

Name modify services

Allows a user to edit the services databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_TIMEZONE

Name modify timezone

Allows a user to edit the timezone databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_TNIDB

Name modify tnidb

Allows a user to edit the tnidb databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_TNRHDB

Name modify tnrhdb

Allows a user to edit the tnrhdb databases via the Database Manager.

Manifest Constant TSOL_AUTH_DB_TNRHTP

Name modify tnrhdp

Allows a user to edit the tnrhdp databases via the Database Manager.

Manifest Constant TSOL_AUTH_CRON_ADMIN

Name modify cron admin

Allows a user to modify and list crontab files of role users and users named in /etc/cron.d/cron.admin.

Manifest Constant TSOL_AUTH_CRON_USER

Name modify cron users

Allows a user to modify and list crontab files of non-administrative users.

Manifest Constant TSOL_AUTH_AT_ADMIN

Name modify at admin

Allows a user to remove and list at jobs of role users and users named in /etc/cron.d/at.admin.

Manifest Constant TSOL_AUTH_AT_USER

Name modify at users

Allows a user to remove and list at jobs of non-administrative users.

FILES /usr/lib/tsol/locale/locale/auth_name Authorizations descriptions
/usr/include/tsol/auth_names.h Manifest constant and ID value definitions

SEE ALSO Intro(3TSOL), auth_to_str(3TSOL), chkauth(3TSOL), auth_name(4TSOL)

Trusted Solaris administrator's document set

Trusted Solaris Developer's Guide

NAME	auth_name – authorization description database
SYNOPSIS	<code>/usr/lib/tsol/locale/locale/auth_name</code>
AVAILABILITY	SUNWtsolu
DESCRIPTION	<p>The auth_name database defines the localized authorization names and descriptions defined on this system. This database is used along with <code><tsol/auth_names.h></code> by auth_to_str(3TSOL), str_to_auth(3TSOL), and get_auth_text(3TSOL) to translate between authorization ID, authorization name string, and description.</p> <p>Each entry in the auth_name database consists of one line with fields separated by colons (:). A line ending with a backslash (\) indicates continuation of the entry on the next line. Lines beginning with a # character are treated as comments. Each entry has the form:</p> <p style="padding-left: 40px;"><i>constant:name:description</i></p> <p>The entry fields are:</p> <p><i>constant</i> The <i>constant</i> field must be identical to the manifest constant defined for the authorization in <code><tsol/auth_names.h></code>, where a unique authorization ID is assigned to each authorization constant.</p> <p><i>name</i> The external name of the authorization. It is returned by auth_to_str and is used by str_to_auth. Authorization names are concise and descriptive so that they may be used in various GUIs. The authorization name can be customized and localized.</p> <p><i>description</i> The description of the activity permitted by the authorization. The description name can be customized and localized.</p>
EXAMPLES	<pre># # Example entries in /usr/lib/tsol/locale/C/auth_name # TSOL_AUTH_ENABLE_LOGIN:enable logins:Allows a user to enable logins on a \ machine that was just booted. Until logins are enabled there is \ no interactive use of the machine's resources. TSOL_AUTH_REMOTE_LOGIN:remote login:Allows a user to remotely login, using \ programs such as TELNET, or FTP, in a way that requires entering \ identification and authentication information. Such a login is \ different from extending an existing login from one machine to \ another without re-authentication because the trusted path is not \ guaranteed for these methods.</pre>
SEE ALSO	auth_to_str (3TSOL), chkauth (3TSOL), auth_desc (4TSOL)

NAME	config.privs – List of window privileges that override system checks
SYNOPSIS	/usr/openwin/server/tsol/config.privs
AVAILABILITY	SUNWxwplt
DESCRIPTION	<p>config.privs contains a list of all window privileges. config.privs lists each privilege in plain text, one per line, separated from the next by a new line. Lines preceded by a comment sign (#) are ignored.</p> <p>Each privilege not preceded by a comment overrides system checks for that privilege. The security administrator can comment out privileges in the list, but cannot add new privileges.</p> <p>By default, config.privs contains all the privileges that are allowed in the file: win_colormap, win_config, win_dga, win_devices, win_fontpath.</p> <p>config.privs should have a sensitivity label of ADMIN_LOW with permission bits 664, owner root, and group bin.</p>
FILES	/usr/openwin/server/tsol/config.privs

NAME	config_data – Trusted Solaris configuration data for network and Jumpstart installations
SYNOPSIS	config_data
DESCRIPTION	<p>The config_data file is used to automatically configure Trusted Solaris user-configurable features during network and Jumpstart installations. Features that can be configured in the config_data file are equivalent to Trusted Solaris features configurable through the Label Configuration dialog during system installation. Each entry in the file has the form:</p> <p style="padding-left: 40px;"><i>feature = configuration_value</i></p> <p>Allowed values for <i>feature</i> are:</p> <p>multiple_user_sl When enabled, allows the user to use multiple sensitivity labels.</p> <p>hide_upgraded_names When enabled, allows an authorized user to create or relabel files and sub-directories at a sensitivity label that strictly dominates the SL of the containing directory.</p> <p>enable_il When enabled, causes information labels to be displayed.</p> <p>enable_il_floating When enabled along with enable_il, causes information labels to float as a result of certain user operations.</p> <p>reset_il_on_exec When enabled along with enable_il, causes the IL of an executed program to be reset to ADMIN_LOW at the start of execution. When disabled along with an enabled enable_il, causes the IL of an executed program to be set to the conjunction of the IL of the the calling process, the IL on the program file being executed, and the IL of any other shared libraries that are read.</p> <p>Allowed values for <i>configuration_value</i> are:</p> <p>yes Enable the feature. no Disable the feature.</p>
EXAMPLES	<pre>multiple_user_sl = yes hide_upgraded_names = no enable_il = yes enable_il_floating = no reset_il_on_exec = yes</pre>
SEE ALSO	<p><i>Trusted Solaris administrator's document set</i> <i>Trusted Solaris Administrator's Procedures</i></p>

NAME	device_allocate – device_allocate file														
SYNOPSIS	/etc/security/device_allocate														
DESCRIPTION	<p>The device_allocate file contains mandatory access control information about each physical device. Each device is represented by a one line entry of the form:</p> <pre><i>device-name ; device-type ; device-minimum ; device-maximum ; device-authorization ; device-clean ; comment</i></pre> <p>where</p> <table border="0"> <tr> <td style="padding-right: 20px;"><i>device-name</i></td> <td>This is an arbitrary text string naming the physical device. This field contains no embedded white space or non-printable characters.</td> </tr> <tr> <td><i>device-type</i></td> <td>This is an arbitrary text string naming the generic device type. This field identifies and groups together devices of like type. This field contains no embedded white space or non-printable characters.</td> </tr> <tr> <td><i>device-minimum</i></td> <td>This is the minimum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.</td> </tr> <tr> <td><i>device-maximum</i></td> <td>This is the maximum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.</td> </tr> <tr> <td><i>device-authorization</i></td> <td>This is a comma-separated list of authorizations required to allocate the device, or an '*' to indicate that the device is not allocatable, or an '@' to indicate that no explicit authorization is needed to allocate the device.</td> </tr> <tr> <td><i>device-clean</i></td> <td>This is the physical device's data purge program to be run any time the device is acted on by allocate(1MVSOL). This is to ensure that all usable data is purged from the physical device before it is reused. This field contains the filename of a program in /etc/security/lib.</td> </tr> <tr> <td><i>comment</i></td> <td>This field contains text that describes the device.</td> </tr> </table> <p>The device_allocate file is a text file that resides in the /etc/security directory. The device_allocate file should not be edited by hand. The designated administrative role should use the Add Allocatable Device action to add a device and use the Device Allocation Manager Configure dialog for modifications to a device, because these tools preserve the desired file permissions, owner, group, and label, and audit all changes.</p> <p>Lines in device_allocate can end with a '\ ' to continue an entry on the next line.</p> <p>Comments may also be included. A '#' makes a comment of all further text until the next NEWLINE not immediately preceded by a '\ '.</p>	<i>device-name</i>	This is an arbitrary text string naming the physical device. This field contains no embedded white space or non-printable characters.	<i>device-type</i>	This is an arbitrary text string naming the generic device type. This field identifies and groups together devices of like type. This field contains no embedded white space or non-printable characters.	<i>device-minimum</i>	This is the minimum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.	<i>device-maximum</i>	This is the maximum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.	<i>device-authorization</i>	This is a comma-separated list of authorizations required to allocate the device, or an '*' to indicate that the device is not allocatable, or an '@' to indicate that no explicit authorization is needed to allocate the device.	<i>device-clean</i>	This is the physical device's data purge program to be run any time the device is acted on by allocate (1MVSOL). This is to ensure that all usable data is purged from the physical device before it is reused. This field contains the filename of a program in /etc/security/lib .	<i>comment</i>	This field contains text that describes the device.
<i>device-name</i>	This is an arbitrary text string naming the physical device. This field contains no embedded white space or non-printable characters.														
<i>device-type</i>	This is an arbitrary text string naming the generic device type. This field identifies and groups together devices of like type. This field contains no embedded white space or non-printable characters.														
<i>device-minimum</i>	This is the minimum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.														
<i>device-maximum</i>	This is the maximum sensitivity label allowed for the device special files associated with the physical device. This field is an HEX label.														
<i>device-authorization</i>	This is a comma-separated list of authorizations required to allocate the device, or an '*' to indicate that the device is not allocatable, or an '@' to indicate that no explicit authorization is needed to allocate the device.														
<i>device-clean</i>	This is the physical device's data purge program to be run any time the device is acted on by allocate (1MVSOL). This is to ensure that all usable data is purged from the physical device before it is reused. This field contains the filename of a program in /etc/security/lib .														
<i>comment</i>	This field contains text that describes the device.														

NAME	device_deallocate – device_deallocate file
SYNOPSIS	<i>/etc/security/tsol/device_deallocate</i>
DESCRIPTION	<p>The device_deallocate file contains device-deallocation information about each physical device. Each device is represented by an entry of the form:</p> <pre style="margin-left: 40px;"><i>device-name ; system-boot ; user-logout ; forced-logout ;</i></pre> <p>A backslash (\) at the end of a line continues the next line as part of the current entry. A pound sign (#) as the first character of a line indicates a comment line, which is ignored. Leading and trailing blanks are allowed in any of the fields.</p> <p><i>device-name</i> The name of the device.</p> <p><i>system-boot</i> Specifies what to do when the named device is found during system boot in an allocated state. This field may be one of these keywords:</p> <pre style="margin-left: 40px;">FORCED_DEALLOCATE Deallocate the device. NO_ACTION Leave the device in the allocated state.</pre> <p><i>user-logout</i> Specifies what to do when the named device is allocated to a user who is logging out. This field may be one of these keywords:</p> <pre style="margin-left: 40px;">ASK_USER Ask the user whether to deallocate the device. FORCED_DEALLOCATE Deallocate the device. NO_ACTION Leave the device in the allocated state.</pre> <p><i>forced-logout</i> Specifies what to do when the named device is allocated to a user who is being forced to log out. This field may be one of these keywords:</p> <pre style="margin-left: 40px;">FORCED_DEALLOCATE Deallocate the device. NO_ACTION Leave the device in the allocated state.</pre> <p>device_allocate should be at a sensitivity label of ADMIN_LOW with permission bits 644, owner root, and group sys.</p> <p>Users may specify their own device-deallocation options locally in their home directories in a file named .device_deallocate. The local file has the same format, and its entries take precedence; however, a local file's <i>system-boot</i> field is ignored and has no effect.</p>
EXAMPLE	<pre>st0;\ FORCED_DEALLOCATE;\ NO_ACTION;\ NO_ACTION;\ # scsi tape</pre>
NOTES	The device_deallocate file is not read by deallocate (1MTSOL). Rather, it is read by a <i>GUI</i> or a script program at the events of system boot, user logout, and forced logout. The program in term invokes deallocate command when actual device deallocation is required.

FILES /etc/security/tsol/device_deallocate, \$HOME/.device_deallocate

SEE ALSO add_allocatable(1MTSOL), allocate(1MTSOL), deallocate(1MTSOL),
list_devices(1MTSOL), rem_allocatable(1MTSOL)

NAME	device_maps – device_maps file
SYNOPSIS	<code>/etc/security/tsol/device_maps</code>
DESCRIPTION	<p>The device_maps file contains access control information about each physical device. Each device is represented by an entry of the form:</p> <p style="padding-left: 40px;"><i>device-name</i> : <i>device-type</i> : <i>device-list</i> :</p> <p>A backslash (\) at the end of a line continues the next line as part of the current entry. A pound sign (#) as the first character of a line indicates a comment line, which is ignored. Leading and trailing blanks are allowed in any of the fields.</p> <p><i>device-name</i> An arbitrary ASCII string that names the physical device. Spaces and nonprintable characters are not allowed.</p> <p><i>device-type</i> An arbitrary ASCII string that names the generic device type. This field identifies and groups together devices of like types. Spaces and nonprintable characters are not allowed.</p> <p><i>device-list</i> A list of the device special files associated with the physical device. This field contains valid device special file pathnames separated by spaces.</p> <p>The device_maps file must be created by the system administrator before device allocation is enabled.</p> <p>device_maps should be at a sensitivity label of ADMIN_LOW with permission bits 644, owner root, and group sys.</p>
EXAMPLES	<pre> # scsi tape st1:\ rmt:\ /dev/rst21 /dev/nrst21 /dev/rst5 /dev/nrst5 /dev/rst13 \ /dev/nrst13 /dev/rst29 /dev/nrst29 /dev/rmt/1l /dev/rmt/1m \ /dev/rmt/1 /dev/rmt/1h /dev/rmt/1u /dev/rmt/1ln /dev/rmt/1mn \ /dev/rmt/1n /dev/rmt/1hn /dev/rmt/1un /dev/rmt/1b /dev/rmt/1bn:\ </pre>
FILES	<code>/etc/security/tsol/device_maps</code>
SEE ALSO	allocate (1MTSOL), deallocate (1MTSOL), dminfo (1MTSOL), list_devices (1MTSOL),

NAME	device_policy – device policy file								
AVAILABILITY	SUNWtsol								
DESCRIPTION	<p>The security policy for device files can differ from that for regular files and is configured through the device_policy database file. Rebooting the system in multiuser mode is required to effect the file's contents,</p> <p>Each entry in the file consists of one or more lines and represents the device policy configuration for one or more device files. A backslash (\) at the end of a line continues the next line as part of the current entry. A pound sign (#) as the first character of a line indicates a comment line, which is ignored. Each entry is of the form:</p> <pre style="margin-left: 40px;"><i>name:minor_name policy_type=value policy_type=value ...</i></pre> <p>name is the name of a device driver.</p> <p>minor_name is the actual name of a minor node, or a string of shell metacharacters that represent several minor nodes [see sh(1)].</p> <p>If two or more entries match a device, devpolicy(1MTSOL) uses the first matching entry. For example, for the following device_policy entries, the policy for /dev/ptyp0 will differ from the policy for other pty devices.</p> <pre style="margin-left: 40px;"># # device_policy file # ptc: typ0 data_mac_policy=DR_MAC_EQ,DW_MAC_EQ # ptc:* data_mac_policy=DR_MAC_ANY,DW_MAC_ANY</pre> <p>policy_type=value specifies a policy for the device nodes. There are five policy types: il_float_policy, data_mac_policy, attr_mac_policy, open_priv, and str_type. The policy types and their allowed values are described below.</p> <p>il_float_policy type</p> <p>This policy type specifies whether to float the device's information label. The value for this type is a comma-separated pair of values: a read-IL value and a write-IL value:</p> <p>The read-IL values are:</p> <table border="0" style="margin-left: 40px;"> <tr> <td>DR_IL_NOFLOAT</td> <td>do not float</td> </tr> <tr> <td>DR_IL_FLOAT</td> <td>float to the reading process' IL</td> </tr> </table> <p>The write-IL values are:</p> <table border="0" style="margin-left: 40px;"> <tr> <td>DW_IL_NOFLOAT</td> <td>do not float</td> </tr> <tr> <td>DW_IL_FLOAT</td> <td>float to the device's IL</td> </tr> </table> <p>The default policy is:</p> <pre style="margin-left: 40px;">il_float_policy=DR_IL_FLOAT,DW_IL_FLOAT</pre>	DR_IL_NOFLOAT	do not float	DR_IL_FLOAT	float to the reading process' IL	DW_IL_NOFLOAT	do not float	DW_IL_FLOAT	float to the device's IL
DR_IL_NOFLOAT	do not float								
DR_IL_FLOAT	float to the reading process' IL								
DW_IL_NOFLOAT	do not float								
DW_IL_FLOAT	float to the device's IL								

data_mac_policy type

This policy type specifies what a process' sensitivity label must be to have access to the device. The specified policy is enforced by the **open(2TSOL)** and **access(2TSOL)** system calls. The value for this type is a comma-separated pair of values: a read-MAC value and a write-MAC value:

The read-MAC values are:

DR_MAC_ANY	Process may have any SL.
DR_MAC_EQ	Process SL must be equal to device SL.
DR_MAC_NEQ	Process SL must not equal device SL.
DR_MAC_NEVER	Device is not read accessible.
DR_MAC_SDOM	Process SL must dominate device SL.
DR_MAC_ODOM	Process SL must be dominated by device SL.

The write-MAC values are:

DW_MAC_ANY	Process may have any SL.
DW_MAC_EQ	Process SL must equal device SL.
DW_MAC_NEQ	Process SL must not equal device SL.
DW_MAC_NEVER	Device is not write accessible.
DW_MAC_SDOM	Process SL must dominate device SL.
DW_MAC_ODOM	Process SL must be dominated by device SL.

The optional read-MAC-modifier or write-MAC-modifier value is:

MOD_AUTO_ALLOC	Automatically allocate the device on behalf of the opening process.
-----------------------	---

The default policy is

`data_mac_policy=DR_MAC_EQ,DW_MAC_EQ`

attr_mac_policy type

This policy type specifies how to handle access to the device's attributes by the operations **acl(2)**, **chmod(2TSOL)**, **chown(2TSOL)**, and **stat(2TSOL)**. The value for this type is a comma-separated set of values: a read-MAC value, a write-MAC value, and an optional read-MAC-modifier:

The read-MAC values are:

DR_MAC_ANY	Process may have any SL.
DR_MAC_EQ	Process SL must equal device SL.
DR_MAC_NEQ	Process SL must not equal device SL.
DR_MAC_NEVER	Device is not read accessible.
DR_MAC_SDOM	Process SL must dominate device SL.
DR_MAC_ODOM	Process SL must be dominated by device SL.

The write-MAC values are:

DW_MAC_ANY	Process may have any SL.
-------------------	--------------------------

	DW_MAC_EQ	Process SL must equal device SL.
	DW_MAC_NEQ	Process SL must not equal device SL.
	DW_MAC_NEVER	Device is not write accessible.
	DW_MAC_SDOM	Process SL must dominate device SL.
	DW_MAC_ODOM	Process SL must be dominated by device SL.
	The optional read-MAC-modifier value is:	
	MOD_FABRICATE	Return fabricated device attributes to the reading process. Fabrication is designed for a process that walks down an array of BSD-style pty's until it encounters an accessible pty (indicated by getting device attributes) or the end of the array.
	The default policy is:	
	<code>attr_mac_policy=DR_MAC_SDOM,DW_MAC_EQ</code>	
open_priv type	This policy type specifies a privilege required to open the device. The specified privilege is required in addition to the data MAC policy. Privilege names can be in upper or lower case; or an integer ordinal can be used. For example,	
	<code>open_priv=sys_devices</code>	
	The default policy is:	
	<code>open_priv=none</code>	
str_type type	The streams type, meaningful only for streams devices, specifies how the kernel streams head should control streams messages. The value can be one of these keywords:	
	DSTR_LOOP	Loop type stream. Unlabeled streams control messages are allowed. Unlabeled data messages are not allowed.
	DSTR_NET	Network type Stream. Unlabeled Stream messages are not allowed.
	DSTR_DEV	Device type Stream. Unlabeled Stream messages are allowed.
	An example is:	
	<code>str_type=DSTR_NET</code>	
	The default policy is:	
	<code>str_type=STR_DEV</code>	
EXAMPLES	<pre>mm:kmem \ data_mac_policy=DR_MAC_EQ,DW_MAC_EQ \ attr_mac_policy=DR_MAC_SDOM,DW_MAC_EQ \ il_float_policy=DR_IL_FLOAT,DW_IL_NOFLOAT</pre>	

```
mm:null \
data_mac_policy=DR_MAC_ANY,DW_MAC_ANY \
attr_mac_policy=DR_MAC_SDOM,DW_MAC_EQ \
il_float_policy=DR_IL_NOFLOAT,DW_IL_NOFLOAT
```

FILES /etc/security/tsol/device_policy

SEE ALSO devpolicy(1MTSOL)

NAME	inetd.conf – Internet servers database										
SYNOPSIS	<i>/etc/inet/inetd.conf</i> <i>/etc/inetd.conf</i>										
DESCRIPTION	<p>The inetd.conf file contains the list of servers that inetd(1M) invokes when it receives an Internet request over a socket. Each server entry is composed of a single line of this form:</p> <p><i>service-name endpoint-type protocol wait-status uid server-program server-arguments</i></p> <p>Fields are separated by either SPACE or TAB characters. A pound sign (#) indicates the beginning of a comment; characters up to the end of the line are not interpreted by routines that search this file.</p> <p><i>service-name</i> The name of a valid service listed in the services file. For RPC services, the value of the <i>service-name</i> field consists of the RPC service name or program number, followed by a slash (/) and either a version number or a range of version numbers (for example, rstatd/2-4).</p> <p><i>endpoint-type</i> Can be one of these:</p> <table border="0" style="margin-left: 2em;"> <tr> <td>stream</td> <td>For a stream socket</td> </tr> <tr> <td>dgram</td> <td>For a datagram socket</td> </tr> <tr> <td>raw</td> <td>For a raw socket</td> </tr> <tr> <td>seqpacket</td> <td>For a sequenced packet socket</td> </tr> <tr> <td>tli</td> <td>For all tli endpoints</td> </tr> </table> <p><i>protocol</i> Must be a recognized protocol listed in the file /etc/inet/protocols. For RPC services, the field consists of the string rpc followed by a slash (/) and either an asterisk (*), one or more nettypes, one or more netids, or a combination of nettypes and netids. Whatever the value, it is first treated as a nettype. If the value is not a valid nettype, then it is treated as a netid. For example, rpc/* for an RPC service using all the transports supported by the system (the list can be found in the /etc/netconfig file), is equivalent to saying rpc/visible rpc/ticots for an RPC service using the Connection-Oriented Transport Service.</p> <p><i>wait-status</i> nowait for all but “single-threaded” datagram servers—servers that do not release the socket until a timeout occurs. These must have the status wait. Do not configure udp services as nowait. Such a configuration would cause a race condition in which the inetd program selects on the socket and the server program reads from the socket. Many server programs would be forked and performance would be severely compromised.</p> <p>A new option exists for udp servers. The poly option, is similar to the wait option except that poly allows a separate server to be started at each sensitivity label. This option is allowed only for udp servers.</p>	stream	For a stream socket	dgram	For a datagram socket	raw	For a raw socket	seqpacket	For a sequenced packet socket	tli	For all tli endpoints
stream	For a stream socket										
dgram	For a datagram socket										
raw	For a raw socket										
seqpacket	For a sequenced packet socket										
tli	For all tli endpoints										

If the server program should inherit the trusted path attribute, the *wait-status* field should include the keyword **trusted**, separated from other keywords in the field by a comma. If the keyword is not present, the trusted path attribute will not be propagated to the server.

If the server program should inherit audit characteristics from the client, the *wait-status* field should include the keyword **setaudit**, separated from other keywords in the field by a comma. If the **setaudit** keyword is present, the audit ID, audit terminal ID, and audit preselection mask of the client will be transferred to the server.

<i>uid</i>	The user ID under which the server should run allows servers to run with access privileges other than those for root. If the server should run with the ID of the client making the call to the server, a keyword of CLIENT should be entered in the <i>uid</i> field. The CLIENT keyword is allowed only for nowait servers. If the CLIENT keyword is present the user ID, group ID, and supplementary groups of the client will be transferred to the server.
<i>server-program</i>	Either the pathname of a server program to be invoked by inetd to perform the requested service, or the value internal if inetd itself provides the service.
<i>server-arguments</i>	If a server must be invoked with command-line arguments, the entire command line (including argument 0) must appear in this field (which consists of all remaining words in the entry). If the server expects inetd to pass it the address of its peer (for compatibility with 4.2BSD executable daemons), then “%A~” should be specified as the first argument to the command. No more than five arguments are allowed in this field.

SUMMARY OF TRUSTED SOLARIS CHANGES

The *wait-status* field is extended to allow a **trusted** keyword to specify that the trusted path attribute should be passed to the server by **inetd**. If you want a server to run with the audit characteristics of the client, the *wait-status* field can now contain a keyword of **setaudit**.

If you want a **nowait** server to run with the user ID of the client, the *uid* field can now contain a keyword of **CLIENT**.

The **poly** option has been added for **udp** servers. The option is similar to the **wait** option except that **poly** allows a separate server to be started at each sensitivity label.

FILES	/etc/netconfig	Network configuration file
	/etc/inet/protocols	Internet protocols
	/etc/inet/services	Internet network services

SEE ALSO **rlogin**(1TSOL), **rsh**(1TSOL), **in.tftpd**(1MTSOL), **inetd**(1MTSOL), **services**(4)

NOTES

/etc/inet/inetd.conf is the official SVR4 name of the **inetd.conf** file. The symbolic link **/etc/inetd.conf** exists for BSD compatibility.

NAME	inittab – script for init
DESCRIPTION	<p>The file <code>/etc/inittab</code> controls process dispatching by init. The processes most typically dispatched by init are daemons.</p> <p>The inittab file is composed of entries that are position dependent and have the following format:</p> <p style="text-align: center;"><i>id:rstate:action:process</i></p> <p>Each entry is delimited by a newline; however, a backslash (\) preceding a newline indicates a continuation of the entry. Up to 512 characters for each entry are permitted. Comments may be inserted in the <i>process</i> field using the convention for comments described in sysh(1TSOL). There are no limits (other than maximum entry size) imposed on the number of entries in the inittab file. The entry fields are:</p> <p><i>id</i> One or two characters used to uniquely identify an entry.</p> <p><i>rstate</i> Define the run level in which this entry is to be processed. Run-levels effectively correspond to a configuration of processes in the system. That is, each process spawned by init is assigned a run level(s) in which it is allowed to exist. The run levels are represented by a number ranging from 0 through 6. For example, if the system is in run level 1, only those entries having a 1 in the <i>rstate</i> field are processed.</p> <p>When init is requested to change run levels, all processes that do not have an entry in the <i>rstate</i> field for the target run level are sent the warning signal SIGTERM and allowed a 5-second grace period before being forcibly terminated by the kill signal SIGKILL. The <i>rstate</i> field can define multiple run levels for a process by selecting more than one run level in any combination from 0 through 6. If no run level is specified, then the process is assumed to be valid at all run levels 0 through 6.</p> <p>There are three other values, a, b and c, which can appear in the <i>rstate</i> field, even though they are not true run levels. Entries which have these characters in the <i>rstate</i> field are processed only when an init or telinit process requests them to be run (regardless of the current run level of the system). See init(1M). These differ from run levels in that init can never enter run level a, b or c. Also, a request for the execution of any of these processes does not change the current run level. Furthermore, a process started by an a, b or c command is not killed when init changes levels. They are killed only if their line in inittab is marked off in the <i>action</i> field, their line is deleted entirely from inittab, or init goes into single-user state.</p> <p><i>action</i> Key words in this field tell init how to treat the process specified in the <i>process</i> field. The actions recognized by init are as follows:</p> <p style="padding-left: 20px;">respawn If the process does not exist, then start the process; do not wait for its termination (continue scanning the inittab file), and when the process dies, restart the process. If the process currently exists, do nothing and continue scanning the inittab file.</p>

- wait** When **init** enters the run level that matches the entry's *rstate*, start the process and wait for its termination. All subsequent reads of the **inittab** file while **init** is in the same run level cause **init** to ignore this entry.
- once** When **init** enters a run level that matches the entry's *rstate*, start the process, do not wait for its termination. When it dies, do not restart the process. If **init** enters a new run level and the process is still running from a previous run level change, the program is not restarted.
- boot** The entry is to be processed only at **init**'s boot-time read of the **inittab** file. **init** is to start the process and not wait for its termination; when it dies, it does not restart the process. In order for this instruction to be meaningful, the *rstate* should be the default or it must match **init**'s run level at boot time. This action is useful for an initialization function following a hardware reboot of the system.
- bootwait** The entry is to be processed the first time **init** goes from single-user to multi-user state after the system is booted. (If **initdefault** is set to **2**, the process runs right after the boot.) **init** starts the process, waits for its termination and, when it dies, does not restart the process.
- powerfail** Execute the process associated with this entry only when **init** receives a power fail signal, **SIGPWR** (see **signal(3C)**).
- powerwait** Execute the process associated with this entry only when **init** receives a power fail signal, **SIGPWR**, and wait until it terminates before continuing any processing of **inittab**.
- off** If the process associated with this entry is currently running, send the warning signal **SIGTERM** and wait 5 seconds before forcibly terminating the process with the kill signal **SIGKILL**. If the process is nonexistent, ignore the entry.
- ondemand** This instruction is really a synonym for the **respawn** action. It is functionally identical to **respawn** but is given a different keyword in order to divorce its association with run levels. This instruction is used only with the **a**, **b** or **c** values described in the *rstate* field.
- initdefault** An entry with this action is scanned only when **init** is initially invoked. **init** uses this entry to determine which run level to enter initially. It does this by taking the highest run level specified in the *rstate* field and using that as its initial state. If the *rstate* field is empty, this is interpreted as **0123456** and **init** will enter run level **6**. This will cause the system to loop (it will go to firmware and reboot continuously). Additionally, if **init** does not find an **initdefault** entry in **inittab**, it requests an initial run level from the user at reboot time.

sysinit Entries of this type are executed before **init** tries to access the console (that is, before the **Console Login:** prompt). It is expected that this entry will be used only to initialize devices that **init** might try to ask the run level question. These entries are executed and **init** waits for their completion before continuing.

process Specify a command to be executed. The entire **process** field is prefixed with **exec** and passed to a forked **sysh** as **sysh -c 'exec command'**. For this reason, any legal **sysh** syntax can appear in the *process* field.

**SUMMARY OF
TRUSTED
SOLARIS
CHANGES**

Trusted Solaris uses **sysh** instead of **sh**.

SEE ALSO

sysh(1TSOL), **who(1)**, **init(1M)**, **ttymon(1M)**, **exec(2TSOL)**, **open(2TSOL)**, **signal(3C)**

NAME	label_encodings – label encodings file
SYNOPSIS	<code>/etc/security/tsol/label_encodings</code>
DESCRIPTION	<p>In addition to the required sections of the label encodings file described in <i>Compartmented Mode Workstation Labeling: Encodings Format</i>, Trusted Solaris accepts optional local extensions. These extensions provide various translation options and an association between ASCII color names and sensitivity labels.</p> <p>The optional local extensions section starts with the LOCAL DEFINITIONS: keyword and is followed by zero or more of the following unordered statements:</p> <p>ADMIN LOW NAME= name The string <i>name</i> is accepted as an alternate name for the <i>Admin Low</i> label when translating from ASCII coded to binary form. The string <i>name</i> is the string returned when translating the ADMIN_LOW label from binary to ASCII-coded form. If this option is not specified, ADMIN_LOW is used.</p> <p>Note that use of this option could lead to interoperability problems with machines which do not have the same alternate name.</p> <p>ADMIN HIGH NAME= name The string <i>name</i> is accepted as an alternate name for the ADMIN_HIGH label when translating from ASCII-coded form to binary form. The string <i>name</i> is the string returned when translating the ADMIN_HIGH label from binary to ASCII-coded form. If this option is not specified, ADMIN_HIGH is used.</p> <p>Note that use of this option could lead to interoperability problems with machines which do not have the same alternate name.</p> <p>DEFAULT LABEL VIEW IS EXTERNAL Unless otherwise specified, when an ADMIN_HIGH or ADMIN_LOW binary label is translated to an ASCII-coded label, the ASCII-coded label will be in external form. In external form ADMIN_HIGH is demoted to the maximum label and ADMIN_LOW is promoted to the minimum label. External label view is the default condition.</p> <p>DEFAULT LABEL VIEW IS INTERNAL Unless otherwise specified, when an ADMIN_HIGH or ADMIN_LOW binary label is translated to an ASCII-coded label, the ASCII-coded label will be in internal form. In internal form ADMIN_HIGH is represented by the string ADMIN_HIGH and ADMIN_LOW is represented by the string ADMIN_LOW.</p> <p>FLOAT PROCESS INFORMATION LABEL Label translations from binary labels ASCII-coded labels float the caller's process information label based on the values specified in the NAME INFORMATION LABELS: section of the label encodings file. The caller's process information label floats by default.</p> <p>DO NOT FLOAT PROCESS INFORMATION LABEL Label translations from binary labels to ASCII-coded labels do not float the</p>

caller's process information label based on the values specified in the **NAME INFORMATION LABELS:** section of the label encodings file.

DEFAULT FLAGS= *value*

This option represents a default GFI **Flags=** keyword value to be used if no flags are specified as a parameter to the translation. Caution must be taken when defining a **DEFAULT FLAGS=** value that the appropriate **Flags=** values have been provided. A non-zero value also implies that label validation during translation from binary to ASCII-coded form is not done. The default value is 0 (zero).

FORCED FLAGS= *value*

This option represents a GFI **Flags=** keyword value to be used in all translations. Caution must be taken when defining a **FORCED FLAGS=** value that the appropriate **Flags=** values have been provided. A non-zero value also implies that label validation during translation from binary to ASCII-coded form is not done. The default value is 0 (zero).

CLASSIFICATION NAME= *name*

This option specifies the string to be displayed in the label builder GUI for the title of the *Classification* names section. Specifying a NULL value for *name* leaves the section without a title. The default value is **CLASSIFICATION**.

COMPARTMENTS NAME= *name*

This option specifies the string to be displayed in the label builder GUI for the title of the *Compartments Word* section. Specifying a NULL value for *name* leaves the section without a title. The default value is **COMPARTMENTS**.

MARKINGS NAME= *name*

This option specifies the string to be displayed in the label builder GUI for the title of the *Markings Word* section. Specifying a NULL value for *name* leaves the section without a title. The default value is **MARKINGS**.

The final part of the **LOCAL DEFINITIONS:** section defines the ASCII color names to be associated with various words, sensitivity labels, or classifications. This section supports the **bltcolor**(3TSOL) function. It consists of the **COLOR NAMES:** keyword and is followed by zero or more color to label assignments. Each statement has one of the following two syntaxes:

word= *word value*; **color=** *color value*;

label= *label value*; **color=** *color value*;

where *color value* is an ASCII color name to be associated with the word *word value*, sensitivity label *label value*, or classification *label value*.

The ASCII color name *color value* for a label is determined by the order of entries in the **COLOR NAMES:** section that make up the label. If a label contains a word *word value* that is specified in this section, the *color value* of the label is the one associated with the first *word value* specified. If no specified word *word value* is contained in the label, the *color value* is the one associated with an exact match of a *label value*. If there is no exact match, the *color value* is the one associated with the first specified *label value* whose classification matches the classification of the label.

EXAMPLES

Below is a sample **LOCAL DEFINITIONS:** section.

LOCAL DEFINITIONS:

ADMIN LOW NAME= LoLo; * It is strongly advised not to use this option
ADMIN HIGH NAME= HiHi; * It is strongly advised not to use this option

DEFAULT LABEL VIEW IS INTERNAL;
FLOAT PROCESS INFORMATION LABEL;

DEFAULT FLAGS= 0x4;
FORCED FLAGS= 0;

CLASSIFICATION NAME=; * No Classification name title
COMPARTMENTS NAME=; * No Compartments word title
MARKINGS NAME=; * No Markings word title

COLOR NAMES:

label= Admin_Low;	color= Pale Blue;	
label= unclassified;	color= light grey;	
word= Project A;	color= bright blue;	
label= c;	color= sea foam green;	
label= secret;	color= #ff0000;	* Hexadecimal RGB value
word= Hotel;	color= Lavender;	
word= KeLO;	color= red;	
label= TS;	color= khaki;	
label= TS Elephant;	color= yellow;	
label= Admin_High;	color= shocking pink;	

FILES

/etc/security/tsol/label_encodings

The label encodings file containing the *Classification* names, *Words*, constraints, and values for the defined labels of this system.

SEE ALSO

bcltobanner(3TSOL), **blinset(3TSOL)**, **bltcolor(3TSOL)**, **bltos(3TSOL)**, **blvalid(3TSOL)**, **labelinfo(3TSOL)**, **labelvers(3TSOL)**, **stobl(3TSOL)**, **chk_encodings(1MSTOL)**

Defense Intelligence Agency document DDS-2600-6216-93, *Compartmented Mode Workstation Labeling: Encodings Format (Release 2.2)*, September 1993.

DIAGNOSTICS

The following diagnostics are in addition to those found in Appendix A of *Compartmented Mode Workstation Labeling: Encodings Format*.

Admin_High color already assigned as "XXX".

A color has already been defined for the ADMIN_HIGH label. Another cannot be defined.

Admin_Low color already assigned as “XXX”.

A color has already been defined for the ADMIN_LOW label. Another cannot be defined.

BOUND TRANSLATION BY CLEARANCE obsolete,**Bound is always a Sensitivity Label.**

This option is obsolete and ignored. All label translations are bound by the calling process' *Sensitivity Label*.

BOUND TRANSLATION BY SENSITIVITY LABEL obsolete,**Bound is always a Sensitivity Label.**

This option is obsolete and ignored. All label translations are bound by the calling process' *Sensitivity Label*.

Can't allocate NNN bytes for ADMIN HIGH NAME=

The system cannot dynamically allocate the memory it needs to process the ADMIN HIGH NAME= option.

Can't allocate NNN bytes for ADMIN LOW NAME=

The system cannot dynamically allocate the memory it needs to process the ADMIN LOW NAME= option.

Can't allocate NNN bytes for CLASSIFICATION NAME=

The system cannot dynamically allocate the memory it needs to process the CLASSIFICATION NAME= option.

Can't allocate NNN bytes for COMPARTMENTS NAME=

The system cannot dynamically allocate the memory it needs to process the COMPARTMENTS NAME= option.

Can't allocate NNN bytes for MARKINGS NAME=

The system cannot dynamically allocate the memory it needs to process the MARKINGS NAME= option.

Can't allocate NNN bytes for color name “XXX”.

The system cannot dynamically allocate the memory it needs to store color name XXX.

Can't allocate NNN bytes for color names table.

The system cannot dynamically allocate the memory it needs to process the COLOR NAMES: section.

Can't allocate NNN bytes for color table entry.

The system cannot dynamically allocate the memory it needs to process a Color Table entry.

Can't allocate NNN bytes for color word entry.

The system cannot dynamically allocate the memory it needs to process a Color Word entry.

Duplicate ADMIN HIGH NAME= ignored.

More than one ADMIN LOW NAME= option was encountered. All but the first are ignored.

Duplicate ADMIN LOW NAME= ignored.

More than one ADMIN LOW NAME= option was encountered. All but the first are ignored.

Duplicate CLASSIFICATION NAME= ignored.

More than one CLASSIFICATION NAME= option was encountered. All but the first are ignored.

Duplicate COMPARTMENTS NAME= ignored.

More than one COMPARTMENTS NAME= option was encountered. All but the first are ignored.

Duplicate MARKINGS NAME= ignored.

More than one MARKINGS NAME= option was encountered. All but the first are ignored.

End of File or LOCAL DEFINITIONS: not found. Found instead: "XXX".

The noted extraneous text was found when the LOCAL DEFINITIONS: section or end of label encodings file was expected.

Found color "XXX" without associated label.

The color XXX was found, however it had no label or word associated with it.

Invalid color label "XXX".

The label XXX cannot be parsed.

Label preceding "XXX" did not have a color specification.

A label or word was found without a matching color name.

Word "XXX" not found as a valid Sensitivity Label word.

The word XXX was not found as a valid word for a *Sensitivity Label*.

WARNINGS

Creation of and modification to the label encodings file should only be undertaken with a thorough understanding of not only the concepts in *Compartmented Mode Workstation Labeling: Encodings Format*, but also the details as well of the local labeling requirements. The following warnings are paraphrased from *Compartmented Mode Workstation Labeling: Encodings Format*.

Take extreme care when modifying a label encodings file that is already loaded and running on a Trusted Solaris system. Once the system runs with the label encodings file, many objects are labeled with sensitivity labels and information labels that are well formed with respect to the loaded label encodings file. If the label encodings file is subsequently changed, it is possible that the existing labels will no longer be well formed. Changing the bit patterns associated with words causes existing objects whose labels contain the words to have possibly invalid labels. Raising the minimum classification or lowering the maximum classification associated with words will likely cause existing objects whose labels contain the words to no longer be well formed.

Changes to a current encodings file that has already been used should be limited only to adding new classifications or words, changing the names of existing words, or modifying the local extensions. As described in *Compartmented Mode Workstation Labeling: Encodings Format*, it is important to reserve extra inverse bits when the label encodings file is first

created to allow for later expansion of the label encodings file to incorporate new inverse words. If an inverse word is added that does not use reserved inverse bits, all existing objects on the system will erroneously have labels that include the new inverse word.

NOTES

Defining the label encodings file is a three-step process. First, the set of human-readable labels to be represented must be identified and understood. The definition of this set includes the list of classifications and other words used in the human-readable labels, relations between and among the words, classification restrictions associated with use of each word, and intended use of the words in mandatory access control and labeling system output. Next, this definition is associated with an internal format of integers, bit patterns, and logical relationship statements. Finally, a label encodings file is created. The *Compartmented Mode Workstation Labeling: Encodings Format* document describes the second and third steps, and assumes that the first has already been performed.

NAME	mnttab – mounted file system table
DESCRIPTION	<p>The file mnttab resides in /etc and contains information about devices that are <i>currently</i> mounted. mnttab is read by programs using the routines described in getmntent(3C). mount(1M) adds entries to this file. umount removes entries from this file. Each entry is a line of fields separated by spaces in the form:</p> <p style="padding-left: 40px;"><i>special mount_point fstype options time</i></p> <p>where</p> <p style="padding-left: 40px;"><i>special</i> The name of the resource to be mounted.</p> <p style="padding-left: 40px;"><i>mount_point</i> The pathname of the directory on which the filesystem is mounted.</p> <p style="padding-left: 40px;"><i>fstype</i> The file system type of the mounted file system.</p> <p style="padding-left: 40px;"><i>options</i> The mount options. (See respective mount file system man page below in SEE ALSO.)</p> <p style="padding-left: 40px;"><i>time</i> The time at which the file system was mounted.</p> <p>Examples of entries for the <i>special</i> field include the pathname of a block-special device, the name of a remote filesystem in <i>host:pathname</i> form, or the name of a “swap file” (for instance, a file made with mkfile(1M)).</p>
SUMMARY OF TRUSTED SOLARIS CHANGES	The /etc/mnttab file must have a sensitivity label of ADMIN_LOW and have an owner UID of 0.
FILES	/etc/mnttab
SEE ALSO	mkfile(1M) , mount_cachefs(1M) , mount_hfs(1M) , mount_nfs(1M) , mount_pcfs(1M) , mount_ufs(1M) , mount(1M) , setmnt(1M) , getmntent(3C)

NAME	nsswitch.conf – configuration file for the name service switch																																																				
SYNOPSIS	<code>/etc/nsswitch.conf</code>																																																				
DESCRIPTION	<p>The operating system uses a number of "databases" of information about hosts, users (passwd/shadow), groups and so forth. Data for these can come from a variety of sources: host-names and host-addresses, for example, may be found in /etc/hosts, NIS, NIS+ or DNS. Zero or more sources may be used for each database; the sources and their lookup order are specified in the /etc/nsswitch.conf file.</p> <p>The following databases use the switch file:</p> <table> <thead> <tr> <th><i>Database</i></th> <th><i>Used by</i></th> </tr> </thead> <tbody> <tr> <td>aliases</td> <td>sendmail(1M)</td> </tr> <tr> <td>automount</td> <td>automount(1M)</td> </tr> <tr> <td>bootparams</td> <td>rpc.bootparamd(1M)</td> </tr> <tr> <td>ethers</td> <td>ethers(3N)</td> </tr> <tr> <td>group</td> <td>getgrnam(3C)</td> </tr> <tr> <td>hosts</td> <td>gethostbyname(3N) (See "Interaction with netconfig" below)</td> </tr> <tr> <td>netgroup</td> <td>innetgr(3N)</td> </tr> <tr> <td>netmasks</td> <td>ifconfig(1M)</td> </tr> <tr> <td>networks</td> <td>getnetbyname(3N)</td> </tr> <tr> <td>passwd</td> <td>getpwnam(3C), getspnam(3C)</td> </tr> <tr> <td>protocols</td> <td>getprotobyname(3N)</td> </tr> <tr> <td>publickey</td> <td>getpublickey(3N), secure_rpc(3N)</td> </tr> <tr> <td>rpc</td> <td>getrpcbyname(3N)</td> </tr> <tr> <td>sendmailvars</td> <td>sendmail(1M)</td> </tr> <tr> <td>services</td> <td>getservbyname(3N) (See "Interaction with netconfig" below)</td> </tr> <tr> <td>tsolprof</td> <td>tsolprof(4TSOL)</td> </tr> <tr> <td>tsoluser</td> <td>tsoluser(4TSOL)</td> </tr> <tr> <td>tnrhdb</td> <td>tnrhdb(4TSOL)</td> </tr> <tr> <td>tnrhtp</td> <td>tnrhtp(4TSOL)</td> </tr> </tbody> </table> <p>The following sources may be used:</p> <table> <thead> <tr> <th><i>Source</i></th> <th><i>Uses</i></th> </tr> </thead> <tbody> <tr> <td>files</td> <td>/etc/hosts, /etc/passwd, /etc/shadow and so forth</td> </tr> <tr> <td>nis</td> <td>NIS (YP)</td> </tr> <tr> <td>nisplus</td> <td>NIS+</td> </tr> <tr> <td>dns</td> <td>Valid only for hosts; uses the Internet Domain Name Service.</td> </tr> <tr> <td>compat</td> <td>Valid only for passwd and group; implements "+" and "-". (See "Interaction with +/- syntax" below)</td> </tr> </tbody> </table> <p>There is an entry in /etc/nsswitch.conf for each database. Typically these entries will be simple, such as "protocols: files" or "networks: files nisplus". However, when multiple sources are specified it is sometimes necessary to define precisely the circumstances</p>	<i>Database</i>	<i>Used by</i>	aliases	sendmail (1M)	automount	automount (1M)	bootparams	rpc.bootparamd (1M)	ethers	ethers (3N)	group	getgrnam (3C)	hosts	gethostbyname (3N) (See "Interaction with netconfig" below)	netgroup	innetgr (3N)	netmasks	ifconfig (1M)	networks	getnetbyname (3N)	passwd	getpwnam (3C), getspnam (3C)	protocols	getprotobyname (3N)	publickey	getpublickey (3N), secure_rpc (3N)	rpc	getrpcbyname (3N)	sendmailvars	sendmail (1M)	services	getservbyname (3N) (See "Interaction with netconfig" below)	tsolprof	tsolprof (4TSOL)	tsoluser	tsoluser (4TSOL)	tnrhdb	tnrhdb (4TSOL)	tnrhtp	tnrhtp (4TSOL)	<i>Source</i>	<i>Uses</i>	files	/etc/hosts , /etc/passwd , /etc/shadow and so forth	nis	NIS (YP)	nisplus	NIS+	dns	Valid only for hosts ; uses the Internet Domain Name Service.	compat	Valid only for passwd and group ; implements "+" and "-". (See "Interaction with +/- syntax" below)
<i>Database</i>	<i>Used by</i>																																																				
aliases	sendmail (1M)																																																				
automount	automount (1M)																																																				
bootparams	rpc.bootparamd (1M)																																																				
ethers	ethers (3N)																																																				
group	getgrnam (3C)																																																				
hosts	gethostbyname (3N) (See "Interaction with netconfig" below)																																																				
netgroup	innetgr (3N)																																																				
netmasks	ifconfig (1M)																																																				
networks	getnetbyname (3N)																																																				
passwd	getpwnam (3C), getspnam (3C)																																																				
protocols	getprotobyname (3N)																																																				
publickey	getpublickey (3N), secure_rpc (3N)																																																				
rpc	getrpcbyname (3N)																																																				
sendmailvars	sendmail (1M)																																																				
services	getservbyname (3N) (See "Interaction with netconfig" below)																																																				
tsolprof	tsolprof (4TSOL)																																																				
tsoluser	tsoluser (4TSOL)																																																				
tnrhdb	tnrhdb (4TSOL)																																																				
tnrhtp	tnrhtp (4TSOL)																																																				
<i>Source</i>	<i>Uses</i>																																																				
files	/etc/hosts , /etc/passwd , /etc/shadow and so forth																																																				
nis	NIS (YP)																																																				
nisplus	NIS+																																																				
dns	Valid only for hosts ; uses the Internet Domain Name Service.																																																				
compat	Valid only for passwd and group ; implements "+" and "-". (See "Interaction with +/- syntax" below)																																																				

under which each source will be tried. A source can return one of the following codes:

<i>Status</i>	<i>Meaning</i>
SUCCESS	Requested database entry was found
UNAVAIL	Source is not responding or corrupted
NOTFOUND	Source responded "no such entry"
TRYAGAIN	Source is busy, might respond to retries

For each status code, two actions are possible:

<i>Action</i>	<i>Meaning</i>
continue	Try the next source in the list
return	Return now

The complete syntax of an entry is

```
<entry> ::= <database> ":" [<source> [<criteria>]]*
<criteria> ::= "[" <criterion>+ "]"
<criterion> ::= <status> "=" <action>
<status> ::= "success" | "notfound" | "unavail" | "tryagain"
<action> ::= "return" | "continue"
```

Each entry occupies a single line in the file. Lines that are blank, or that start with white space are ignored. Everything on a line following a # character is also ignored; the # character can begin anywhere in a line, to be used to begin comments. The <database> and <source> names are case-sensitive, but <action> and <status> names are case-insensitive.

The library functions contain compiled-in default entries that are used if the appropriate entry in **nsswitch.conf** is absent or syntactically incorrect.

The default criteria are to continue on anything except **SUCCESS**; in other words, [SUCCESS=return NOTFOUND=continue UNAVAIL=continue TRYAGAIN=continue].

The default, or explicitly specified, criteria are meaningless following the last source in an entry; and are ignored since the action is always to return to the caller irrespective of the status code the source returns.

Interaction with netconfig

In order to ensure that they all return consistent results, **gethostbyname(3N)**, **getservbyname(3N)**, and **netdir_getbyname(3N)** functions are all implemented in terms of the same internal library function. This function obtains the system-wide source lookup policy for **hosts** and **services** based on the **inet** family entries in **netconfig(4)** and uses the switch entries only if the netconfig entries have a "-" in the last column for name-toaddr libraries. See the **NOTES** section in **gethostbyname(3N)** and **getservbyname(3N)** for details.

Interaction with NIS+ NIS/YP- compatibility Mode

The NIS+ server can be run in "YP-compatibility mode", where it handles NIS (YP) requests as well as NIS+ requests. In this case, the clients get much the same results (except for **getspnam(3C)**) from the "nis" source as from "nisplus"; however, "nisplus" is recommended instead of "nis".

Interaction with NIS (YP) server in DNS-forwarding Mode

The NIS (YP) server can be run in "DNS-forwarding mode", where it forwards lookup requests to DNS for host-names and -addresses that do not exist in its database. In this case, specifying "nis" as a source for "hosts" is sufficient to get DNS lookups; "dns" need not be specified explicitly as a source.

Since SunOS 5.3 (Solaris 2.3), the NIS+ server in "NIS/YP-compatibility mode" can also be run in "DNS-forwarding mode" (see **rpc.nisd(1M)**). Forwarding is effective only for requests originating from its YP clients; "hosts" policy on these clients should be configured appropriately.

Interaction with Password Aging

When password aging is turned on only a limited set of possible name services are permitted for the **passwd:** database in the **/etc/nsswitch.conf** file.

```
passwd:      files
passwd:      files nis
passwd:      files nisplus
passwd:      compat

passwd:      compat
passwd_compat: nisplus
```

Any other settings will cause the **passwd(1)** command to fail when it attempts to change the password after expiration and will prevent the user from logging in. These are the *only* permitted settings when password aging has been turned on. Otherwise you can work around incorrect **passwd:** lines by using the **-r repository** argument to the **passwd(1)** command and using **passwd -r repository** to override the **nsswitch.conf** settings and specify in which name service you want to modify your password.

Interaction with +/- syntax

Releases prior to SunOS 5.0 did not have the name service switch but did allow the user some policy control. In **/etc/passwd** one could have entries of the form **+user** (include the specified user from NIS **passwd.byname**), **-user** (exclude the specified user) and **+** (include everything, except excluded users, from NIS **passwd.byname**). The desired behavior was often "everything in the file followed by everything in NIS", expressed by a solitary **+** at the end of **/etc/passwd**. The switch provides an alternative for this case ("passwd: files nis") that does not require **+** entries in **/etc/passwd** and **/etc/shadow** (the latter is a new addition to SunOS 5.0, see **shadow(4)**).

If this is not sufficient, the NIS/YP compatibility source provides full +/- semantics. It reads **/etc/passwd** for **getpwnam(3C)** functions and **/etc/shadow** for **getspnam(3C)** functions and, if it finds +/- entries, invokes an appropriate source. By default the source is "nis", but this may be overridden by specifying "nisplus" as the source for the pseudo-database **passwd_compat**.

Note that for every **/etc/passwd** entry, there should be a corresponding entry in the **/etc/shadow** file.

The NIS/YP compatibility source also provides full +/- semantics for **group**; the relevant pseudo-database is **group_compat**.

**Useful
Configurations**

The compiled-in default entries for all databases use NIS (YP) as the enterprise level name service and are identical to those in the default configuration of this file:

```
passwd:      files nis
group:      files nis
hosts:      nis [NOTFOUND=return] files
networks:   nis [NOTFOUND=return] files
protocols:  nis [NOTFOUND=return] files
rpc:        nis [NOTFOUND=return] files
ethers:     nis [NOTFOUND=return] files
netmasks:  nis [NOTFOUND=return] files
bootparams: nis [NOTFOUND=return] files
publickey:  nis [NOTFOUND=return] files
netgroup:   nis
automount:  files nis
aliases:    files nis
services:   files nis
sendmailvars: files
```

The policy "nis [NOTFOUND=return] files" implies "if **nis** is UNAVAIL, continue on to **files**, and if **nis** returns NOTFOUND, return to the caller; in other words, treat **nis** as the authoritative source of information and try **files** only if **nis** is down." This, and other policies listed in the default configuration above, are identical to the hard-wired policies in SunOS releases prior to 5.0.

If compatibility with the +/- syntax for passwd and group is required, simply modify the entries for **passwd** and **group** to:

```
passwd:      compat
group:       compat
```

If NIS+ is the enterprise level name service, the default configuration should be modified to use **nisplus** instead of **nis** for every database on client machines. The file `/etc/nsswitch.nisplus` contains a sample configuration that can be copied to `/etc/nsswitch.conf` to set this policy.

If the use of +/- syntax is desired in conjunction with **nisplus**, use the following four entries:

```
passwd:      compat
passwd_compat: nisplus
group:       compat
group_compat: nisplus
```

In order to get information from the Internet Domain Name Service for hosts that are not listed in the enterprise level name service, NIS+, use the following configuration and set up the `/etc/resolv.conf` file (see `resolv.conf(4)` for more details):

```
hosts:      nisplus dns [NOTFOUND=return] files
```

**Enumeration --
getXXXent()**

Many of the databases have enumeration functions: **passwd** has `getpwent()`, **hosts** has `gethostent()`, and so on. These were reasonable when the only source was **files** but often make little sense for hierarchically structured sources that contain large numbers of

entries, much less for multiple sources. The interfaces are still provided and the implementations strive to provide reasonable results, but the data returned may be incomplete (enumeration for **hosts** is simply not supported by the **dns** source), inconsistent (if multiple sources are used), formatted in an unexpected fashion (for a host with a canonical name and three aliases, the **nisplus** source will return four hostents, and they may not be consecutive), or very expensive (enumerating a **passwd** database of 5000 users is probably a bad idea). Furthermore, multiple threads in the same process using the same reentrant enumeration function (**getXXXent_r()**) are supported beginning with SunOS 5.3 share the same enumeration position; if they interleave calls, they will enumerate disjoint subsets of the same database.

In general the use of the enumeration functions is deprecated. In the case of **passwd**, **shadow** and **group**, it may sometimes be appropriate to use **fgetgrent()**, **fgetpwent()** and **fgetspent()** (see **getgrnam(3C)**, **getpwnam(3C)**, and **getspnam(3C)**, respectively), which use only the **files** source.

SUMMARY OF TRUSTED SOLARIS CHANGES

The following Trusted Solaris files have been added: **tsolprof**, **tsoluser**, **tnrhdb**, and **tnrhtp**.

In the default Trusted Solaris environment, an administrative role uses the Application Manager to edit the **nsswitch.conf** file. This file should not be edited directly.

FILES

A source named SSS is implemented by a shared object named **nss_SSS.so.1** that resides in **/usr/lib**.

/etc/nsswitch.conf	configuration file
/usr/lib/nss_compat.so.1	implements "compat" source
/usr/lib/nss_dns.so.1	implements "dns" source
/usr/lib/nss_files.so.1	implements "files" source
/usr/lib/nss_nis.so.1	implements "nis" source
/usr/lib/nss_nisplus.so.1	implements "nisplus" source
/etc/netconfig	configuration file for netdir(3N) functions that redirects hosts/sevices policy to the switch
/etc/nsswitch.files	sample configuration file that uses "files" only
/etc/nsswitch.nis	sample configuration file that uses "files" and "nis"
/etc/nsswitch.nisplus	sample configuration file that uses "files" and "nisplus"

SEE ALSO

nis+(1), **passwd(1)**, **automount(1M)**, **ifconfig(1M)**, **rpc.bootparamd(1M)**, **rpc.nisd(1M)**, **sendmail(1M)**, **ethers(3N)**, **getgrnam(3C)**, **gethostbyname(3N)**, **getnetbyname(3N)**, **getnetgrent(3N)**, **getprotobyname(3N)**, **getpublickey(3N)**, **getpwnam(3C)**, **getrpcbyname(3N)**, **getservbyname(3N)**, **getspnam(3C)**, **netdir(3N)**, **secure_rpc(3N)**, **netconfig(4)**, **resolv.conf(4)**, **ypfiles(4)**

NOTES

Within each process that uses **nsswitch.conf**, the entire file is read only once; if the file is later changed, the process will continue using the old configuration.

Programs that use the **getXXbyYY()** functions cannot be linked statically since the implementation of these functions requires dynamic linker functionality to access the shared objects **/usr/lib/nss_SSS.so.1** at run time.

The use of both **nis** and **nisplus** as sources for the same database is strongly discouraged since both the name services are expected to store similar information and the lookups on the database may yield different results depending on which name service is operational at the time of the request.

Misspelled names of sources and databases will be treated as legitimate names of (most likely nonexistent) sources and databases.

The following functions do *not* use the switch: **fgetgrent(3C)**, **fgetpwent(3C)**, **fgetspent(3C)**, **getpw(3C)**, **putpwent(3C)**. **shadow(4)**

NAME	priv_desc – Descriptions of defined privileges
SYNOPSIS	#include <tsol/priv.h>
AVAILABILITY	SUNWtsolu
DESCRIPTION	<p>Every defined privilege has a manifest constant for use in programs, a name for use in user interfaces, and a description displayed by certain administrative tools. When a process has a privilege in its <i>effective</i> set, that process has the power to bypass security policy and perform the task allowed by that privilege. The manifest constant, name, and description for each privilege defined on this system follows.</p> <p>Manifest Constant PRIV_FILE_AUDIT Name file_audit</p> <p>Allows a process to get or set a file's or directory's audit preselection information. The audit preselection information may override the preselection information associated with a process' access to a file or directory. Allows a process to get or set a file's or directory's public object flag. The public object flag may override the successful read/search access preselection information associated with a process' access to a file or directory. Allows a process to write to or modify a file or directory without the file's or directory's audit preselection information or public object flag being cleared.</p> <p>Manifest Constant PRIV_FILE_CHOWN Name file_chown</p> <p>Allows a process to change a file's owner user ID. Allows a process to change a file's group ID to one other than the process' effective group ID or one of the process' supplemental group IDs.</p> <p>Manifest Constant PRIV_FILE_DAC_EXECUTE Name file_dac_execute</p> <p>Allows a process to execute an executable file whose permission bits or ACL do not allow the process execute permission.</p> <p>Manifest Constant PRIV_FILE_DAC_READ Name file_dac_read</p> <p>Allows a process to read a file or directory whose permission bits or ACL do not allow the process read permission.</p> <p>Manifest Constant PRIV_FILE_DAC_SEARCH Name file_dac_search</p>

Allows a process to search a directory whose permission bits or ACL do not allow the process search permission.

Manifest Constant PRIV_FILE_DAC_WRITE

Name file_dac_write

Allows a process to write a file or directory whose permission bits or ACL do not allow the process write permission.

Manifest Constant PRIV_FILE_DOWNGRADE_IL

Name file_downgrade_il

Allows a process to set the Information Label of a file or directory to an Information Label that does not dominate the existing Information Label. Allows a process to specify an Information Label in the pwritel(2TSOL), writel(2TSOL), and writevl(2TSOL) interfaces which does not dominate the process' Information Label.

Manifest Constant PRIV_FILE_DOWNGRADE_SL

Name file_downgrade_sl

Allows a process to set the Sensitivity Label of a file or directory to a Sensitivity Label that does not dominate the existing Sensitivity Label.

Manifest Constant PRIV_FILE_LOCK

Name file_lock

Allows a process to get accurate lock information for a file lock that it does not hold.

Manifest Constant PRIV_FILE_MAC_READ

Name file_mac_read

Allows a process to read a file or directory whose Sensitivity Label is not dominated by the process' Sensitivity Label. Note that a process reading such a file or directory may also need the proc_nofloat privilege to avoid illegal Information Label floating. Allows a process to get accurate file attributes of a file or directory whose Sensitivity Label is not dominated by the process' Sensitivity Label. Allows a process, when upgraded directory names are hidden, to get directory entries whose Sensitivity Label is not dominated by the process' Sensitivity Label.

Manifest Constant PRIV_FILE_MAC_SEARCH

Name file_mac_search

Allows a process to search a directory whose Sensitivity Label is not dominated by the process' Sensitivity Label.

Manifest Constant PRIV_FILE_MAC_WRITE

Name file_mac_write

Allows a process to write a file or directory whose Sensitivity Label does not dominate the process' Sensitivity Label, or whose Sensitivity Label dominates the process' Clearance.

Manifest Constant PRIV_FILE_NOFLOAT

Name file_nofloat

Allows a process to write to a file or directory without modifying its Information Label.

Manifest Constant PRIV_FILE_OWNER

Name file_owner

Allows a process which is not the owner of a file to modify that file's access and modification times, audit preselection attributes, privileges, or downgrade labels. Allows a process which is not the owner of a directory to modify that directory's access and modification times or downgrade labels. Allows a process which is not the owner of a file or directory to remove or rename a file or directory whose parent directory has the "save text image after execution" (sticky) bit set. Allows a process which is not the owner of a file to mount a "namefs" upon that file. (Does not apply to setting access permission bits or ACLs.)

Manifest Constant PRIV_FILE_SETDAC

Name file_setdac

Allows a process which is not the owner of a file or directory to modify that file's or directory's permission bits or ACL.

Manifest Constant PRIV_FILE_SETID

Name file_setid

Allows a process to change the ownership of a file or write to a file without the set-user-ID and set-group-ID bits being cleared. Allows a process to set the set-group-ID bit on a file whose group is not the process' effective group or one of the process' supplemental groups.

Manifest Constant PRIV_FILE_SETPRIV

Name file_setpriv

Allows a process to set the privilege sets on an executable file that the process owns. Allows a process to write to an executable file without the file's allowed and forced privilege sets being emptied.

Manifest Constant PRIV_FILE_UPGRADE_IL

Name file_upgrade_il

Allows a process to set the Information Label of a file or directory to an Information Label that dominates the existing Information Label. Allows a process to specify an Information Label in the `pwrite(2TSOL)`, `writel(2TSOL)`, and `writelv(2TSOL)` interfaces which dominates the process' Information Label.

Manifest Constant PRIV_FILE_UPGRADE_SL

Name file_upgrade_sl

Allows a process to set the Sensitivity Label of a file or directory to a Sensitivity Label that dominates the existing Sensitivity Label.

Manifest Constant PRIV_IPC_DAC_READ

Name ipc_dac_read

Allows a process to read a System V IPC Message Queue, Semaphore Set, or Shared Memory Segment whose permission bits or ACL do not allow the process read permission.

Manifest Constant PRIV_IPC_DAC_WRITE

Name ipc_dac_write

Allows a process to write a System V IPC Message Queue, Semaphore Set, or Shared Memory Segment whose permission bits or ACL do not allow the process write permission.

Manifest Constant PRIV_IPC_DOWNGRADE_IL

Name ipc_downgrade_il

Allows a process to set the Information Label on a Semaphore or set of Semaphores written to a System V IPC Semaphore Set to an Information Label that does not dominate the process' Information Label. Allows a process to set the Information Label on a message written to a System V IPC Message Queue to an Information Label that does not dominate the process' Information Label.

Manifest Constant PRIV_IPC_MAC_READ

Name ipc_mac_read

Allows a process to read a System V IPC Message Queue, Semaphore Set, or Shared Memory Segment whose Sensitivity Label is not dominated by the process' Sensitivity Label. Reading a System V IPC Message Queue also requires MAC write access. Note that a process reading such a System V IPC object may also need the `proc_nofloat` privilege to avoid illegal Information Label floating.

Manifest Constant PRIV_IPC_MAC_WRITE

Name ipc_mac_write

Allows a process to write a System V IPC Message Queue, Semaphore Set, or Shared Memory Segment whose Sensitivity Label does not dominate the process' Sensitivity Label, or whose Sensitivity Label dominates the process' Clearance.

Manifest Constant PRIV_IPC_NOFLOAT

Name ipc_nofloat

Allows a process to send a System V IPC Message or write to a System V IPC Semaphore Set without changing its Information Label. Allows changes to a process' Information Label to not modify the Information Label of a System V Shared Memory Segment that the process has attached for writing.

Manifest Constant PRIV_IPC_OWNER

Name ipc_owner

Allows a process which is not the owner of a System V IPC Message Queue, Semaphore Set, or Shared Memory Segment to remove, change ownership of, or change permission bits or ACL of the Message Queue, Semaphore Set, or Shared Memory Segment.

Manifest Constant PRIV_IPC_UPGRADE_IL

Name ipc_upgrade_il

Allows a process to set the Information Label on a Semaphore or set of Semaphores written to a System V IPC Semaphore Set to an Information Label that dominates the process' Information Label. Allows a process to set the Information Label on a Message written to a System V IPC Message Queue to an Information Label that dominates the process' Information Label.

Manifest Constant PRIV_NET_BROADCAST

Name net_broadcast

Allows a process to send broadcast or multicast packets. Because broadcast packets are delivered to all machines on the local network, they are not labeled.

Manifest Constant PRIV_NET_DOWNGRADE_IL

Name net_downgrade_il

Allows a process to specify an Information Label for data being written or to set the network endpoint default Information Label to an Information Label which does not dominate the process' Information Label.

Manifest Constant PRIV_NET_DOWNGRADE_SL

Name net_downgrade_sl

Allows a process to specify a Sensitivity Label for data being written or to set the network endpoint default Sensitivity Label to an Sensitivity Label which does not dominate the process' Sensitivity Label.

Manifest Constant PRIV_NET_MAC_READ

Name net_mac_read

Allows a process to bind to or accept with a multi-level port. Binding to a multi-level port allows the process to read all data sent to that port socket for which there is not a bound single level port that matches the Sensitivity Label of the data. Accepting with a multi-level port allows a process to receive all data sent to that connected port. (There can be no single level connected port for the accept to succeed.) Allows a process to create a multi-level RPC port mapping. Note that this privilege is intended for server processes. Applying it to client processes may have undesirable side effects.

Manifest Constant PRIV_NET_NOFLOAT

Name net_nofloat

Allows a process to send data that when received will not cause the receiving process' Information Label to float.

Manifest Constant PRIV_NET_PRIVADDR

Name net_privaddr

Allows a process to bind to a privileged port number. The privilege port numbers are 1-1023 (the traditional UNIX privileged ports) and 6000-6002 (the XSun server ports). Privileged port numbers include the Internet reserved (well known) port numbers.

Manifest Constant PRIV_NET_RAWACCESS

Name net_rawaccess

Allows a process to have direct access to the network layer. Direct access to the network layer bypasses network labeling. Auditing is not bypassed.

Manifest Constant PRIV_NET_REPLY_EQUAL

Name net_reply_equal

Allows a process to reply with the Sensitivity Label and Information Label of the last packet received rather than its own Sensitivity Label and Information Label. A combination of net_mac_read and net_reply_equal allow unmodified programs to successfully receive and reply at all Sensitivity Labels. This privilege exists for unmodified program compatibility and is not used by modified Trusted Solaris programs.

Manifest Constant PRIV_NET_SETCLR

Name net_setclr

Allows a process to specify a Clearance for data being written or to set the network endpoint default Clearance to a value different from the process' Clearance.

Manifest Constant PRIV_NET_SETID

Name net_setid

Allows a process to specify an effective user ID, effective group ID, or set of supplemental groups for data being written or to set the network endpoint default effective user ID, effective group ID, or set of supplemental groups to values different from the process' values. Allows a process which is not the owner of a RPC port mapping to remove the mapping.

Manifest Constant PRIV_NET_SETPRIV

Name net_setpriv

Allows a process to specify the effective privilege set for data being written or to set the network endpoint default effective privilege set to privileges contained in the process' permitted privilege set.

Manifest Constant PRIV_NET_UPGRADE_IL

Name net_upgrade_il

Allows a process to specify an Information Label for data being written or to set the network endpoint default Information Label to an Information Label which dominates the process' Information Label.

Manifest Constant PRIV_NET_UPGRADE_SL

Name net_upgrade_sl

Allows a process to specify a Sensitivity Label for data being written or to set the network endpoint default Sensitivity Label to a Sensitivity Label which dominates the process' Sensitivity Label.

Manifest Constant PRIV_PROC_AUDIT_APPL

Name proc_audit_appl

Allows a process to generate audit records with an audit event outside the Trusted Solaris TCB event number range. Allows a process to get its own audit preselection information.

Manifest Constant PRIV_PROC_AUDIT_TCB

Name proc_audit_tcb

Allows a process to generate audit records with an audit event in the Trusted Solaris TCB event number range. Allows a process to get its own audit preselection information.

Manifest Constant PRIV_PROC_CHROOT

Name proc_chroot

Allows a process to change its root directory.

Manifest Constant PRIV_PROC_DUMPCORE

Name proc_dumpcore

Allows a TCB process to execute a new program which is set-user-ID, set-group-ID, or permits the use of privilege to have a “core” file created for it when taking the default action for SIGQUIT, SIGILL, SIGTRAP, SIGABRT, SIGEMT, SIGFPE, SIGBUS, SIGSEGV, SIGSYS, SIGXCPU, or SIGXFSZ signals. Allows a TCB process to have a “core” file created for it when taking the default action for SIGQUIT, SIGILL, SIGTRAP, SIGABRT, SIGEMT, SIGFPE, SIGBUS, SIGSEGV, SIGSYS, SIGXCPU, or SIGXFSZ signals.

Manifest Constant PRIV_PROC_MAC_READ

Name proc_mac_read

Allows a process to read another process whose Sensitivity Label is not dominated by the reading process' Sensitivity Label. Note that a process reading such a process may also need the proc_nofloat privilege to avoid illegal Information Label floating.

Manifest Constant PRIV_PROC_MAC_WRITE

Name proc_mac_write

Allows a process to write another process whose Sensitivity Label does not dominate the writing process' Sensitivity Label, or whose Sensitivity Label dominates the writing process' Clearance.

Manifest Constant PRIV_PROC_NODELAY

Name proc_nodelay

Allows a process to not be delayed when doing operations that are identified as covert channels.

Manifest Constant PRIV_PROC_NOFLOAT

Name proc_nofloat

Allows a process to read information into its address space without the Information Label of that information causing the process' Information Label to be floated. Note that this privilege may also be used to avoid illegal Information Label floating caused by reading of upgraded objects through the use of a MAC read privilege.

Manifest Constant PRIV_PROC_OWNER

Name proc_owner

Allows a process to read from and write to another process with a different process owner. Allows a process to bind a process to a CPU with a different process owner.

Manifest Constant PRIV_PROC_SETCLR

Name proc_setclr

Allows a process to set its Clearance to a Clearance that is not equal to the process' current Clearance.

Manifest Constant PRIV_PROC_SETID

Name proc_setid

Allows a process to set its user or group IDs to one different from its current effective, real, or saved IDs. Allows a process to set its supplemental group IDs. Allows a process to set the process group of a controlling terminal to one not in the process' process group. Allows a process to set the window size on a terminal not in its session.

Manifest Constant PRIV_PROC_SETIL

Name proc_setil

Allows a process to set its Information Label to an Information Label that is not equal to the process' current Information Label.

Manifest Constant PRIV_PROC_SETSL

Name proc_setsl

Allows a process to set its Sensitivity Label to a Sensitivity Label that is not equal to the process' current Sensitivity Label.

Manifest Constant PRIV_PROC_TRANQUIL

Name proc_tranquil

Allows a process to set the Sensitivity Label of an object to a Sensitivity Label that is not equal to the current Sensitivity Label when the object is in use by another process.

Manifest Constant PRIV_SYS_AUDIT

Name sys_audit

Allows a process to start the (kernel) audit daemon. Allows a process to view and set the audit state (audit user ID, audit terminal ID, audit session ID, audit preselection mask). Allows a process to turn off and on auditing. Allows a process to configure the audit parameters (cache and queue sizes, event to class mappings, policy options).

Manifest Constant PRIV_SYS_BOOT

Name sys_boot

Allows a process to halt, re-boot, or suspend a Trusted Solaris machine.

Manifest Constant PRIV_SYS_CONFIG

Name sys_config

Allows a process to lock into memory and unlock from memory a memory mapped file or Shared Memory Segment. Allows a process to change the scheduling priority of a process not owned by this process, or increase this process' priority. Allows a process to increase its resource or process limits. Allows a process to set the "save text image after execution" (sticky) bit on executable files. Allows a process to turn on and off accounting. Allows a process to change the machine time of day clock. Allows a process to change the machine high resolution timer clock. Allows a process to reconfigure scheduling classes. Allows a process to create and delete (hard) links to directories. Allows a process to place a processor on-line or off-line. Allows a process to modify kernel driver statistics values.

Manifest Constant PRIV_SYS_CONSOLE

Name sys_console

Allows a process to redirect console output to another device.

Manifest Constant PRIV_SYS_DEVICES

Name sys_devices

Allows a process to create device special files. Allows a process to use mknod(2) to create directory and regular files. Allows a process to revoke all access to a device special file. Allows a process to reassign a controlling terminal from one process to another. Allows a process to open a terminal already exclusively opened. Allows a process to revoke access to its controlling terminal. Allows a process to enable or disable keyboard abort processing. Allows a process to map frame buffer devices into its address space. Allows a process to enable or disable a disk's write-check capability. Allows a process to load a kernel loadable driver. Allows a process to control the Floating Point Accelerator. Allows a process to configure autopush STREAMS modules. Allows a process to configure the device driver policy table. Allows a process to successfully call a third party loadable module that calls the kernel drv_priv(9F) function to check for allowed access.

Manifest Constant PRIV_SYS_FS_CONFIG

Name sys_fs_config

Allows a process to manipulate filesystem locks. Allows a process to set/clear the automatic update (delayed I/O) state of a filesystem. Allows a process to get meta disk allocation information. Allows a process to open a specified inode in a filesystem. Allows a process to set the last access time of a file system object.

Manifest Constant PRIV_SYS_IPC_CONFIG

Name sys_ipc_config

Allows a process to increase the size of a System V IPC Message Queue buffer.

Manifest Constant PRIV_SYS_MAXPROC

Name sys_maxproc

Allows a process to create processes when the maximum number of processes for this process' owning user is exceeded. Allows a process to create the last available process in the system.

Manifest Constant PRIV_SYS_MINFREE

Name sys_minfree

Allows a process to write to a filesystem whose available storage space is below the minimum allowed.

Manifest Constant PRIV_SYS_MOUNT

Name sys_mount

Allows a process to mount filesystems which are restricted from being freely mounted. Such filesystems include those of type ufs, nfs, tmpfs, procfs, ... Allows a process to remount the root filesystem. Allows a process to add and remove swap filesystems. Allows a process to determine the users of a filesystem.

Manifest Constant PRIV_SYS_NET_CONFIG

Name sys_net_config

Allows a process to configure a machine's network interfaces and routes. Allows a process to set a machine's host and domain names. Allows a process to set a machine's kerberos realm. Allows a process to load and unload host type, accreditation, and default information. Allows a process direct access to network devices. Allows a process to set endpoint names. Allows a process to use the rpcmod STREAMS module.

Manifest Constant PRIV_SYS_NFS

Name sys_nfs

Allows a process to start a kernel NFS daemon. Allows a process to start and stop a kernel NFS lock manager daemon. Allows a process to export directories for use by NFS clients. Allows a process to retrieve the NFS file handle for a path name. Allows a process to revoke NFS RPC credentials for a client it does not own.

Manifest Constant PRIV_SYS_SUSER_COMPAT

Name sys_suser_compat

Allows a process to successfully call a third party loadable module that calls the kernel suser(9K) function to check for allowed access. This privilege exists only for third party loadable module compatibility and is not used by Trusted Solaris.

Manifest Constant PRIV_SYS_TRANS_LABEL

Name sys_trans_label

Allows a process to translate labels to and from “external string form” that are not dominated by the process’ Sensitivity Label.

Manifest Constant PRIV_WIN_COLORMAP

Name win_colormap

Allows a process to override colormap restrictions. Allows a process to install or remove colormaps. Allows a process to retrieve colormap cell entries allocated by other processes.

Manifest Constant PRIV_WIN_CONFIG

Name win_config

Allows a process to configure or destroy resources that are permanently retained by the X server. Allows a process to use SetScreenSaver to set the screen saver timeout value. Allows a process to use ChangeHosts to modify the display access control list. Allows a process to use GrabServer. Allows a process to use the SetCloseDownMode request which may retain window, pixmap, colormap, property, cursor, font, or graphic context resources.

Manifest Constant PRIV_WIN_DAC_READ

Name win_dac_read

Allows a process to read from a window resource that it does not own (has a different user ID).

Manifest Constant PRIV_WIN_DAC_WRITE

Name win_dac_write

Allows a process to write to or create a window resource that it does not own (has a different user ID). A newly created window property is created with the window’s user ID.

Manifest Constant PRIV_WIN_DEVICES

Name win_devices

Allows a process to perform operations on window input devices. Allows a process to get and set keyboard and pointer controls. Allows a process to modify pointer button and key mappings.

Manifest Constant PRIV_WIN_DGA

Name win_dga

Allows a process to use the direct graphics access (DGA) X protocol extensions. Direct process access to the frame buffer is still required. Thus the process must have MAC and DAC privileges that allow access to the frame buffer, or the frame buffer must be allocated to the process.

Manifest Constant PRIV_WIN_DOWNGRADE_IL

Name win_downgrade_il

Allows a process to set the Information Label of a window resource to an Information Label that does not dominate the existing Information Label.

Manifest Constant PRIV_WIN_DOWNGRADE_SL

Name win_downgrade_sl

Allows a process to set the Sensitivity Label of a window resource to a Sensitivity Label that does not dominate the existing Sensitivity Label.

Manifest Constant PRIV_WIN_FONTPATH

Name win_fontpath

Allows a process to set a font path.

Manifest Constant PRIV_WIN_MAC_READ

Name win_mac_read

Allows a process to read from a window resource whose Sensitivity Label is not equal to the process Sensitivity Label. Note that a process reading such a window resource may also need the the proc_nofloat privilege to avoid illegal Information Label floating.

Manifest Constant PRIV_WIN_MAC_WRITE

Name win_mac_write

Allows a process to write to create a window resource whose Sensitivity Label is not equal to the process Sensitivity Label. A newly created window property is created with the window's Sensitivity Label.

Manifest Constant PRIV_WIN_NOFLOAT

Name win_nofloat

Allows a process to write to a window resource without modifying the resource's Information Label.

Manifest Constant PRIV_WIN_SELECTION

Name win_selection

Allows a process to request inter-window data moves without the intervention of the selection arbitrator.

Manifest Constant PRIV_WIN_UPGRADE_IL

Name win_upgrade_il

Allows a process to set the Information Label of a window resource to an Information Label that dominates the existing Information Label.

Manifest Constant PRIV_WIN_UPGRADE_SL

Name win_upgrade_sl

Allows a process to set the Sensitivity Label of a window resource to a Sensitivity Label that dominates the existing Sensitivity Label.

FILES

`/usr/lib/tsol/locale/locale/priv_name` Privileges descriptions

`/usr/include/sys/tsol/priv_names.h` Manifest constant and ID value definitions

SEE ALSO

Intro(2TSOL), **getfpriv**(2TSOL), **setfpriv**(2TSOL), **priv_to_str**(3TSOL), **set_effective_priv**(3TSOL) **priv_name**(4TSOL), **priv_macros**(5TSOL),

Trusted Solaris administrator's document set

Trusted Solaris Developer's Guide

NAME	priv_name – privilege description database
SYNOPSIS	<code>/usr/lib/tsol/locale/locale/priv_name</code>
AVAILABILITY	SUNWtsolu
DESCRIPTION	<p>The priv_name database specifies localized privilege names and descriptions defined on this system. This database is used along with the <code><sys/tsol/priv_names.h></code> file by priv_to_str(3TSOL), str_to_priv(3TSOL), and get_priv_text(3TSOL) to translate between privilege ID, privilege name string, and description.</p> <p>Each entry in the priv_name database consists of one line with fields separated by colons (:). A line ending with a backslash (\) indicates continuation of the entry on the next line. Lines beginning with a # character are treated as comments. Each entry has the form:</p> <p style="text-align: center;"><i>constant:name:description</i></p> <p>The entry fields are:</p> <p><i>constant</i> The <i>constant</i> field must be identical to the manifest constant defined for the privilege in the <code><sys/tsol/priv_names.h></code> file, where a unique privilege ID is assigned to each privilege constant.</p> <p><i>name</i> The external name of the privilege. It is returned by priv_to_str and is used by str_to_priv. It is also used by commands like ppriv and pprivtest. The external name can be customized and localized.</p> <p><i>description</i> The description of the privilege. It is returned by get_priv_text. The description can be customized and localized.</p>
EXAMPLES	<pre># # Example entry in /usr/lib/tsol/locale/C/priv_name # PRIV_PROC_SETID:proc_setid:Allows a process to set its user or group ID to \ one different from its current effective, real, or saved IDs. \ Allows a process to set its supplemental group IDs. \ Allows a process to set the process group of a controlling terminal \ to one not in the process' process group. \ Allows a process to set the window size on a terminal not in its \ session.</pre>
SEE ALSO	priv_to_str (3TSOL), priv_desc (4TSOL)

NAME	proc – /proc, the process file system
DESCRIPTION	<p>/proc is a file system that provides access to the image of each process in the system. The name of each entry in the /proc directory is a decimal number corresponding to the process-ID. The owner of each “file” is determined by the process’s real user-ID.</p> <p>Standard system call interfaces are used to access /proc files: open(2TSOL), close(2), read(2TSOL), write(2TSOL), and ioctl(2). An open for reading and writing enables process control; a read-only open allows inspection but not control. As with ordinary files, more than one process can open the same /proc file at the same time. <i>Exclusive open</i> is provided to allow controlling processes to avoid collisions: an open(2TSOL) for writing that specifies O_EXCL fails if the file is already open for writing; if such an exclusive open succeeds, subsequent attempts to open the file for writing, with or without the O_EXCL flag, fail until the exclusively-opened file descriptor is closed. (Exception: an open(2TSOL) by a process with the PRIV_SYS_DEVICES privilege that does not specify O_EXCL succeeds even if the file is exclusively opened.) There can be any number of read-only opens, even when an exclusive write open is in effect on the file.</p> <p>Data may be transferred from or to any locations in the traced process’s address space by applying lseek(2TSOL) to position the file followed by read(2TSOL) or write(2TSOL). The PIOCMAP operation can be applied to determine the accessible areas (mappings) of the address space. I/O transfers may span contiguous mappings. An I/O request extending into an unmapped area is truncated at the boundary. An I/O request beginning at an unmapped virtual address fails with EIO.</p> <p>Information and control operations are provided through ioctl(2). These have the form:</p> <pre> #include <sys/types.h> #include <sys/signal.h> #include <sys/fault.h> #include <sys/syscall.h> #include <sys/procfs.h> void *p; retval = ioctl(fildes, code, p); </pre> <p>The argument <i>p</i> is a generic pointer whose type depends on the specific ioctl code. Where not specifically mentioned below, its value should be zero. <sys/procfs.h> contains definitions of ioctl codes and data structures used by the operations.</p> <p>Every active process contains at least one <i>light-weight process</i>, or <i>lwp</i>. Each <i>lwp</i> represents a flow of execution that is independently scheduled by the operating system. The PIOCOPENLWP operation can be applied to the process file descriptor to obtain a specific <i>lwp</i> file descriptor. I/O operations produce identical results whether applied to the process file descriptor or to an <i>lwp</i> file descriptor. All /proc ioctl operations can be applied to either type of file descriptor and, where not stated otherwise, produce identical results.</p> <p>Process information and control operations involve the use of sets of flags. The set types sigset_t, fltset_t, and sysset_t correspond, respectively, to signal, fault, and system call enumerations defined in <sys/signal.h>, <sys/fault.h>, and <sys/syscall.h>. Each set</p>

type is large enough to hold flags for its own enumeration. Although they are of different sizes, they have a common structure and can be manipulated by these macros:

```
prfillset(&set);           /* turn on all flags in set */
premptyset(&set);         /* turn off all flags in set */
praddset(&set, flag);     /* turn on the specified flag */
prdelset(&set, flag);     /* turn off the specified flag */
r = prismember(&set, flag); /* != 0 iff flag is turned on */
```

One of `prfillset()` or `premptyset()` must be used to initialize `set` before it is used in any other operation. `flag` must be a member of the enumeration corresponding to `set`.

IOCTLS

The allowable `ioctl` codes follow. Certain of these can be used only if the process or lwp file descriptor is open for writing; these include all operations that affect process control. Those requiring write access are marked with an asterisk (*). Except where noted, an `ioctl` to a process or lwp that has terminated elicits the error `ENOENT`.

The following IOCTLS provided by Trusted Solaris are used to get information about the security attributes of a process: `PIOCLABEL`, `PIOCCLEAR`, `PIOCEPRIV`, `PIOCIPRIV`, `PIOCPPRIV`, `PIOCSPRIV`, `PIOCTRED`, and `PIOCATTR`. See the `DESCRIPTION` and `NOTES` sections for information about privileges and MAC policies that apply to the use of the `/proc` file system in Trusted Solaris.

PIOCSTATUS

`PIOCSTATUS` returns status information for the process and one of its `lpws`. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the `PRIV_FILE_MAC_READ` privilege. `p` is a pointer to a `prstatus` structure containing at least the following fields:

```
typedef struct prstatus {
    long          pr_flags;           /* Flags */
    short         pr_why;             /* Reason for stop (if stopped) */
    short         pr_what;           /* More detailed reason */
    id_t          pr_who;            /* Specific lwp identifier */
    u_short       pr_nlwps;          /* Number of lwps in the process */
    short         pr_cursig;         /* Current signal */
    sigset_t      pr_sigpend;        /* Set of process pending signals */
    sigset_t      pr_lwppend;        /* Set of lwp pending signals */
    sigset_t      pr_sighold;        /* Set of lwp held signals */
    struct siginfo pr_info;           /* Info associated with signal or fault */
    struct sigaltstack pr_altstack;  /* Alternate signal stack info */
    struct sigaction pr_action;       /* Signal action for current signal */
    struct ucontext *pr_oldcontext;   /* Address of previous ucontext */
    caddr_t       pr_brkbase;        /* Address of the process heap */
    u_long        pr_brksize;        /* Size of the process heap, in bytes */
    caddr_t       pr_stkbase;        /* Address of the process stack */
    u_long        pr_stksize;        /* Size of the process stack, in bytes */
    short         pr_syscall;        /* System call number (if in syscall) */
};
```

```

short      pr_nsysarg;          /* Number of arguments to this syscall */
long      pr_sysarg[PRSYSARGS]; /* Arguments to this syscall */
pid_t     pr_pid;              /* Process id */
pid_t     pr_ppid;            /* Parent process id */
pid_t     pr_pgrp;            /* Process group id */
pid_t     pr_sid;             /* Session id */
timestruc_t pr_utime;          /* Process user cpu time */
timestruc_t pr_stime;          /* Process system cpu time */
timestruc_t pr_cutime;         /* Sum of children's user times */
timestruc_t pr_cstime;        /* Sum of children's system times */
char      pr_clname[PRCLSZ];  /* Scheduling class name */
long      pr_instr;           /* Current instruction */
pgregset_t pr_reg;            /* General registers */
} prstatus_t;

```

pr_flags is a bit-mask holding these flags:

```

PR_STOPPED    lwp is stopped
PR_ISTOP      lwp is stopped on an event of interest (see PIOCSTOP)
PR_DSTOP      lwp has a stop directive in effect (see PIOCSTOP)
PR_STEP       lwp has a single-step directive in effect (see PIOCRUN)
PR_ASLEEP     lwp is in an interruptible sleep within a system call
PR_PCINVAL    lwp's current instruction (pr_instr) is undefined
PR_ISSYS      process is a system process (see PIOCSTOP)
PR_FORK       process has its inherit-on-fork flag set (see PIOCSET)
PR_RLC        process has its run-on-last-close flag set (see PIOCSET)
PR_KLC        process has its kill-on-last-close flag set (see PIOCSET)
PR_ASYNC      process has its asynchronous-stop flag set (see PIOCSET)
PR_PCOMPAT    process has its ptrace-compatibility flag set (see PIOCSET)

```

pr_why and **pr_what** together describe, for a stopped *lwp*, the reason for the stop. Possible values of **pr_why** are:

```

PR_REQUESTED  indicates that the stop occurred in response to a stop directive, normally because PIOCSTOP was applied or because another lwp stopped on an event of interest and the asynchronous-stop flag (see PIOCSET) was not set for the process. pr_what is unused in this case.

PR_SIGNALED   indicates that the lwp stopped on receipt of a signal (see PIOCSTRACE); pr_what holds the signal number that caused the stop (for a newly-stopped lwp, the same value is in pr_cursig).

PR_FAULTED    indicates that the lwp stopped on incurring a hardware fault (see PIOCFAULT); pr_what holds the fault number that caused the stop.

```

```

PR_SYSENTRY and PR_SYSEXIT
                    indicate a stop on entry to or exit from a system call (see

```

	PIOCSENTRY and PIOCSEXIT); pr_what holds the system call number.
PR_JOBCONTROL	indicates that the <i>lwp</i> stopped due to the default action of a job control stop signal (see sigaction(2)); pr_what holds the stopping signal number.
PR_SUSPENDED	indicates that the <i>lwp</i> stopped due to internal synchronization of <i>lwps</i> within the process. pr_what is unused in this case.

pr_who names the specific *lwp*. **pr_nlwps** is the total number of *lwps* in the process.

pr_cursig names the current signal, that is, the next signal to be delivered to the *lwp*.

pr_sigpend identifies any other signals pending for the process. **pr_lwppend** identifies any synchronously-generated or directed signals pending for the *lwp*. **pr_sighold** identifies those signals whose delivery is being delayed if sent to the *lwp*.

pr_info, when the *lwp* is in a **PR_SIGNALLED** or **PR_FAULTED** stop, contains additional information pertinent to the particular signal or fault (see `<sys/siginfo.h>`).

pr_altstack contains the alternate signal stack information for the *lwp* (see **sigaltstack(2)**).

pr_action contains the signal action information pertaining to the current signal (see **sigaction(2)**); it is undefined if **pr_cursig** is zero.

pr_oldcontext, if not **NULL**, contains the address in the process of a **ucontext** structure describing the previous user-level context (see **ucontext(5)**). It is non-**NULL** only if the *lwp* is executing in the context of a signal handler and is the same as the **ucontext** pointer passed to the signal handler.

pr_brkbase is the virtual address of the process heap and **pr_brksize** is its size in bytes. The address formed by the sum of these values is the process *break* (see **brk(2)**).

pr_stkbase and **pr_stksize** are, respectively, the virtual address of the process stack and its size in bytes. (Each *lwp* runs on a separate stack; the distinguishing characteristic of the “process stack” is that the operating system will grow it when necessary.)

pr_syscall is the number of the system call, if any, being executed by the *lwp*; it is non-zero only if the *lwp* is stopped on **PR_SYSENTRY** or **PR_SYSEXIT** or is asleep within a system call (**PR_ASLEEP** is set). If **pr_syscall** is non-zero, **pr_nsysarg** is the number of arguments to the system call and the **pr_sysarg** array contains the actual arguments.

pr_pid, **pr_ppid**, **pr_pgrp**, and **pr_sid** are, respectively, the process id, the id of the process’s parent, the process’s process group id, and the process’s session id.

pr_utime, **pr_stime**, **pr_cutime**, and **pr_cstime** are, respectively, the user CPU and system CPU time consumed by the process, and the cumulative user CPU and system CPU time consumed by the process’s children, in seconds and nanoseconds.

pr_clname contains the name of the *lwp*’s scheduling class.

pr_processor is the ordinal number of the processor that last ran this *lwp*. **pr_bind** is the ordinal number of the processor to which this *lwp* is bound, or **PBIND_NONE** if the *lwp* is not bound to a processor.

pr_instr contains the machine instruction to which the *lwp*'s program counter refers. The amount of data retrieved from the process is machine-dependent. On SPARC machines, it is a 32-bit word. On x86 machines, it is a single byte. In general, the size is that of the machine's smallest instruction. If **PR_PCINVAL** is set, **pr_instr** is undefined; this occurs whenever the *lwp* is not stopped or when the program counter refers to an invalid virtual address.

SPARC: **pr_reg** is an array holding the contents of a stopped *lwp*'s general registers. On SPARC machines the predefined constants **R_G0** ... **R_G7**, **R_O0** ... **R_O7**, **R_L0** ... **R_L7**, **R_I0** ... **R_I7**, **R_PSR**, **R_PC**, **R_nPC**, **R_Y**, **R_WIM**, and **R_TBR** can be used as indices to refer to the corresponding registers; previous register windows can be read from their overflow locations on the stack (see, however, **PIOCGWIN**). If the *lwp* is not stopped, all register values are undefined.

x86: **pr_reg** is an array holding the contents of a stopped *lwp*'s general registers. On x86 machines, the predefined constants **SS**, **UESP**, **EFL**, **CS**, **EIP**, **ERR**, **TRAPNO**, **EAX**, **ECX**, **EDX**, **EBX**, **ESP**, **EBP**, **ESI**, **EDI**, **DS**, **ES**, **FS**, and **GS** can be used as indices to refer to the corresponding registers. If the *lwp* is not stopped, all register values are undefined.

When applied to an *lwp* file descriptor, **PIOCSTATUS** returns the status for the specific *lwp*. When applied to the process file descriptor, an *lwp* is chosen by the system for the operation. The chosen *lwp* is a stopped *lwp* only if all of the process's *lwps* are stopped, is stopped on an event of interest only if all of the *lwps* are so stopped (excluding **PR_SUSPENDED** *lwps*), is in a **PR_REQUESTED** stop only if there are no other events of interest to be found, or failing everything else is in a **PR_SUSPENDED** stop (implying that the process is deadlocked). The chosen *lwp* remains fixed so long as all of the *lwps* are either stopped on events of interest or are **PR_SUSPENDED** and **PIOCRUN** is not applied to any of them.

When applied to the process file descriptor, every **/proc ioctl** operation that must act on an *lwp* uses the same algorithm to choose which *lwp* to act upon. Together with synchronous stopping (see **PIOCSET**), this enables a debugger to control a multiple-*lwp* process using only the process file descriptor if it so chooses. More fine-grained control can be achieved using individual *lwp* file descriptors.

PIOCLSTATUS

The **PIOCLSTATUS** operation fills in an array of **prstatus** structures addressed by *p*, one element (one structure) for each *lwp* in the process, containing the status that would be returned by applying **PIOCSTATUS** to the corresponding *lwp* file descriptor, plus an additional element at the beginning containing the status that would be returned by applying **PIOCSTATUS** to the process file descriptor.

Mandatory read access to the file descriptor is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege.

***PIOCSTOP** ***PIOCWSTOP**

When applied to the process file descriptor, **PIOCSTOP** directs all *lwps* to stop and waits for them to stop; **PIOCWSTOP** simply waits for all *lwps* to stop.

Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_WRITE** privilege.

When applied to an lwp file descriptor, **PIOCSTOP** directs the specific lwp to stop and waits until it has stopped; **PIOCWSTOP** simply waits for the lwp to stop. When applied to an lwp file descriptor, these operations complete when the lwp stops on an event of interest, immediately if already so stopped. When applied to the process file descriptor, they complete when every lwp has stopped on an event of interest or has come to a **PR_SUSPENDED** stop. If *p* is non-zero it points to a **prstatus** structure to be filled with status information for the specific or chosen stopped lwp (see **PIOCSTATUS**).

An “event of interest” is either a **PR_REQUESTED** stop or a stop that has been specified in the process’s tracing flags (set by **PIOCTRACE**, **PIOCSFAULT**, **PIOCSENTRY**, and **PIOCSEXIT**). **PR_JOBCONTROL** and **PR_SUSPENDED** stops are specifically not events of interest. (An lwp may stop twice due to a stop signal, first showing **PR_SIGNALLED** if the signal is traced and again showing **PR_JOBCONTROL** if the lwp is set running without clearing the signal.) If **PIOCSTOP** is applied to an lwp that is stopped, but not on an event of interest, the stop directive takes effect when the lwp is restarted by the competing mechanism; at that time the lwp enters a **PR_REQUESTED** stop before executing any user-level code.

ioctls are interruptible by signals so that, for example, an **alarm(2)** can be set to avoid waiting forever for a process or lwp that may never stop on an event of interest. If **PIOCSTOP** is interrupted, the lwp stop directives remain in effect even though the **ioctl** returns an error.

A system process (indicated by the **PR_ISSYS** flag) never executes at user level, has no user-level address space visible through **/proc**, and cannot be stopped. Applying **PIOCSTOP** or **PIOCWSTOP** to a system process or any of its *lwps* elicits the error **EBUSY**.

*PIOCRUN

An lwp is made runnable again after a stop. Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_WRITE** privilege.

If *p* is non-zero it points to a **prrun** structure describing additional actions to be performed. The **prrun** structure contains at least the following fields:

```
typedef struct prrun {
    long      pr_flags;      /* Flags */
    sigset_t  pr_trace;     /* Set of signals to be traced */
    sigset_t  pr_sighold;   /* Set of signals to be held */
    fltset_t  pr_fault;     /* Set of faults to be traced */
    caddr_t   pr_vaddr;     /* Virtual address at which to resume */
} prrun_t;
```

pr_flags is a bit-mask describing optional actions; the remainder of the entries are meaningful only if the appropriate bits are set in **pr_flags**. Flag definitions:

PRCSIG	clears the current signal, if any (see PIOCSSIG).
PRCFAULT	clears the current fault, if any (see PIOCCFAULT).
PRSTRACE	sets the traced signal set to pr_trace (see PIOCTRACE).
PRSHOLD	sets the held signal set to pr_sighold (see PIOCSHOLD).

PRSFault	sets the traced fault set to pr_fault (see PIOCSFault).
PRSVADDR	sets the address at which execution resumes to pr_vaddr .
PRSTEP	directs the lwp to execute a single machine instruction. On completion of the instruction, a trace trap occurs. If FLTRACE is being traced, the lwp stops, otherwise it is sent SIGTRAP ; if SIGTRAP is being traced and not held, the lwp stops. When the lwp stops on an event of interest the single-step directive is cancelled, even if the stop occurs before the instruction is executed. This operation requires hardware and operating system support and may not be implemented on all processors. It is implemented on SPARC and x86 machines.
PRSABoRT	is meaningful only if the lwp is in a PR_SYSENTRY stop or is marked PR_ASLEEP ; it instructs the lwp to abort execution of the system call (see PIOCSENTRY , PIOCSEXIT).
PRSToP	directs the lwp to stop again as soon as possible after resuming execution (see PIOCSToP). In particular if the lwp is stopped on PR_SIGNALLEd or PR_FAULTEd , the next stop will show PR_REQUESTEd , no other stop will have intervened, and the lwp will not have executed any user-level code.

When applied to an lwp file descriptor **PIOCRUN** makes the specific lwp runnable. The operation fails (**EBUSY**) if the specific lwp is not stopped on an event of interest.

When applied to the process file descriptor an lwp is chosen for the operation as described under **PIOCSTATUS**. The operation fails (**EBUSY**) if the chosen lwp is not stopped on an event of interest. If **PRSTEP** or **PRSToP** was requested, the chosen lwp is made runnable; otherwise, the chosen lwp is marked **PR_REQUESTEd**. If as a consequence all *lwps* are in the **PR_REQUESTEd** or **PR_SUSPENDED** stop state, all *lwps* showing **PR_REQUESTEd** are made runnable.

PIOCLWPIDS This returns, in an array of **id_t**'s addressed by *p*, the lwp identifiers of all the *lwps* that exist in the process, plus an extra identifier containing zero to mark the end of the list. The number of *lwps* in the process can be determined from the **pr_nlwp** field of the **prstatus** structure.

Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege.

PIOCNLDT
PIOCLDT These operations apply only to x86 machines. They provide read-only access to the traced process's local descriptor table (LDT). A process's LDT is maintained by the operating system. **PIOCNLDT** returns, in an integer addressed by *p*, the number of LDT entries currently active. This value can be used with the **PIOCLDT** operation. The **PIOCLDT** operation returns the set of currently active LDT entries. For **PIOCLDT**, *p* addresses an array of elements of type **struct ssd**, defined in `<sys/sysi86.h>`. One array element (one structure) is returned for each active LDT entry, plus an additional element containing all zeroes to mark the end of the list.

	<p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.</p> <p>If information label floating is configured for the system, the information label of the calling process floats unless the PRIV_PROC_NOFLOAT privilege is asserted.</p>
PIOCOPENLWP	<p>The return value <i>retval</i> provides a /proc file descriptor that refers to the lwp named in the id_t addressed by <i>p</i>. The read/write attributes of the newly-acquired file descriptor are the same as those of the file descriptor used in the operation. The new file descriptor has an independent file offset for lseek(2TSOL). On error (no such <i>lwp</i>), -1 is returned and errno is set to ENOENT.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.</p>
*PIOCTRACE	<p>This defines a set of signals to be traced in the process: the receipt of one of these signals by an lwp causes the lwp to stop. The set of signals is defined via an instance of sigset_t addressed by <i>p</i>. Receipt of SIGKILL cannot be traced; if specified, it is silently ignored.</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>If a signal that is included in an <i>lwp</i>'s held signal set is sent to the <i>lwp</i>, the signal is not received and does not cause a stop until it is removed from the held signal set, either by the lwp itself or by setting the held signal set with PIOCSHOLD or the PRSHOLD option of PIOCRUN.</p>
PIOCGTRACE	<p>The current traced signal set is returned in an instance of sigset_t addressed by <i>p</i>.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.</p>
*PIOCSSIG	<p>The current signal and its associated signal information for the specific or chosen <i>lwp</i> are set according to the contents of the siginfo structure addressed by <i>p</i> (see <sys/siginfo.h>).</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>If the specified signal number is zero or if <i>p</i> is zero, the current signal is cleared. The semantics of this operation are different from that of kill(2TSOL) or PIOCKILL in that the signal is delivered to the lwp immediately after execution is resumed (even if the signal is being held) and an additional PR_SIGNALED stop does not intervene even if the signal is being traced. Setting the current signal to SIGKILL terminates the process immediately.</p>
*PIOCKILL	<p>If applied to the process file descriptor, a signal is sent to the process with semantics identical to that of kill(2TSOL).</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p>

*	<p>If applied to an <i>lwp</i> file descriptor, a directed signal is sent to the specific <i>lwp</i>. <i>p</i> points to an <i>int</i> naming the signal. Sending SIGKILL terminates the process immediately, even if the signal is sent to a specific <i>lwp</i>.</p>														
*	<p>PIOCUNKILL A signal is deleted, that is, it is removed from the set of pending signals. If applied to the process file descriptor, the signal is deleted from the process's pending signals. If applied to an <i>lwp</i> file descriptor, the signal is deleted from the <i>lwp</i>'s pending signals. The current signal (if any) is unaffected. <i>p</i> points to an <i>int</i> naming the signal. It is an error (EINVAL) to attempt to delete SIGKILL. Mandatory write access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p>														
*	<p>PIOCGHOLD PIOCGHOLD returns the set of held signals for the specific or chosen <i>lwp</i> (signals whose delivery will be delayed if sent to the <i>lwp</i>) in an instance of sigset_t addressed by <i>p</i>. PIOCSHOLD correspondingly sets the <i>lwp</i>'s held signal set but does not allow SIGKILL or SIGSTOP to be held; if specified, they are silently ignored.</p> <p>For PIOCSHOLD mandatory write access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege. For PIOCGHOLD mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.</p>														
*	<p>PIOCMAXSIG PIOCACTION These operations provide information about the signal actions associated with the traced process (see sigaction(2)). PIOCMAXSIG returns, in the <i>int</i> addressed by <i>p</i>, the maximum signal number understood by the system. This can be used to allocate storage for use with the PIOCACTION operation, which returns the traced process's signal actions in an array of sigaction structures addressed by <i>p</i>. Signal numbers are displaced by 1 from array indices, so that the action for signal number <i>n</i> appears in position <i>n-1</i> of the array.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.</p>														
*	<p>PIOCSFAULT This defines a set of hardware faults to be traced in the process: on incurring one of these faults an <i>lwp</i> stops.</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>The set is defined via an instance of fltset_t addressed by <i>p</i>. Fault names are defined in <sys/fault.h> and include the following. Some of these may not occur on all processors; there may be processor-specific faults in addition to these.</p> <table border="0" style="margin-left: 2em;"> <tr> <td>FLTILL</td> <td>illegal instruction</td> </tr> <tr> <td>FLTPRIV</td> <td>privileged instruction</td> </tr> <tr> <td>FLTBPT</td> <td>breakpoint trap</td> </tr> <tr> <td>FLTTRACE</td> <td>trace trap</td> </tr> <tr> <td>FLTACCESS</td> <td>memory access fault (bus error)</td> </tr> <tr> <td>FLTBOUNDS</td> <td>memory bounds violation</td> </tr> <tr> <td>FLTIOVF</td> <td>integer overflow</td> </tr> </table>	FLTILL	illegal instruction	FLTPRIV	privileged instruction	FLTBPT	breakpoint trap	FLTTRACE	trace trap	FLTACCESS	memory access fault (bus error)	FLTBOUNDS	memory bounds violation	FLTIOVF	integer overflow
FLTILL	illegal instruction														
FLTPRIV	privileged instruction														
FLTBPT	breakpoint trap														
FLTTRACE	trace trap														
FLTACCESS	memory access fault (bus error)														
FLTBOUNDS	memory bounds violation														
FLTIOVF	integer overflow														

FLTIZDIV	integer zero divide
FLTFPE	floating-point exception
FLTSTACK	unrecoverable stack fault
FLTPAGE	recoverable page fault

When not traced, a fault normally results in the posting of a signal to the lwp that incurred the fault. If an lwp stops on a fault, the signal is posted to the lwp when execution is resumed unless the fault is cleared by **PIOCCFAULT** or by the **PRCFAULT** option of **PIOCRUN**. **FLTPAGE** is an exception; no signal is posted. There may be additional processor-specific faults like this. **pr_info** in the **prstatus** structure identifies the signal to be sent and contains machine-specific information about the fault.

PIOCGFAULT	The current traced fault set is returned in an instance of fltset_t addressed by <i>p</i> . Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
*PIOCCFAULT	The current fault (if any) is cleared; the associated signal is not sent to the specific or chosen <i>lwp</i> . Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.
PIOCLABEL	Returns both the sensitivity and information label of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
PIOCCLEAR	Returns the clearance of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
PIOCEPRIV	Returns the effective privilege set of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
PIOCIPRIV	Returns the inheritable privilege set of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
PIOCPPRIV	Returns the permitted privilege set of the process associated with the file descriptor. Read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
PIOCSPRIV	Returns the saved privilege set of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.

PIOCATTR	Returns the Trusted Solaris process attributes of the process associated with the file descriptor. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege. For the calling process to receive the file system object's PAF_LABEL_XLAT attribute flags, the PAF_TRUSTED_PATH attribute flag of the calling process must be set.
PIOCAPSA	Returns the audit attributes of the calling process. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege. The calling process may assert one of the following: PRIV_PROC_AUDIT_APPL or PRIV_PROC_AUDIT_TCB or PRIV_SYS_AUDIT privilege.
PIOCTCRED	Returns the Trusted Solaris process credentials of the process associated with the file descriptor. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
* PIOCSENTRY * PIOCSEXIT	<p>These operations instruct the process's <i>lwps</i> to stop on entry to or exit from specified system calls.</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>The set of system calls to be traced is defined via an instance of sysset_t addressed by <i>p</i>. When entry to a system call is being traced, an <i>lwp</i> stops after having begun the call to the system but before the system call arguments have been fetched from the <i>lwp</i>. When exit from a system call is being traced, an <i>lwp</i> stops on completion of the system call just prior to checking for signals and returning to user level. At this point all return values have been stored into the <i>lwp</i>'s registers.</p> <p>If an <i>lwp</i> is stopped on entry to a system call (PR_SYSENTRY) or when sleeping in an interruptible system call (PR_ASLEEP is set), it may be instructed to go directly to system call exit by specifying the PR_SABORT flag in a PIOCRUN request. Unless exit from the system call is being traced the <i>lwp</i> returns to user level showing error EINTR.</p>
PIOCGENTRY PIOCGEXIT	These return the current traced system call entry or exit set in an instance of sysset_t addressed by <i>p</i> . Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
* PIOCSET * PIOCRESET	<p>PIOCSET sets one or more modes of operation for the traced process.</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>PIOCRESET resets these modes. The modes to be set or reset are specified by flags in a <i>long</i> addressed by <i>p</i>:</p> <p style="padding-left: 2em;">PR_FORK (inherit-on-fork): When set, the process's tracing flags are inherited by the child of a fork(2TSOL) or vfork(2TSOL). When reset, child processes start with all tracing flags cleared.</p> <p style="padding-left: 2em;">PR_RLC (run-on-last-close): When set and the last writable /proc file descriptor</p>

referring to the traced process or any of its *lwps* is closed, all of the process's tracing flags are cleared, any outstanding stop directives are canceled, and if any *lwps* are stopped on events of interest, they are set running as though **PIOCRUN** had been applied to them. When reset, the process's tracing flags are retained and *lwps* are not set running on last close.

PR_KLC (kill-on-last-close): When set and the last writable **/proc** file descriptor referring to the traced process or any of its *lwps* is closed, the process is terminated with **SIGKILL**.

PR_ASYNC (asynchronous-stop): When set, a stop on an event of interest by one lwp does not directly affect any other lwp in the process. When reset and an lwp stops on an event of interest other than **PR_REQUESTED**, all other *lwps* in the process are directed to stop.

PR_PCOMPAT (ptrace-compatibility): When set, a stop on an event of interest by the traced process is reported to the parent of the traced process via **wait(2)**, **SIGTRAP** is sent to the traced process when it executes a successful **exec(2TSOL)**, **setuid/setgid** flags are not honored for execs performed by the traced process, any **exec** of an object file that the traced process cannot read fails, and the traced process dies when its parent dies. This mode is deprecated; it is provided only to allow **ptrace(2)** to be implemented as a library function using **/proc**.

PR_MSACCT (microstate accounting): When set, microstate accounting is enabled for the process. This allows **PIOCUSAGE** to return accurate values for the times the *lwps* spent in their various processing states. If **PR_FORK** (inherit-on-fork) is also set, microstate accounting will be enabled for future child processes. When reset (the default) the overhead of microstate accounting is avoided and **PIOCUSAGE** can only return an estimate of times spent in the various states.

PR_BPTADJ (breakpoint trap pc adjustment): On x86 machines, a breakpoint trap leaves the program counter (the **EIP**) referring to the breakpointed instruction plus one byte. When **PR_BPTADJ** is set, the system will adjust the program counter back to the location of the breakpointed instruction when the lwp stops on a breakpoint. This flag has no effect on SPARC machines, where breakpoint traps leave the program counter referring to the breakpointed instruction.

It is an error (**EINVAL**) to specify flags other than those described above or to apply these operations to a system process. The current modes are reported in the **prstatus** structure (see **PIOCSTATUS**).

***PIOCSFORK**
***PIOCRFORK**

PIOCSFORK sets the inherit-on-fork flag in the traced process. **PIOCRFORK** turns this flag off.

Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_WRITE** privilege.

(Obsolete, see **PIOCSET**.)

<p>*PIOCSRLC *PIOCRRLC</p>	<p>PIOCSRLC sets the run-on-last-close flag in the traced process. PIOCRRLC turns this flag off. (Obsolete, see PIOCSET.)</p> <p>Mandatory write access is required to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p>
<p>PIOCGREG *PIOCSREG</p>	<p>These operations respectively get and set the general registers for the specific or chosen lwp into or out of an array addressed by <i>p</i>; the array has type prgregset_t. Register contents are accessible using a set of predefined indices (see PIOCSTATUS).</p> <p>On SPARC systems, only certain bits of the processor-status register (R_PS) can be modified by PIOCSREG: these include only the condition-code bits. Other privileged registers cannot be modified at all.</p> <p>On x86 systems, only certain bits of the flags register (EFL) can be modified by PIOCSREG: these include the condition codes, direction-bit, trace-bit, and overflow-bit.</p> <p>For PIOCGREG mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege. For PIOCSFPREG mandatory write access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege.</p> <p>PIOCSREG fails (EBUSY) if the lwp is not stopped on an event of interest. If the lwp is not stopped, the register values returned by PIOCGREG are undefined.</p>
<p>PIOCGFPREG *PIOCSFPREG</p>	<p>These operations respectively get and set the floating-point registers for the specific or chosen lwp into or out of a structure addressed by <i>p</i>; the structure has type prfpregset_t. An error (EINVAL) is returned if the system does not support floating-point operations (no floating-point hardware and the system does not emulate floating-point machine instructions).</p> <p>For PIOCGFPREG mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege. For PIOCSFPREG mandatory write access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_WRITE privilege. PIOCSFPREG fails (EBUSY) if the lwp is not stopped on an event of interest. If the lwp is not stopped, the register values returned by PIOCGFPREG are undefined.</p>
<p>*PIOCNICE</p>	<p>The traced process's nice(2) priority is incremented by the amount contained in the <i>int</i> addressed by <i>p</i>. Only the process asserting the PRIV_SYS_CONFIG privilege may better a process's priority in this way, but any user may lower the priority. This operation is meaningful only when applied to a process in the time-sharing scheduling class.</p> <p>Mandatory write access to the file system object is required. To override this restriction, the calling process may assert the FILE_MAC_WRITE privilege.</p>
<p>PIOCPSINFO</p>	<p>This returns miscellaneous process information such as that reported by ps(1).</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege. <i>p</i> is a pointer to a prpsinfo structure containing at least the following fields:</p>

```

typedef struct prpsinfo {
    char    pr_state;          /* numeric process state (see pr_sname) */
    char    pr_sname;         /* printable character representing pr_state */
    char    pr_zomb;          /* !=0: process terminated but not waited for */
    char    pr_nice;          /* nice for cpu usage */
    u_long  pr_flag;          /* process flags */
    int     pr_wstat;         /* if zombie, the wait() status */
    uid_t   pr_uid;           /* real user id */
    uid_t   pr_euid;          /* effective user id */
    gid_t   pr_gid;           /* real group id */
    gid_t   pr_egid;          /* effective group id */
    pid_t   pr_pid;           /* process id */
    pid_t   pr_ppid;          /* process id of parent */
    pid_t   pr_pgrp;          /* pid of process group leader */
    pid_t   pr_sid;           /* session id */
    caddr_t pr_addr;          /* physical address of process */
    long    pr_size;          /* size of process image in pages */
    long    pr_rssize;        /* resident set size in pages */
    u_long  pr_bysize;        /* size of process image in bytes */
    u_long  pr_byrssize;      /* resident set size in bytes */
    caddr_t pr_wchan;         /* wait addr for sleeping process */
    short   pr_syscall;       /* system call number (if in syscall) */
    id_t    pr_aslwpid;       /* lwp id of the aslwp; zero if no aslwp */
    timestruc_t pr_start;     /* process start time, sec+nsec since epoch */
    timestruc_t pr_time;      /* usr+sys cpu time for this process */
    timestruc_t pr_ctime;     /* usr+sys cpu time for reaped children */
    long    pr_pri;           /* priority, high value is high priority */
    char    pr_oldpri;        /* pre-SVR4, low value is high priority */
    char    pr_cpu;           /* pre-SVR4, cpu usage for scheduling */
    u_short pr_pctcpu;        /* % of recent cpu time, one or all lwps */
    u_short pr_pctmem;        /* % of system memory used by the process */
    dev_t   pr_ttydev;        /* controlling tty device (PRNODEV if none) */
    char    pr_cname[PRCLSZ]; /* scheduling class name */
    char    pr_fname[PRFNSZ]; /* last component of exec()ed pathname */
    char    pr_psargs[PRARGSZ]; /* initial characters of arg list */
    int     pr_argc;          /* initial argument count */
    char    **pr_argv;        /* initial argument vector */
    char    **pr_envp;        /* initial environment vector */
} prpsinfo_t;

```

Some of the entries in **prpsinfo**, such as **pr_state** and **pr_flag**, are system-specific and should not be expected to retain their meanings across different versions of the operating system. **pr_addr** is a vestige of the past and has no real meaning in current systems.

pr_pctcpu and **pr_pctmem** are 16-bit binary fractions in the range 0.0 to 1.0 with the binary point to the right of the high-order bit (1.0 == 0x8000). When obtained from the process file descriptor, **pr_pctcpu** is the summation over all *lwps* in the process. When

obtained from an lwp file descriptor, it represents just the cpu time used by the *lwp*. On a multi-processor machine, the maximum value of **pr_pctcpu** for one lwp or for a single-threaded process is 1/N, where N is the number of cpus.

PIOCPINFO can be applied to a *zombie* process (one that has terminated but whose parent has not yet performed a **wait(2)** on it).

PIOCNMAP
PIOCMAP

These operations provide information about the memory mappings (virtual address ranges) associated with the traced process.

Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege. **PIOCNMAP** returns, in the *int* addressed by *p*, the number of mappings that are currently active. This can be used to allocate storage for use with the **PIOCMAP** operation, which returns the list of currently active mappings. For **PIOCMAP**, *p* addresses an array of elements of type **prmap_t**; one array element (one structure) is returned for each mapping, plus an additional element containing all zeros to mark the end of the list. The **prmap** structure contains at least the following fields:

```
typedef struct prmap {
    caddr_t   pr_vaddr;    /* Virtual address */
    u_long    pr_size;     /* Size of mapping in bytes */
    u_long    pr_pagesize; /* pagesize in bytes for this mapping */
    off_t     pr_off;     /* Offset into mapped object, if any */
    long      pr_mflags;   /* Protection and attribute flags */
} prmap_t;
```

pr_vaddr is the virtual address of the mapping within the traced process and **pr_size** is its size in bytes. **pr_pagesize** is the size in bytes of virtual memory pages for this mapping. **pr_off** is the offset within the mapped object (if any) to which the virtual address is mapped.

pr_mflags is a bit-mask of protection and attribute flags:

MA_READ	mapping is readable by the traced process
MA_WRITE	mapping is writable by the traced process
MA_EXEC	mapping is executable by the traced process
MA_SHARED	mapping changes are shared by the mapped object
MA_BREAK	mapping is grown by the brk(2) system call (obsolete)
MA_STACK	mapping is grown automatically on stack faults (obsolete)

A contiguous area of the address space having the same underlying mapped object may appear as multiple mappings due to varying read/write/execute/shared attributes. The underlying mapped object does not change over the range of a single mapping. An I/O operation to a mapping marked **MA_SHARED** fails if applied at a virtual address not corresponding to a valid page in the underlying mapped object. A write to a **MA_SHARED** mapping that is not marked **MA_WRITE** fails. Reads and writes to private mappings always succeed. Reads and writes to unmapped addresses always fail.

	The <code>MA_BREAK</code> and <code>MA_STACK</code> flags are provided for compatibility with older versions of the system and should not be relied upon. The <code>pr_brkbase</code> , <code>pr_brksize</code> , <code>pr_stkbase</code> and <code>pr_stksize</code> members of the <code>prstatus</code> structure should be used instead.
PIOCOPENM	<p>The return value <i>retval</i> provides a read-only file descriptor for a mapped object associated with the traced process. If <i>p</i> is zero the traced process's executable file is found. This enables a debugger to find the object file symbol table without having to know the path name of the executable file. If <i>p</i> is non-zero it points to a <code>caddr_t</code> containing a virtual address within the traced process and the mapped object, if any, associated with that address is found; this can be used to get a file descriptor for a shared library that is attached to the process. On error (invalid address or no mapped object for the designated address), <code>-1</code> is returned and <code>errno</code> is set to <code>EINVAL</code>.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege.</p>
PIOCCRED	<p>Fetch the set of credentials associated with the process. <i>p</i> points to an instance of <code>prcred_t</code> which is filled by the operation.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege.</p> <p>The <code>prcred</code> structure contains at least the following fields:</p> <pre>typedef struct prcred { uid_t pr_euid; /* Effective user id */ uid_t pr_ruid; /* Real user id */ uid_t pr_suid; /* Saved user id (from exec) */ gid_t pr_egid; /* Effective group id */ gid_t pr_rgid; /* Real group id */ gid_t pr_sgid; /* Saved group id (from exec) */ u_int pr_ngroups; /* Number of supplementary groups */ } prcred_t;</pre>
PIOCGROUPS	<p>Fetch the set of supplementary group IDs associated with the process. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege. <i>p</i> points to an array of elements of type <code>gid_t</code>, which will be filled by the operation. <code>PIOCCRED</code> can be applied beforehand to determine the number of groups (<code>pr_ngroups</code>) that will be returned and the amount of storage that should be allocated to hold them.</p>
PIOCNAUXV PIOCAUXV	<p>These operations provide values of entries in the aux vector that is passed by the operating system as startup information to the dynamic loader. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege. <code>PIOCNAUXV</code> returns, in the <i>int</i> addressed by <i>p</i>, the number of available aux vector entries. This can be used to allocate storage for use with the <code>PIOCAUXV</code> operation, which returns the initial values of the process's aux vector in an array of <code>auxv_t</code> structures addressed by <i>p</i> (see <code><sys/auxv.h></code>).</p>

PIOCUSAGE

When applied to the process file descriptor, **PIOCUSAGE** returns the process usage information; when applied to an lwp file descriptor, it returns usage information for the specific *lwp*. Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege. *p* points to a **prusage** structure which is filled by the operation. The **prusage** structure contains at least the following fields:

```
typedef struct prusage {
    id_t      pr_lwpid;    /* lwp id. 0: process or defunct */
    u_long    pr_count;    /* number of contributing lwps */
    timestruc_t pr_stamp; /* current time stamp */
    timestruc_t pr_create; /* process/lwp creation time stamp */
    timestruc_t pr_term;   /* process/lwp termination time stamp */
    timestruc_t pr_rtime;  /* total lwp real (elapsed) time */
    timestruc_t pr_utime;  /* user level CPU time */
    timestruc_t pr_stime;  /* system call CPU time */
    timestruc_t pr_ttime;  /* other system trap CPU time */
    timestruc_t pr_tftime; /* text page fault sleep time */
    timestruc_t pr_dftime; /* data page fault sleep time */
    timestruc_t pr_kftime; /* kernel page fault sleep time */
    timestruc_t pr_ltime;  /* user lock wait sleep time */
    timestruc_t pr_slptime; /* all other sleep time */
    timestruc_t pr_wtime;  /* wait-cpu (latency) time */
    timestruc_t pr_stoptime; /* stopped time */
    u_long    pr_minf;     /* minor page faults */
    u_long    pr_majf;     /* major page faults */
    u_long    pr_nswap;    /* swaps */
    u_long    pr_inblk;    /* input blocks */
    u_long    pr_oublk;    /* output blocks */
    u_long    pr_msnd;     /* messages sent */
    u_long    pr_mrcv;     /* messages received */
    u_long    pr_sigs;     /* signals received */
    u_long    pr_vctx;     /* voluntary context switches */
    u_long    pr_ictx;     /* involuntary context switches */
    u_long    pr_sysc;     /* system calls */
    u_long    pr_ioch;     /* chars read and written */
} prusage_t;
```

PIOCUSAGE can be applied to a *zombie* process (see **PIOCPSINFO**).

Applying **PIOCUSAGE** to a process that does not have microstate accounting enabled will enable microstate accounting and return an estimate of times spent in the various states up to this point. Further invocations of **PIOCUSAGE** will yield accurate microstate time accounting from this point. To disable microstate accounting, use **PIOCRESET** with the **PR_MSACCT** flag.

PIOCLUSAGE

The **PIOCLUSAGE** operation fills in an array of **prusage** structures addressed by *p*, one element for each *lwp* in the process plus an additional element at the beginning that contains the summation over all defunct *lwps* (*lwps* that once existed but no longer exist in the process).

Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege.

Excluding the **pr_lwpid**, **pr_tstamp**, **pr_create** and **pr_term** entries, the entry-by-entry summation over all these structures is the definition of the process usage information.

PIOCLUSAGE can be applied to a *zombie* process (see **PIOCPSINFO**).

PIOCLUSAGE enables microstate accounting as described above for **PIOCUSAGE**.

PIOCOPENPD

The return value *retval* provides a read-only file descriptor for a “page data file”, enabling tracking of address space references and modifications on a per-page basis.

Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege.

A **read(2)** of the page data file descriptor returns structured page data and atomically clears the page data maintained for the file by the system. That is to say, each read returns data collected since the last read; the first read returns data collected since the file was opened. When the call completes, the read buffer contains the following structure as its header and thereafter contains a number of section header structures and associated byte arrays that must be accessed by walking linearly through the buffer.

```
typedef struct prpageheader {
    timestruc_t    pr_tstamp;    /* real time stamp */
    u_long         pr_nmap;      /* number of address space mappings */
    u_long         pr_npage;     /* total number of pages */
} prpageheader_t;
```

The header is followed by **pr_nmap prsmap** structures and associated data arrays. The **prsmmap** structure contains at least the following elements.

```
typedef struct prsmap {
    caddr_t        pr_vaddr;     /* virtual address */
    u_long         pr_npage;     /* number of pages in mapping */
    off_t          pr_off;       /* offset into mapped object, if any */
    u_long         pr_mflags;    /* protection and attribute flags */
    u_long         pr_pagesize;  /* pagesize in bytes for this mapping */
} prsmap_t;
```

Each section header is followed by **pr_npage** bytes, one byte for each page in the mapping, plus enough null bytes at the end so that the next **prsmmap** structure begins on a long-aligned boundary. Each data byte may contain these flags:

```
PG_REFERENCED    page has been referenced
PG_MODIFIED      page has been modified
```

PIOCOPENPD	<p>If the read buffer is not large enough to contain all of the page data, the read fails with E2BIG and the page data is not cleared. The required size of the read buffer can be determined through <code>fstat(2TSOL)</code>. Application of <code>lseek(2TSOL)</code> to the page data file descriptor is ineffective. Closing the page data file descriptor terminates the system overhead associated with collecting the data.</p> <p>More than one page data file descriptor for the same process can be opened, up to a system-imposed limit per traced process. A read of one does not affect the data being collected by the system for the others.</p> <p>The <code>PIOCOPENPD</code> operation returns -1 on failure. Reasons for failure are application to a system process (EINVAL) or too many page data file descriptors were requested (ENOMEM).</p>
PIOCGWIN	<p>This operation applies only to SPARC machines. <i>p</i> is a pointer to a <code>gwindows_t</code> structure, defined in <code><sys/reg.h></code>, that is filled with the contents of those SPARC register windows that could not be stored on the stack when the <i>lwp</i> stopped.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege.</p> <p>If information label floating is configured for the system, the information label of the file system object floats unless the <code>PRIV_FILE_NOFLOAT</code> privilege is asserted. Conditions under which register windows are not stored on the stack are: the stack pointer refers to nonexistent process memory or the stack pointer is improperly aligned. If the specific or chosen <i>lwp</i> is not stopped, the operation returns undefined values.</p>
PIOCGETPR PIOCGETU	<p>These operations copy, respectively, the traced process's <i>proc</i> structure and <i>user</i> structure into the buffer addressed by <i>p</i>. They are provided for completeness but it should be unnecessary to access either of these structures directly since relevant status information is available through other control operations. Their use is discouraged because a program making use of them is tied to a particular version of the operating system.</p> <p>Mandatory read access to the file system object is required. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege.</p> <p><code>PIOCGETPR</code> can be applied to a <i>zombie</i> process (see <code>PIOCPSINFO</code>).</p>
PIOCGXREGSIZE	<p>This operation gets the size, in bytes, of the extra state registers referenced by <code>PIOCGXREG</code> and <code>PIOCSXREG</code>. The extra state register set size is architecture dependent. An error (EINVAL) is returned if the system does not support extra state registers. Mandatory read access is required to the file system object. To override this restriction, the calling process may assert the <code>PRIV_FILE_MAC_READ</code> privilege.</p>
PIOCGXREG *PIOCSXREG	<p>These operations get and set, respectively, the extra state registers for the specific or chosen <i>lwp</i> into or out of a structure addressed by <i>p</i>; the structure has type <code>prxregset_t</code> and is architecture dependent. An error (EINVAL) is returned if the system does not support extra state registers. <code>PIOCSXREG</code> fails (EBUSY) if the <i>lwp</i> is not stopped on an event of interest. If the <i>lwp</i> is not stopped, the register values returned by <code>PIOCGXREG</code> are undefined.</p>

For **PIOCGXREG** mandatory read access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_READ** privilege.

For **PIOCSXREG** mandatory write access to the file system object is required. To override this restriction, the calling process may assert the **PRIV_FILE_MAC_WRITE** privilege.

FILES

/proc	directory (list of processes)
/proc/nnnnn	process file

SUMMARY OF TRUSTED SOLARIS CHANGES

Appropriate privilege is required to override mandatory access checks. Discretionary access checks have already been performed when the object was opened.

If information label floating is turned on for the system, the information label of the calling process and object float as follows:

For write operations, the information label of the calling process floats the information label of the object unless the **PRIV_FILE_NOFLOAT** privilege is asserted.

For read operations, the information label of the object floats the information label of the calling process unless the **PRIV_PROC_NOFLOAT** privilege is asserted.

An **open(2TSOL)** by a process with the **PRIV_SYS_DEVICES** privilege that does not specify **O_EXCL** succeeds even an exclusive write open is in effect on the file.

A traced process's **nice(2TSOL)** priority is incremented by the amount contained in the *int* addressed by *p* when the process asserts the **PRIV_SYS_CONFIG** privilege. This operation is meaningful only when applied to a process in the time-sharing scheduling class.

The following IOCTLS added to Trusted Solaris are used to get information about the security attributes of a process: **PIOCLABEL**, **PIOCCLEAR**, **PIOCEPRIV**, **PIOCIPRIV**, **PIOCPPRIV**, **PIOCSPRIV**, **PIOCATTR**, **PIOCAPSA**, and **PIOCTCRED**.

SEE ALSO

ps(1), **alarm(2)**, **brk(2)**, **close(2)**, **exec(2TSOL)**, **getpattr(2TSOL)**, **getaudit(2TSOL)**, **fork(2TSOL)**, **fstat(2TSOL)**, **ioctl(2)**, **kill(2TSOL)**, **lseek(2TSOL)**, **nice(2TSOL)**, **open(2TSOL)**, **poll(2)**, **ptrace(2)**, **read(2TSOL)**, **sigaction(2)**, **sigalstack(2)**, **vfork(2TSOL)**, **wait(2)**, **write(2TSOL)**, **signal(3C)**, **siginfo(5)**, **signal(5)**, **ucontext(5)**

DIAGNOSTICS

Errors that can occur in addition to the errors normally associated with file system access:

EACCES	The calling process does not have mandatory read access to the file system object. To override this restriction, the calling process may assert the PRIV_FILE_MAC_READ privilege.
ENOENT	The traced process or lwp has terminated after being opened.
EIO	I/O was attempted at an illegal address in the traced process.
EBADF	An I/O or ioctl operation requiring write access was attempted on a file descriptor not open for writing.
EBUSY	PIOCSTOP or PIOCWSTOP was applied to a system process; an exclusive open(2TSOL) was attempted on a process file already open for writing; an open9(2TSOL) for writing was attempted and an exclusive open is in effect on the process file; PIOCRUN , PIOCSREG , PIOCSXREG , or

	PIOCSFPREG was applied to a process or lwp not stopped on an event of interest; an attempt was made to mount /proc when it is already mounted.
EPERM	Someone other than the process asserting the PRIV_SYS_CONFIG privilege attempted to better a process's priority by issuing PIOCNICE .
ENOSYS	An attempt was made to perform an unsupported operation (such as create, remove, link, or unlink) on an entry in /proc .
EFAULT	An I/O or ioctl request referred to an invalid address in the controlling process.
EINVAL	In general this means that some invalid argument was supplied to a system call. The list of conditions eliciting this error includes: the ioctl code is undefined; an ioctl operation was issued on a file descriptor referring to the /proc directory; the PRSTEP option of the PIOCRUN operation was used on an implementation that does not support single-stepping; an out-of-range signal number was specified with PIOCSSIG , PIOCKILL , or PIOCUNKILL ; SIGKILL was specified with PIOCUNKILL ; an illegal virtual address was specified in a PIOCOPENM request; PIOCGFPREG or PIOCSFPREG was issued on a system that does not support floating-point operations.
ENOMEM	The system-imposed limit on the number of page data file descriptors was reached on a PIOCOPENPD request.
E2BIG	Data to be returned in a read(2TSOL) of the page data file exceeds the size of the read buffer provided by the caller.
EINTR	A signal was received by the controlling process while waiting for the traced process or lwp to stop via PIOCSTOP or PIOCWSTOP .
EAGAIN	The traced process has performed an exec(2TSOL) of a setuid/setgid object file or of an object file that it cannot read; all further operations on the process or lwp file descriptor (except close(2)) elicit this error.

NOTES

Each **ioctl** operation is guaranteed to be atomic with respect to the traced process, except when applied to a system process. I/O to the traced process's memory is not guaranteed to be atomic unless all the *lwps* in the process are stopped, the memory is not shared by another running process, and the memory is not the target of asynchronous I/O.

For security reasons, a process must have both discretionary and mandatory read and write access to a traced process as well as discretionary and mandatory read access to the process's executable file. Files corresponding to **setuid**, **setgid**, and privileged processes (those with permitted privileges) can only be opened by a process which in addition to having discretionary and mandatory read access has asserted the **PRIV_PROC_OWNER** privilege. If a traced process performs an **exec(2TSOL)**, the open process or lwp file descriptor will become invalid if the new object file cannot be read. If a traced process performs an **exec(2TSOL)**, the open process or lwp file descriptor will become invalid if the new object file is either a **setuid/setgid** object file, or will have the use or privileges upon execution. The tracing process may assert the **PRIV_PROC_OWNER** privilege to

override this restriction. Any operation performed on an invalid file descriptor, except **close(2)**, fails with **EAGAIN**. In this situation, if any tracing flags are set and the process file descriptor or any lwp file descriptor is open for writing, the process will have been directed to stop and its run-on-last-close flag will have been set (see **PIOCSET**). This enables a controlling process (if it has permission) to reopen the process file to get a new valid file descriptor, close the invalid file descriptors, and proceed. Just closing the invalid file descriptors causes the traced process to resume execution with no tracing flags set. Any process not currently open for writing via **/proc** but that has left-over tracing flags from a previous open and that *execs* a *setuid/setgid* or unreadable object file will not be stopped but will have all its tracing flags cleared.

To wait for one or more of a set of processes or *lwps* to stop or terminate, **/proc** file descriptors can be used in a **poll(2)** system call. When requested and returned, the polling event **POLLPRI** indicates that the process or lwp stopped on an event of interest. Although they cannot be requested, the polling events **POLLHUP**, **POLLERR** and **POLLNVAL** may be returned. **POLLHUP** indicates that the process or lwp has terminated. **POLLERR** indicates that the file descriptor has become invalid. **POLLNVAL** is returned immediately if **POLLPRI** is requested on a file descriptor referring to a system process (see **PIOCSTOP**). The requested events may be empty to wait simply for termination.

Descriptions of structures in this document include only interesting structure elements, not filler and padding fields, and may show elements out of order for descriptive clarity. The actual structure definitions are contained in **<sys/procfs.h>**.

The **PIOCLSTATUS**, **PIOCLWPIDS**, **PIOCLDT**, **PIOCMAP**, **PIOCGROUPS**, and **PIOCLUSAGE** operations return arrays whose actual sizes can only be known through previously-applied operations. Applying these operations to a process that is not stopped runs the risk of overrunning the buffer passed to the system.

For reasons of symmetry and efficiency there are more control operations than strictly necessary.

BUGS

The types **gregset_t** and **fpregset_t** defined in **<sys/reg.h>** are similar to but not the same as the types **prgregset_t** and **prfpregset_t** defined in **<sys/procfs.h>**.

NAME	resolv.conf – configuration file for name server routines
DESCRIPTION	<p>The resolver configuration file contains information that is read by the resolver routines the first time they are invoked in a process. The file is designed to be human readable and contains a list of keyword-value pairs that provide various types of resolver information of the form:</p> <p style="text-align: center;"><i>keyword value</i></p> <p>The different configuration options are:</p> <p>nameserver <i>address</i> The Internet address (in dot (‘.’) notation) of a name server that the resolver should query. At least one name server should be listed. Up to MAXNS (currently three) name servers may be listed, in that case the resolver library queries tries them in the order listed. The algorithm used is to try a name server, and if the query times out, try the next until out of name servers, then repeat trying all the name servers until a maximum number of retries are made.</p> <p>domain <i>name</i> The default domain to append to names that do not have a dot (‘.’) in them. <i>name</i> should be immediately terminated by a newline character.</p> <p>The keyword-value pair must appear on a single line, and the keyword (for instance, nameserver) must start the line. The value follows the keyword, separated by white space.</p> <p>To protect /etc/resolv.conf from unauthorized modification, it must have a sensitivity label of ADMIN_LOW. The DNS name servers specified in these files can reside on either Trusted Solaris hosts or non-trusted hosts. Administrators are advised to configure only DNS name servers on Trusted Solaris hosts in the /etc/resolv.conf file.</p>
SUMMARY OF TRUSTED SOLARIS CHANGES	/etc/resolv.conf must have a sensitivity label of ADMIN_LOW .
FILES	/etc/resolv.conf
SEE ALSO	in.named (1MTSOL), gethostbyname (3N), resolver (3NTSOL)

NAME	rmtab – remote mounted file system table
SYNOPSIS	/etc/rmtab
DESCRIPTION	<p>rmtab contains a table of filesystems that are remotely mounted by NFS clients. This file is maintained by mountd(1MITSOL), the mount daemon. The data in this file should be obtained only from mountd(1MITSOL) using the MOUNTPROC_DUMP remote procedure call.</p> <p>The file contains a line of information for each remotely mounted filesystem. There are a number of lines of the form:</p> <p style="text-align: center;"><i>hostname:fsname</i></p> <p>The mount daemon adds an entry for any client that successfully executes a mount request and deletes the appropriate entries for an unmount request.</p> <p>Lines beginning with a hash ('#') are commented out. These lines are removed from the file by mountd(1MITSOL) when it first starts up. Stale entries may accumulate for clients that crash without sending an unmount request.</p>
SUMMARY OF TRUSTED SOLARIS CHANGES	The /etc/rmtab file must have a sensitivity label of ADMIN_LOW and be owned by UID 0.
FILES	/etc/rmtab
SEE ALSO	mountd (1MITSOL), showmount (1MITSOL)

NAME	sel_config – Selection rules for copy, cut, paste, drag and drop operations																
SYNOPSIS	/usr/dt/config/sel_config																
AVAILABILITY	SUNWtsolu																
DESCRIPTION	<p>The sel_config file specifies how the system behaves when a user performs cut-and-paste, copy-and-paste, and drag-and-drop operations on data between windows that have different sensitivity labels, information labels, or both. There are three types of entries in this file: automatic confirmation, hidden information label action, and automatic reply.</p>																
Automatic Confirmation	<p>This type of entry specifies whether a confirmation window (the selection confirmer) displays. There are typically 16 entries, each a combination of SLs, ILs, or both. Each entry has the form:</p> <p style="padding-left: 40px;"><i>relationship: confirmation IL_source</i></p> <p><i>relationship</i> identifies the result of comparing the selected data's source and destination windows' SLs or ILs. There are 16 allowed values; each value is a period-separated pair of one or two of the following values:</p> <table border="0" style="margin-left: 40px;"> <tr> <td>upgradesl, upgradeil</td> <td>The source window's label is less than the destination window's label.</td> </tr> <tr> <td>downgradesl, downgradeil</td> <td>The source window's label is higher than the destination window's label.</td> </tr> <tr> <td>equalsl, equalil</td> <td>The source and destination windows' labels are equal.</td> </tr> <tr> <td>disjointsl, disjointil</td> <td>The source and destination windows' labels are disjoint (neither dominates the other).</td> </tr> </table> <p><i>confirmation</i> specifies whether to perform automatic confirmation. Allowed values are:</p> <table border="0" style="margin-left: 40px;"> <tr> <td>y</td> <td>Use automatic confirmation (that is, do not display the selection confirmer window).</td> </tr> <tr> <td>n</td> <td>Use manual confirmation (that is, display the selection confirmer window).</td> </tr> </table> <p><i>IL_source</i> is used for IL comparisons and specifies how to handle the IL of the destination window. Allowed values are:</p> <table border="0" style="margin-left: 40px;"> <tr> <td>s</td> <td>Use the IL of the destination window.</td> </tr> <tr> <td>d</td> <td>Float the IL of the destination window with the IL of the source window.</td> </tr> </table>	upgradesl, upgradeil	The source window's label is less than the destination window's label.	downgradesl, downgradeil	The source window's label is higher than the destination window's label.	equalsl, equalil	The source and destination windows' labels are equal.	disjointsl, disjointil	The source and destination windows' labels are disjoint (neither dominates the other).	y	Use automatic confirmation (that is, do not display the selection confirmer window).	n	Use manual confirmation (that is, display the selection confirmer window).	s	Use the IL of the destination window.	d	Float the IL of the destination window with the IL of the source window.
upgradesl, upgradeil	The source window's label is less than the destination window's label.																
downgradesl, downgradeil	The source window's label is higher than the destination window's label.																
equalsl, equalil	The source and destination windows' labels are equal.																
disjointsl, disjointil	The source and destination windows' labels are disjoint (neither dominates the other).																
y	Use automatic confirmation (that is, do not display the selection confirmer window).																
n	Use manual confirmation (that is, display the selection confirmer window).																
s	Use the IL of the destination window.																
d	Float the IL of the destination window with the IL of the source window.																
Hidden Information Label Action	<p>This entry specifies how to handle invalid ILs when ILs are not visible to the user. The action specified here depends on whether automatic confirmation is set up for the action. If automatic confirmation is specified for the user's operation and applying the <i>IL_source</i> would result in a valid IL, that valid IL is used. If the result would be an invalid IL, the action specified by this entry is used.</p>																

If manual confirmation is specified for the user's operation and applying the *IL_source* would result in an invalid IL, the action specified by this entry is used.

Each entry as the form:

hidden_il_action: *value*

There are three allowed values:

- c** Display a confirmer that shows ILs.
- d** Use the IL of the destination. For **sel_mgr** applications, use the IL of the destination window. For **dtfile** applications, use the IL of the destination directory.
- l** Set the IL to ADMIN_LOW.

The default value is **c**.

Automatic Reply

This set of entries provides a means to reduce the number of confirmations that are required of the user, since a single user operation may involve several flows of information between the source and destination windows.

There must be one entry of this form:

autoreply: *value*

If *value* is **y** (for yes), then the remaining entries of the set are used as attributes for the selection data (rather than the actual contents) to complete the operation without confirmation. If *value* is **n** (for no), then the remaining entries are ignored.

Defaults can be specified for any Type field that appears in the Confirmer window.

Below are some examples entries for defaults.

replytype: TARGETS
replytype: Pixel Sets
replytype: LENGTH
replytype: Type Of Monitor

The **TARGETS** entry, when used, returns the list of target atoms that are supported by the source window. The **Pixel Sets** and **Type Of Monitor** entries, are used for animation during a drag-and-drop operation. The **LENGTH** entry specifies the number of bytes in the selection.

NOTE

Even if automatic confirmation is enabled (that is, *confirmation* is **y**), if the source and destination windows have different UIDs, the selection confirmer will display.

SEE ALSO

Trusted Solaris administrator's document set

NAME	sharetab – shared file system table										
DESCRIPTION	<p>sharetab resides in directory /etc/dfs and contains a table of local resources shared by the share command.</p> <p>Each line of the file consists of the following fields:</p> <p style="padding-left: 40px;"><i>pathname resource fstype specific_options description</i></p> <p>where</p> <table border="0" style="padding-left: 40px;"> <tr> <td><i>pathname</i></td> <td>Indicate the path name of the shared resource.</td> </tr> <tr> <td><i>resource</i></td> <td>Indicate the symbolic name by which remote systems can access the resource.</td> </tr> <tr> <td><i>fstype</i></td> <td>Indicate the file system type of the shared resource.</td> </tr> <tr> <td><i>specific_options</i></td> <td>Indicate file-system-type-specific options that were given to the share command when the resource was shared.</td> </tr> <tr> <td><i>description</i></td> <td>Describe the shared resource provided by the system administrator when the resource was shared.</td> </tr> </table>	<i>pathname</i>	Indicate the path name of the shared resource.	<i>resource</i>	Indicate the symbolic name by which remote systems can access the resource.	<i>fstype</i>	Indicate the file system type of the shared resource.	<i>specific_options</i>	Indicate file-system-type-specific options that were given to the share command when the resource was shared.	<i>description</i>	Describe the shared resource provided by the system administrator when the resource was shared.
<i>pathname</i>	Indicate the path name of the shared resource.										
<i>resource</i>	Indicate the symbolic name by which remote systems can access the resource.										
<i>fstype</i>	Indicate the file system type of the shared resource.										
<i>specific_options</i>	Indicate file-system-type-specific options that were given to the share command when the resource was shared.										
<i>description</i>	Describe the shared resource provided by the system administrator when the resource was shared.										
SUMMARY OF TRUSTED SOLARIS CHANGES	The /etc/dfs/sharetab file must have a sensitivity label of ADMIN_LOW and be owned by UID 0.										
FILES	/etc/dfs/sharetab										
SEE ALSO	share(1M)										

NAME	tndlog – Log of tnd debugging information
SYNOPSIS	/var/tsol/tndlog
DESCRIPTION	<p>/var/tsol/tndlog is the default log file for debugging tnd(1MTSOL). This file contains one record for each debugging message. Each record contains the debugging message and time.</p> <p>tndlog is an ASCII file. Each field within each entry is separated from the next by a colon. Each entry is separated from the next by a new line.</p> <p>By default, tndlog does not exist, so no logging is done. To enable logging, tnd must be started with a debug level, or tnctl(1MTSOL) must be issued to turn on debugging. The log file can be created either by tnd or by administrators running with appropriate privileges. /var/tsol/tndlog should have a sensitivity label of ADMIN_LOW with permission bits 200, owner root, and group sys.</p>
FILES	/var/tsol/tndlog
SEE ALSO	tnd (1MTSOL)

NAME	tnidb – Trusted network interface-control database
SYNOPSIS	<code>/etc/security/tsol/tnidb</code>
AVAILABILITY	SUNWtsolr
DESCRIPTION	<p>The tnidb database specifies the accreditation range and default security attributes for each network interface. Network traffic is not permitted for any interface, including the loopback interface, unless the interface has a valid entry.</p> <p>Each entry in the interface database consists of one long line, with fields of the entry separated by semicolons (;):</p> <pre style="margin-left: 40px;"><i>interface_name:field1;field2;field3;fieldn;</i></pre> <p>A pound sign (#) as the first character of a line indicates a comment line, which is ignored. Each entry consists of a line of this form:</p> <pre style="margin-left: 40px;"><i>interface_name:min_sl=value;max_sl=value;def_label=value;def_cl=value; def_uid=value;def_gid=value;forced_privs=value;</i></pre> <p>NOTE: The width of this man page prevents showing the foregoing entry on a single line. However, each entry in the database <i>must</i> be a single line.</p> <p>The first field for each entry is the interface name. Each entry must contain valid specifications for the accreditation range of the interface for all enforceable security attributes. All fields are mandatory; each entry contains these fields:</p> <p>min_sl, max_sl Specify the accreditation range of the interface. Only packets with a sensitivity label within the specified accreditation range are allowed into or out of the interface. For a configuration that allows for traffic at all labels, the range should be ADMIN_LOW (in hex) to ADMIN_HIGH (in hex).</p> <p>def_label Apply this default label to a packet received from an approved remote host that does not support mandatory access control. Under these conditions, all packets imported from the interface that are not labeled with a sensitivity label or information label are assigned this default label. If an information label is not specified, ADMIN_LOW will be used.</p> <p>def_cl Apply this default clearance to a packet received from an approved remote host that does not support mandatory access control.</p> <p>def_uid, def_gid Apply this default effective user ID and default effective group ID to a packet. Specify the def_uid and def_gid fields by using the user ID name and group ID names, respectively.</p> <p>forced_privs Define the effective privileges to be applied to the incoming packet received from a host that does not support privileges. The format of the privilege set is:</p> <pre style="margin-left: 40px;">forced_privs=priv[,priv][...] none empty all</pre> <p>where</p>

NAME	tnrhdb – Trusted network remote-host database
AVAILABILITY	SUNWtsolr
DESCRIPTION	<p>The tnrhdb database specifies which remote-host template to use for each host, including the local host, in the distributed system. tnrhdb works together with the tnrhtp(4TSOL) database in allowing the administrator to establish the security and network-accreditation attributes for each host. The trusted-network software uses a network “hierarchical fallback” mechanism in looking for a tnrhdb entry for a host. The software looks first for an entry specific to the host; if it does not find one, the software falls back to searching for a matching class-C network entry, and then a class-B entry, a class-A entry, and finally a wildcard entry (IP address 0.0.0.0). In the search for a class-C, a class-B, and a class-A entry, the network environment is assumed to be subnetted on a natural (octet) boundary. (Netmasks will be supported in a future release.) If a host’s IP address cannot be matched to some entry in the tnrhdb database, communication with the host is not permitted.</p> <p>Each entry consists of a line of this form:</p> <p style="text-align: center;"><i>IP_address:template_name</i></p> <p><i>IP_address</i> This field is the IP address of the host or network that has the security properties specified by the <i>template_name</i> defined in the tnrhtp database. IP addresses are specified in the standard Internet decimal dotted notation. The IP addresses of the hosts and networks should match the IP addresses used for the hosts in the hosts(4) database.</p> <p><i>template_name</i> This value must be a valid template name in the tnrhtp(4TSOL) database. [See man pages for tnrhtp(4TSOL) for information on the security attributes.] More than one IP address can use the same template. If this database is modified while the network is up, the changes do not take effect until after tnctl(1MTSOL) is used to update the remote-host entries. Administrators are allowed to add new entries and modify existing entries while network is up.</p> <p>Errors in the format of this file can be detected by running tnchkdb(1MTSOL), which should be run every time the database is modified or created. [Refer to the tnchkdb man page for more information.]</p> <p>/etc/security/tsol/tnrhdb should have a sensitivity label of ADMIN_LOW with permission bits 444, owner sys, and group sys .</p>

EXAMPLES	<p>The following example shows a host that uses template unlabeled1, a host that uses template tsol, a subnet that uses template tsol, a subnet that uses template unlabeled2; and every other host uses the default template specified in the wildcard entry.</p> <pre># # Assume that templates default, tsol, unlabeled1, and unlabeled2 are # defined in the tnrhdb (4TSOL) database. # # first one is the localhost entry 192.110.120.6:tsol 192.110.120.0:tsol 192.110.120.7:unlabeled1 192.110.121.0:unlabeled2 0.0.0.0:default</pre>
FILES	<pre>/etc/security/tsol/tnrhdb</pre>
SEE ALSO	<pre>tnd(1MTSOL), tnchfdb(1MTSOL), tnctl(1MTSOL), tsol_getrhbyaddr(3NTSOL), tnidb(4TSOL), tnrhdb(4TSOL)</pre>
WARNING	<p>For proper functioning, the primary host name must point to templates that have min_sl = ADMIN_LOW (in hex) and max_sl = ADMIN_HIGH (in hex).</p> <p>Changing a template while the network is up can change the security view of an undetermined number of hosts.</p>
NOTES	<p>The administrator may wish to make one tnrhdb entry for each host running Trusted Solaris 2.x and make one subnet entry that applies to all unlabeled hosts that have the same security attributes. Then, the administrator may make a separate entry for each host that must be assigned a different set of security attributes.</p>

NAME	tnrhttp – Trusted network remote-host templates
SYNOPSIS	<code>/etc/tnrhttp</code>
AVAILABILITY	SUNWtsolr
DESCRIPTION	<p>The tnrhttp database of templates is specified by the administrator for convenience when assigning accreditation and security attributes for each host in the distributed system, including the local host and network. tnrhttp works together with tnrhdb(4TSOL); IP addresses in tnrhdb can be assigned only to templates defined in the tnrhttp database. The administrator should run tnchkdb(1MTSOL) to check the syntax after each modification to the tnrhttp database.</p> <p>Each entry in the interface database must be formed as one long line, with fields of the entry separated by semicolons (;):</p> <p style="padding-left: 40px;"><i>template_name:field1;field2;field3;fieldn;</i></p> <p>A pound sign (#) as the first character of a line indicates a comment line, which is ignored.</p> <p>Each entry consists of a line of this form:</p> <p style="padding-left: 40px;"><i>template_name:field_name=value;[field_name=value; ...]</i></p> <p>Six types of hosts are currently supported: unlabeled, sun_tsol, ripso, cipso, tsix, and msix.</p> <p>All fields of a particular host_type are mandatory even if no value is set other than none. If this database is modified while the network is up, the changes do not take effect immediately unless tnctl(1MTSOL) is used to update the template entries; otherwise, the changes take effect when next polled by the trusted network daemon, tn(1MTSOL). Administrators are allowed to add new templates and modify attributes of existing templates while the network is up.</p> <p><code>/etc/security/tsol/tnrhttp</code> should be at a sensitivity label of ADMIN_LOW with permission bits 444, owner sys, and group sys.</p> <p>When specifying a name for a template, note that only the first 31 characters of the template name are read and interpreted. These characters must be unique. You can use any printable character in a template name except for field delimiters, new-line, or the comment character.</p>
Template for unlabeled Hosts	<p>The template for the unlabeled host type has these fields:</p> <p><i>template_name</i> Specify a name for the template.</p> <p>host_type unlabeled</p> <p>def_label, def_cl, def_uid, def_gid</p> <p style="padding-left: 40px;">Define the default attributes to be applied to incoming data from the remote hosts that do not support these attributes. These defaults override the defaults specified for an interface in the tnidb(4TSOL) database.</p>

To take defaults from **tnidb**(4TSOL), these **def_*** fields must be set equal to **empty**; for example, **def_uid=empty**.

The information label is assumed to be ADMIN_LOW if the information label is not specified in the **def_label** field.

min_sl, max_sl Specify the accreditation range for remote host gateways using this template. The format is the same as that in the **tnidb**(4TSOL) database. All labels are specified in their hex format.

forced_privs Define the effective privileges to be applied to the incoming packet received from a host that does not support privileges. The format of the privilege set is:

forced_privs=priv[,priv][...] | none | empty | all

where

priv The ASCII string (such as **net_mac_read**) for privilege.
(**forced_privs=net_mac_read**)

none Apply no privileges. (**forced_privs=none**)

empty Take the default from **tnidb**(4TSOL). (**forced_privs=empty**)

all Apply all privileges. (**forced_privs=all**)

def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid

Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.

def_audit_auid represents the user's audit ID and can be a positive or negative decimal number.

def_audit_mask consists of two 32-bit success and failure masks, specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in **/etc/security/audit_class**.

def_audit_termid consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.

def_audit_asid represents the audit session ID and can be a positive or negative decimal number.

Template for sun_tsol Hosts

Host type **sun_tsol** has these fields:

template_name Specify a name for the template.

host_type **sun_tsol**

min_sl, max_sl Specify the accreditation range for the remote hosts using this template. The format is the same as that in the **tnidb**(4TSOL) database.

All labels are specified in their hex format.

allowed_privs Limit the effective privilege set for an incoming packet. If a source host associated with this template sends a packet to a destination host, the destination will limit the privilege set of the arrival packet to that specified in this field. The format of the privilege set is:

	allowed_privs=priv[,priv][...] none empty all
	where
	priv The ASCII string (such as net_mac_read) for privilege. (allowed_privs=net_mac_read)
	none Apply no privileges. (allowed_privs=none)
	empty Take the default from tnidb(4TSOL) . (allowed_privs=empty)
	all Apply all privileges. (allowed_privs=all)
ip_label	Provide for IP labeling. These are valid types for ip_label :
	none ripso and cipso options are not used to label data sent to the host. However, ripso and cipso security options may be sent to the host if the host is acting as a gateway.
	ripso For hosts that label their packets with the Revised IP Security Option per RFC 1108. If ripso is selected for a host, the ripso_label field is required.
	cipso For hosts that label their packets according to the Common IP Security Options (Tag Type 1 only) as detailed by the Trusted Systems Interoperability Group (TSIG). If cipso is selected for a host, the cipso_doi field is required.
ripso_label	If ip_label is set to RIPSO , then packets for which the host is the final destination will be labeled with the specified RIPSO label. If the host is configured as a gateway, then the host will be able to route packets with the specified RIPSO label. If ip_label is set to none and ripso_label is set, then the host will be able to forward packets labeled with the specified RIPSO label even though packets addressed to the host will not contain a RIPSO label. Set this field explicitly to empty if no value is to be assigned. These are supported classification level encodings: TOP_SECRET , SECRET , CONFIDENTIAL , UNCLASSIFIED or a hexadecimal representation. These are supported protection authority flags: GENSER , SIOP-ESI , SCI , NSA , DOE , or a hexadecimal representation.
ripso_error	These are the protection authority flags that are used to label ICMP messages generated in response to incoming RIPSO -labeled packets: GENSER , SIOP-ESI , SCI , NSA , DOE , or a hexadecimal representation. The classification level is taken from the ripso_label field. The sender's template is always used when labeling ICMP error messages with RIPSO labels. This field can take multiple values; these must be separated by commas. Set this field explicitly to empty if no value is to be assigned.
cipso_doi	This number is the host's domain of interpretation for CIPSO -labeled packets. The domain of interpretation is a field within the CIPSO security option. If ip_label is set to CIPSO , then packets for which the host is

the final destination will be labeled with a **CIPSO** label containing the specified **cipso_doi**. If the host is configured as a gateway, then the host will be able to route **CIPSO**-labeled packets containing the specified **cipso_doi**. To prevent a gateway from routing **CIPSO**-labeled packets, set this field to **none**; to allow a nongateway machine to send and receive **CIPSO**-labeled packets, set this field to the appropriate DOI.

If **ip_label** is set to **none** and **cipso_doi** is set, then the host will be able to forward **CIPSO**-labeled packets containing the specified **cipso_doi** even though packets addressed to the host will not contain a **CIPSO** label.

Set this field explicitly to **empty** if no value is to be assigned.

def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid

Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.

def_audit_auid represents the user's audit ID and can be a positive or negative decimal number.

def_audit_mask consists of two 32-bit success and failure masks, specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in `/etc/security/audit_class`.

def_audit_termid consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.

def_audit_asid represents the audit session ID and can be a positive or negative decimal number.

**Template for ripso
Hosts**

The template for **ripso** host type is for non-TSOL hosts that label packets with the **RIPSO** basic security option. This template has these fields:

template_name Specify a name for the template.

host_type **ripso**

ripso_label These are supported classification level encodings: **TOP_SECRET**, **SECRET**, **CONFIDENTIAL**, **UNCLASSIFIED**, or a hexadecimal representation. These are supported protection authority flags: **GENSER**, **SIOP-ESI**, **SCI**, **NSA**, **DOE**, or a hexadecimal representation.

ripso_error These are the protection authority flags that are used to label ICMP messages generated in response to incoming **RIPSO**-labeled packets.

This field can take multiple values; these must be separated by commas.

def_label, def_cl, def_uid, def_gid

Define the default attributes to be applied to incoming data from the remote hosts that do not support these attributes. These defaults override the defaults specified for an interface in the **tnidb**(4TSOL) database.

Set this field explicitly to **empty** if no value is to be assigned.

Default labels are not required for the remote-host entry if there are

interface defaults that would be the same for the remote host. The information label is assumed to be ADMIN_LOW if the information label is not specified in the **def_label** field.

- min_sl, max_sl** Specify the accreditation range for the remote host gateway using this template. The format is the same as that in the **tnidb**(4TSOL) database. All labels are specified in their hex format.
- forced_privs** Define the effective privileges to be applied to the incoming packet received from a host that does not support privileges. Having no privileges specified is *not* the same as specifying the word **none**. The format of the privilege set is:
- forced_privs=priv[,priv][...] | none | empty | all**
- where
- priv** The ASCII string (such as **net_mac_read**) for privilege. (**forced_privs=net_mac_read**)
- none** Apply no privileges. (**forced_privs=none**)
- empty** Take the default from **tnidb**(4TSOL). (**forced_privs=empty**)
- all** Apply all privileges. (**forced_privs=all**)
- def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid** Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.
- def_audit_auid** represents the user's audit ID and can be a positive or negative decimal number.
- def_audit_mask** consists of two 32-bit success and failure masks, specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in **/etc/security/audit_class**.
- def_audit_termid** consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.
- def_audit_asid** represents the audit session ID and can be a positive or negative decimal number.

Template for cipso Hosts

The template for **cipso** host type is for hosts that use CIPSO Tag Type 1 to label packets. This template has these fields:

- template_name** Specify a name for the template.
- host_type** **cipso**
- cipso_doi** This number is the host's domain of interpretation for CIPSO-labeled packets.
- min_sl, max_sl** Specify the accreditation range for the remote hosts using this template. The format is the same as that in the **tnidb**(4TSOL) database. All labels are specified in their hex format.
- def_il** Specify the default information label to be applied to incoming data

from remote hosts using this template.

def_cl, def_uid, def_gid

Define the default attributes to be applied to incoming data from the remote hosts that do not support these attributes. These defaults override the defaults specified for an interface in the **tnidb**(4TSOL) database. To take defaults from **tnidb**(4TSOL), these **def_*** fields must be set equal to **empty**, for example, **def_uid=empty**.

forced_privs

Defines the effective privileges to be applied to the incoming packet received from a host that does not support privileges. Having no privileges specified is *not* the same as specifying the word **none**. The format of the privilege set is:

forced_privs=priv[,priv][...] | none | empty | all

where

priv The ASCII string (such as **net_mac_read**) for privilege.
(**forced_privs=net_mac_read**)

none Apply no privileges. (**forced_privs=none**)

empty Take the default from **tnidb**(4TSOL). (**forced_privs=empty**)

all Apply all privileges. (**forced_privs=all**)

def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid

Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.

def_audit_auid represents the user's audit ID and can be a positive or negative decimal number.

def_audit_mask consists of two 32-bit success and failure masks, specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in **/etc/security/audit_class**.

def_audit_termid consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.

def_audit_asid represents the audit session ID and can be a positive or negative decimal number.

Template for tsix Hosts

The template for **tsix** host type is for hosts that use TSIX(RE) 1.1 protocols with token mapping to label packets. This template has these fields:

template_name Specify a name for the template.

host_type **tsix**

min_sl, max_sl Specify the accreditation range for the remote hosts using this template.
host_type

All labels are specified in their hex format.

allowed_privs Limit the effective privilege set for an incoming packet. If a source host associated with this template sends a packet to a destination host, the

destination will limit the privilege set of the arrival packet to that specified in this field. The format of the privilege set is:

allowed_privs=priv[,priv][...] | none | empty | all

where

priv The ASCII string (such as **net_mac_read**) for privilege. (**allowed_privs=net_mac_read**)

none Apply no privileges. (**allowed_privs=none**)

empty Take the default from **tnidb(4TSOL)**. (**allowed_privs=empty**)

all Apply all privileges. (**allowed_privs=all**)

forced_privs Define the effective privileges to be applied to the incoming packet received from a host that is not supplying privileges. Having no privileges specified is *not* the same as specifying the word **none**. The format of the privilege set is:

forced_privs=priv[,priv][...] | none | empty | all

where

priv The ASCII string (such as **net_mac_read**) for privilege. (**forced_privs=net_mac_read**)

none Apply no privileges. (**forced_privs=none**)

empty Take the default from **tnidb(4TSOL)**. (**forced_privs=empty**)

all Apply all privileges. (**forced_privs=all**)

def_label, def_cl, def_uid, def_gid

Define the default attributes to be applied to incoming data from the remote hosts that are not supplying these attributes. These defaults override the defaults specified for an interface in the **tnidb(4TSOL)** database.

If you want to take defaults from **tnidb(4TSOL)**, you must set these **def_*** fields equal to **empty**; for example, **def_uid=empty**;

Default labels are not required for the remote-host entry if there are interface defaults that would be the same for the remote host. The information label is assumed to be **ADMIN_LOW** if the information label is not specified in the **def_label** field.

ip_label

Provide for IP labeling. These are valid types for **ip_label**:

NONE **RIPSO** and **CIPSO** options are not used to label data sent to the host. However, **RIPSO** and **CIPSO** security options may be sent to the host if the host is acting as a gateway.

RIPSO For hosts that label their packets with the Revised IP Security Option per RFC 1108. If **RIPSO** is selected for a host, the **ripso_label** field is required.

CIPSO For hosts that label their packets according to the Common IP Security Options (Tag Type 1 only) as detailed by the

	Trusted Systems Interoperability Group (TSIG). If CIPSO is selected for a host, the cipso_doi field is required.
ripso_label	<p>If ip_label is set to RIPSO, then packets for which the host is the final destination will be labeled with the specified RIPSO label. If the host is configured as a gateway, then the host will be able to route packets with the specified RIPSO label.</p> <p>If ip_label is set to NONE and ripso_label is set, then the host will be able to forward packets labeled with the specified RIPSO label even though packets addressed to the host will not contain a RIPSO label.</p> <p>These are supported classification level encodings: TOP_SECRET, SECRET, CONFIDENTIAL, UNCLASSIFIED. These are supported protection authority flags: GENSER, SIOP-ESI, SCI, NSA, DOE.</p>
ripso_error	<p>These are the protection authority flags that are used to label ICMP messages generated in response to incoming RIPSO-labeled packets. These are supported protection authority flags: GENSER, SIOP-ESI, SCI, NSA, DOE. The classification level is taken from the ripso_label field. The sender's template is always used when labeling ICMP error messages with RIPSO labels.</p> <p>This field can take multiple values; these must be separated by commas. If you do not want to assign a value, you must set this field equal to empty.</p>
cipso_doi	<p>This number is the host's domain of interpretation for CIPSO-labeled packets. The domain of interpretation is a field within the CIPSO security option. If ip_label is set to CIPSO, then packets for which the host is the final destination will be labeled with a CIPSO label containing the specified cipso_doi. If the host is configured as a gateway, then the host will be able to route CIPSO-labeled packets containing the specified cipso_doi. To prevent a gateway from routing CIPSO-labeled packets, set this field to 0; to allow a nongateway machine to send and receive CIPSO-labeled packets, set this field to nonzero.</p> <p>If ip_label is set to NONE and cipso_doi is set, then the host will be able to forward CIPSO-labeled packets containing the specified cipso_doi even though packets addressed to the host will not contain a CIPSO label.</p> <p>If you do not want to assign a value, you must set this field equal to empty.</p>
def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid	<p>Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.</p> <p>def_audit_auid represents the user's audit ID and can be a positive or negative decimal number.</p> <p>def_audit_mask consists of two 32-bit success and failure masks,</p>

specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in `/etc/security/audit_class`.

def_audit_termid consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.

def_audit_asid represents the audit session ID and can be a positive or negative decimal number.

Template for msix Hosts

The template for **msix** host type is for hosts that use **msix** protocol to label packets. For example, the template can be used for interoperating with TS 1.2 hosts. The template has these fields:

template_name Specify a name for the template.

host_type **msix**

min_sl, max_sl Specify the accreditation range for the remote hosts using this template. The format is the same as that in the **tnidb**(4TSOL) database.

All labels are specified in their hex format.

def_label, def_cl, def_uid, def_gid

Define the default attributes to be applied to incoming data from the remote hosts that do not support these attributes. These defaults override the defaults specified for an interface in the **tnidb**(4TSOL) database.

If you want to take defaults from **tnidb**(4TSOL), you must set these **def_*** fields equal to **empty**; for example, **def_uid=empty**;

Default labels are not required for the remote-host entry if there are interface defaults that would be the same for the remote host. The information label is assumed to be `ADMIN_LOW` if the information label is not specified in the **def_label** field.

def_audit_auid, def_audit_mask, def_audit_termid, def_audit_asid

Define the default audit attributes to be applied to incoming data from remote hosts that do not support these attributes.

def_audit_auid represents the user's audit ID and can be a positive or negative decimal number.

def_audit_mask consists of two 32-bit success and failure masks, specified by an 8-byte hexadecimal number. The bits correspond to audit classes defined in `/etc/security/audit_class`.

def_audit_termid consists of a 4-byte port number followed by a 4-byte machine number. These are specified by an 8-byte hexadecimal number.

def_audit_asid represents the audit session ID and can be a positive or negative decimal number.

EXAMPLES

For the sake of clarity on this man page, examples are shown using a continuation character (`\`). In the database file, however, the backslash is not permitted because each entry is made on a single line.

FILES	<code>/etc/security/tsol/tnrhttp</code>
SEE ALSO	<code>tnchkdb</code> (1MTSOL), <code>tnd</code> (1MTSOL), <code>tnctl</code> (1MTSOL), <code>tnidb</code> (4TSOL)
WARNING	<p>Changing a template while the network is up can change the security view of an undetermined number of hosts.</p> <p>Allowing unlabeled hosts onto a Trusted Solaris 2.x network is a security risk to the system. In order to avoid compromising the rest of your network, such hosts must be <i>trusted</i> in the sense that the administrator is certain that they will not be used to compromise the TCB. These hosts should also be physically protected to restrict access to authorized individuals.</p> <p>If you cannot guarantee the physical security of an unlabeled host, it and similar hosts should be isolated on a separate branch of the network. The gateway to the untrusted hosts must be a type <code>sun_tsol</code> host, and its database entries for these untrusted hosts and the interface connected to them must be set to reflect the accreditation of these hosts. This setting allows the gateway to label appropriately all packets received from these hosts and to filter packets bound for them.</p>

NAME	tsolgateways – Static routing configuration file
SYNOPSIS	<i>/etc/tsolgateways</i>
DESCRIPTION	<p>The <i>/etc/tsolgateways</i> file is used to configure static routes for a host. At system start up, if <i>/etc/tsolgateways</i> exists, its contents are used to set up static routes. If <i>/etc/tsolgateways</i> does not exist, <i>/etc/defaultrouter</i> is checked. If <i>/etc/defaultrouter</i> exists, its contents are used to set up static routes. If neither <i>/etc/tsolgateways</i> nor <i>/etc/defaultrouter</i> exists, then the host uses dynamic routing. For dynamic routing, if <i>in.rdisc</i>(1MTSOL) exists, it is used. If the program file <i>/usr/sbin/in.rdisc</i> does not exist, <i>in.routed</i>(1MTSOL) is used.</p> <p>The <i>tsolgateways</i> file differs from the <i>defaultrouter</i> file in several ways. The <i>defaultrouter</i> file can be used only to specify default gateways along with simple metrics that indicate the hop count to the destination. <i>tsolgateways</i> can be used not only to specify default gateways but also to specify gateways for specific hosts and networks. Host and network routing entries in <i>tsolgateways</i> can be specified with an optional <i>emetric</i> that includes security attributes associated with the route. The <i>emetric</i> is used for trusted routing through the shortest route to a destination through gateways whose security level matches the sensitivity of the data being sent out. The <i>emetric</i> is made up of the simple metric plus additional security routing information (SRI). The SRI includes a sensitivity label range and other optional keywords described below.</p> <p>The format of <i>/etc/tsolgateways</i> is shown below:</p> <pre style="margin-left: 40px;">default gateway metric</pre> <p>or</p> <pre style="margin-left: 40px;">[net host] destination gateway metric</pre> <p>or</p> <pre style="margin-left: 40px;">[net host] destination gateway -m emetric</pre> <p>where:</p> <p><i>destination</i> is the IP address of the network.</p> <p><i>gateway</i> is the IP address or hostname of the gateway. If a hostname is used, it must be in the <i>/etc/hosts</i> file. Any destination host(s), network(s), and gateway(s) must be specified with an appropriate host type and template in the local or NIS+ versions of the <i>tnrhdb</i>/<i>tnrhtp</i> databases.</p> <p><i>metric</i> is an integer representing the number of hops to the destination network.</p> <p><i>emetric</i> combines the metric and the SRI of a route, as described below.</p> <p>The first form uses the default keyword to specify a default gateway through which packets are routed if the destination does not match another route specified in the file. If no default is specified and no match can be found among the host or network entries, the packet is dropped.</p>

The second form uses either the **net** or **host** keywords to set up a route to a specific network or host using a simple metric.

The third form is like the second form but it uses the **-m** option to specify the emetric. The emetric is specified in the following form (with the single line shown as two for readability):

```
metric=val,min_sl=val,max_sl=val,doi=val,
ripso_label=val,ripso_error=val,ripso_only,cipso_only,msix_only
```

If *val* contains a space, the space must be protected by double quotes around the value.

The keywords to be used for the emetric are described below:

metric=

Specify an integer from 0 to 15 for the number of hops to the destination. Mandatory.

min_sl and **max_sl**

Specify a sensitivity label in either hexadecimal or string form. Mandatory.

doi= Specify a nonzero integer corresponding to a CIPSO domain of interpretation. If this keyword is specified, do not specify **ripso_label**, **ripso_error**, **ripso_only**, or **msix_only**.

ripso_label=

Specify the classification, followed by a space, followed by a list of protection authority flags (PAF) separated by semicolons (;). The classification and the PAF flags can be specified either in hexadecimal or string form. The supported classifications are TOP SECRET, SECRET, CONFIDENTIAL, and UNCLASSIFIED. The PAF flags (also referred to as the Send PAF) are GENSER, SIOP-ESI, SCI, NSA, DOE. If this keyword is specified, **ripso_error** is required. If this keyword is specified, do not specify **doi**, **cipso_only**, or **msix_only**.

ripso_error=

Specify a list of protection flags separated by semicolons (;) in either hexadecimal or string form. The supported PAF flags (also referred to as the Return PAF) are GENSER, SIOP-ESI, SCI, NSA, and DOE. If this keyword is specified, **ripso_label** is required. If this keyword is specified, do not specify **doi**, **cipso_only**, or **msix_only**.

ripso_only

Specify without a value. If a SUN_RIPSO gateway is involved in a route, use this keyword to indicate that a route can only forward packets having RIPSO labels. If this keyword is specified, **ripso_error** and **ripso_label** are required. If this keyword is specified, do not specify **doi**, **cipso_only** or **msix_only**.

cipso_only

Specify without a value. If a SUN_CIPSO gateway is involved in a route, use this keyword to indicate that a route can only forward packets having CIPSO labels. If this keyword is specified, a **doi** is required. If this keyword is specified, do not specify **ripso_label**, **ripso_error**, **ripso_only** or **msix_only**.

msix_only

Specify without a value. If a SUN_MSIX gateway is involved in a route, use this keyword to indicate that a route can only forward packets having MSIX labels. If this keyword is specified, do not specify doi, ripso_label, ripso_error, ripso_only or cipso_only.

EXAMPLES

The first two lines in the following example show a default and a network entry, each with a simple metric. The third line shows an entry for a network that specifies the gateway name as merlot-118, and the metric as 2, and that assigns an SRI that specifies a label range from UNCLASSIFIED to CONFIDENTIAL, a ripso label of CONFIDENTIAL GENSER, and a ripso error of GENSER. The fourth line is an entry for a host, with an IP address 129.150.101.3. The host entry specifies a gateway called trusted, with a label range of TOP SECRET to TOP SECRET, a cipso doi of 1, and the optional keyword cipso_only. (The long lines are broken because they do not fit on a single line.)

```
default 129.150.117.1 1
net 129.150.113.0 merlot 1
net 129.150.116.0 merlot-118 -m metric=2,min_sl="UNCLASSIFIED",
max_sl="CONFIDENTIAL",ripso_label="CONFIDENTIAL GENSER",
ripso_error="GENSER"
host 129.150.101.3 trusted -m metric=3,min_sl="TOP SECRET",max_sl="TOP SECRET",
doi=1,cipso_only
```

SEE ALSO

in.rdisc(1MTSOL), **in.routed**(1MTSOL), **route**(1MTSOL), *Trusted Solaris Administrator's Procedures*

NAME	tsolinfo – package security-attribute description file												
DESCRIPTION	<p>tsolinfo describes security attributes used as overrides for file attributes of files contained in a package. This ASCII file is created by the developer of a software package and is included in the package. If the file is not included in the package, a set of default file-system security attributes will be used.</p> <p>Each entry in the tsolinfo file describes a single file security attribute for a specific file. The entry consists of several fields of information, each field separated by a space. Lines that begin with “#” are comment lines and are ignored. The fields are described below and must appear in the order shown.</p> <p><i>attribute</i> A character field that defines the attribute type. Valid attribute types are:</p> <table border="0" style="margin-left: 2em;"> <tr> <td>label</td> <td>A CMW label in ASCII</td> </tr> <tr> <td>acl</td> <td>A comma-separated list of acl entries terminated with a comma.</td> </tr> <tr> <td>allowed_privs</td> <td>A list of comma-separated allowed privileges.</td> </tr> <tr> <td>forced_privs</td> <td>A list of comma-separated forced privileges.</td> </tr> <tr> <td>mld</td> <td>Specifies a multi-level directory.</td> </tr> <tr> <td>public</td> <td>Specifies that read operations on this file should not be audited. Do not set an attribute value for this type.</td> </tr> </table> <p><i>pathname</i> A character file that defines the name of the file for which the attribute is being defined.</p> <p><i>attribute-value</i> A character string that defines the value of the attribute. This field is not valid for the "mld" attribute.</p> <p>The tsolinfo file also provides a special set of entries to define a set of default security attributes associated with all of the files within a package. The "default" attribute is used to denote a default attribute entry. The pathname component of the entry is replaced with the name of the attribute for which the default is being set. Package defaults can be set for any of the attributes described above. The package defaults override the filesystem default security attributes.</p> <p>The tsolinfo file should be created at the same time the package prototype file and should be located in the same directory. The tsolinfo file must be included in the package prototype file by using the package prototype include command.</p> <p>When the pkgmk(1) command is used to create a package, the tsolinfo file is relocated to the install/ subdirectory of the newly created package directory.</p>	label	A CMW label in ASCII	acl	A comma-separated list of acl entries terminated with a comma.	allowed_privs	A list of comma-separated allowed privileges.	forced_privs	A list of comma-separated forced privileges.	mld	Specifies a multi-level directory.	public	Specifies that read operations on this file should not be audited. Do not set an attribute value for this type.
label	A CMW label in ASCII												
acl	A comma-separated list of acl entries terminated with a comma.												
allowed_privs	A list of comma-separated allowed privileges.												
forced_privs	A list of comma-separated forced privileges.												
mld	Specifies a multi-level directory.												
public	Specifies that read operations on this file should not be audited. Do not set an attribute value for this type.												
EXAMPLES	<p>The following is an example of a tsolinfo file:</p> <pre> default label ADMIN_LOW[ADMIN_LOW] default allowed_privs all default forced_privs all label usr/sbin/myfile ADMIN_HIGH[ADMIN_HIGH] forced_privs usr/sbin/myfile file_mac_read allowed_privs usr/sbin/myfile file_mac_read,file_mac_write </pre>												

SEE ALSO**pkginfo(4), pkgmap(4), pkgmk(1), prototype(4), setfsattr(1TSOL)****NOTES**

The **tsolinfo** file should only contain entries for pathnames that require special file security attributes, other than the default ones supplied by the UFS filesystem. If the package does not contain any files that require special file security attributes, the **tsolinfo** file should not be created.

If the **tsolinfo** file is not present during package installation, the files contained within a package are assigned default file security attributes provided by the UFS filesystem.

If the **tsolinfo** file contains only the default entries, all of the files within a package are installed with security attributes specified by the **tsolinfo** file entries, along with any non-conflicting default UFS attributes.

NAME	tsolprof – user profiles database
SYNOPSIS	<code>/etc/security/tsol/tsolprof</code>
DESCRIPTION	<p>The tsolprof database specifies security attributes associated with profiles. A profile is a logical grouping of authorizations, commands, and actions, which is interpreted by dtwm(1) and pfsh(1M)TSOL) to form a secure execution environment. Each user or role account is assigned zero or more profiles in the tsoluser(4TSOL) database file.</p> <p>tsolprof can be used with other profile sources, namely the NIS+ table tsolprof. Programs use the getprofent(3TSOL) routines to gain access to this information.</p> <p>Administrators can modify tsolprof and should do so only through the Profile Manager application, which is accessible through solstice(1M). Editing the file directly, such as through a text editor, is strongly discouraged.</p> <p>Each entry in the tsolprof database consists of one line of text, containing at least these five elements: the profile name, a short description of its use, a list of authorizations, a list of permitted actions, and a list of permitted commands. Line continuations and comments are not allowed. The basic format of each entry is:</p> <pre>profile:description:authorizations:actions:commands:links:flags</pre>
profile field	The name of the profile. If this field contains a carat (^), the entry is interpreted as a continuation of another profile entry.
description field	Descriptive text. The field should explain why a user might be assigned the profile.
authorizations field	A comma-separated list of authorization numbers or names; or the key word all or none .
actions field	<p>Zero or more semicolon-separated sets of action information; or the key word <i>none</i>, which indicates that execution of actions is not permitted. A set of action information is specified in the form:</p> <pre>actname;argclass;argtype;argmode;argcount;privs;euid;egid;min;max[;...]</pre> <p>actname is the name of the action as defined by CDE. This field also accepts the asterisk (*), indicating that all actions executed will gain any specified privileges, UID, and GID, and be restricted by the given label range. When the asterisk is used, the next four fields (<i>argclass</i>, <i>argtype</i>, <i>argmode</i>, and <i>argcount</i>) are irrelevant.</p> <p>argclass is the argument class (for example, FILE or SESSION).</p> <p>argtype is the type of argument (for example, POST_SCRIPT or GIF).</p> <p>argmode is the argument mode (for example, f, x, d, r, or &).</p> <p>argcount is the number of arguments expected for that action.</p> <p>privs is a comma-separated list of privilege numbers, which are defined in <code>/usr/include/sys/tsol/priv_names.h</code>. The keyword all selects all privileges.</p>

uid contains a single UID or the keyword **none**. Actions designated with the **uid** set will run with the effective UID of the user indicated. This is similar to setting the **setuid** bit on a file.

egid contains a single GID or the keyword **none**. Actions designated with the **egid** set will run with the effective GID of the user indicated. This is similar to setting the **setgid** bit on a file.

min specifies the minimum label at which the user must operate in order to execute the action.

max specifies the maximum label at which the user must operate in order to execute the action.

To specify another action, place a semicolon after the current **actions** field and repeat the required information for the next action.

commands field

Zero or more semicolon-separated sets of command information; or the keyword *none*, indicating that no commands are allowed. Each set of command information is specified in the form:

```
dir;filename;privs;uid;egid;min;max[;...]
```

dir is an absolute path to the directory containing the subsequent file names; or the at symbol (@), indicating that the entry applies to the previous **dir**; or an asterisk (*), indicating that all commands executed should have the privileges, UID, and GID specified, and should be restricted by the given label range. The first **dir** entry must be an absolute path. When the asterisk is used, the next field (filename) is not interpreted.

filename is the name of the file to which the subsequent attributes apply when the file is executed.

privs is a comma-separated list of privilege numbers that make up the effective privileges when the command is executed. The keyword **all** selects all privileges.

uid and **egid** are similar to the **setuid** and **setgid** bits on a file.

min specifies the minimum label at which the user must operate in order to execute the action.

max specifies the maximum label at which the user must operate in order to execute the action.

links field

This field specifies how many profile entries the profile has (in most cases, 1). This field is used to create a profile with more than 8000 bytes of data.

flags field

Reserved for future use. It contains the keyword **none**.

EXAMPLE

For the sake of clarity on this man page, examples are shown using a continuation character (\). In the database file, however, the backslash is not permitted because each entry is made on a single line.

```
All:Foundation.  A Standard Solaris User.:none:\
*;*;*;*;*; ; ; ; ; :* ; ; ; ; ; :1:none
Custom Admin Role:Modify this profile to customize the admin\
role for your site.:none:TrustedEditor;*;*;*;*;4,5,6,43,44\
; ; ; ; :/usr/dt/bin;trusted_edit;4,5,6,43,44; ; ; ; :1:none
```

The All profile is noted as "Foundation" profile, meaning that it is best used as the first profile assigned to a user. It happens to provide access to all commands and actions, but without any privileges, label restrictions, or the ability to set the UID or GID of the executed command or action.

The Custom Admin Role profile grants a user the **dac_read** (4), **dac_search** (5), **dac_write** (6), **proc_audit_appl** (43), and **proc_audit_tcb** (44) privileges. With this profile a user can edit files. Both the TrustedEditor action and the trusted_edit command are assigned enough privilege to get past any discretionary file access restrictions and to write to the audit trail, which is necessary to record changes made to administrative files.

FILES /etc/security/tsol/tsolprof

CAVEATS The maximum length of a single line in the profile database is TSOLPROF_MAX_NIS_ENT characters as defined in **/etc/default/libtsolddb**. The Profile Manager and the interfaces described in **getprofent**(3TSOL) are designed to transparently prevent an entry from exceeding this limit. If the limit is exceeded and the **tsolprof** database is loaded into NIS+, the **rpc.nisd**(1MTSOL) may terminate and dump core.

Do not use the following symbols within a profile field: colon (:), semicolon (;), comma (,), carat (^), tab (\t), carriage return (\n), pound (#), or backslash (\).

SEE ALSO **solstice**(1M), **getprofent**(3TSOL), **auth_name**(4TSOL), **priv_name**(4TSOL), **tsoluser**(4TSOL)

NAME	tsoluser – Trusted Solaris 2 User Security Attributes Database
SYNOPSIS	<code>/etc/security/tsol/tsoluser</code>
DESCRIPTION	<p>The tsoluser database specifies additional security attributes associated with users and roles.</p> <p>Each entry in the file consists of a single line, with fields separated by a colons (:). Line continuations and comments are not allowed. Each entry has the form (for readability, the line is shown in this man page as two lines):</p> <pre>user:lock:gen:profiles:roles:idletime:idlecmd:labelview:\ labeltrans:labelmin:clearance:usertype:res1:res2:res3</pre> <p>user is the name of the user as specified in the passwd database.</p> <p>lock contains one of the keywords: <i>locked</i>, or <i>open</i>. <i>locked</i> specifies that the user is not allowed to log in to the system. <i>open</i> specifies that the user is allowed to log in. Programs such as login(1) and dtlogin(1X) may choose to change the keyword <i>open</i> to <i>locked</i>, for example, when a user enters an invalid password too frequently.</p> <p>gen contains either of the strings: automatic, or manual. automatic specifies that a user must choose a machine-generated password to change a password. manual specifies that a user may devise a password of his or her choice.</p> <p>profiles contains a ordered, comma-separated list of profile names chosen from tsolprof(4TSOL); or the key word none, indicating that dtwm(1) will not permit the user to use any actions and pfsh(1MTSOL) will not permit the user to execute commands.</p> <p>roles contains a comma-separated list of role names from the set of user accounts in this database whose usertype field indicates the account is a role; or the keyword <i>none</i>, indicating that the user is not permitted to assume any role.</p> <p>idletime contains a number representing the number of minutes a workstation may remain idle before the the window manager attempts the task specified in idlecmd. A zero in this field specifies that the idlecmd command is never executed.</p> <p>idlecmd contains one of two keywords which dtwm(1) interprets when a workstation is idle for too long. The keyword lock specifies that the workstation is to be locked (and thus requires the user to provide a password to resume the session). The keyword <i>logout</i> specifies that session is to be terminated (thus killing the user's processes launched in the current session).</p> <p>labelview contains three comma-separated keywords. The first word can be either internal, specifying that the user may see the ADMIN_LOW and ADMIN_HIGH labels displayed by various commands and applications; or external, specifying that the user may not see the labels; or sysdef, indicating that the label visibility should be determined by the system default, as recorded in the label_encodings file. The second word may be either showil, indicating that information labels are not displayed; or hideil, indicating that the labels are not displayed. The third word may be showsl, indicating that sensitivity labels are displayed; or hidesl, indicating that the labels are not displayed.</p>

utnorm:res1:res2:res3

The first four entries (root, admin, secadmin, and oper) are the default Trusted Solaris roles. The first entry, root, has a handful of profiles which allow it to install software and bootstrap the system. Note that the root entry also has the profile named "All," which gives the role wide birth as far as which commands it can execute while installing software.

Each of the next three has a much more restrictive set of profiles. These profiles are designed to allow the user to perform only the tasks for which that role is responsible.

The fifth entry is for the user named install, who has the "Nothing" profile and thus can execute no commands nor actions. The user can, however, assume the **root** role, which provides sufficient capability to configure the system immediately after it has been installed.

FILES /etc/security/tsol/tsoluser

SEE ALSO dtwm(1), getprofent(3TSOL), getuserent(3TSOL), pfsh(1MTSOL), tsolprof(4TSOL).

NAME	vfstab – table of file system defaults							
DESCRIPTION	<p>The file /etc/vfstab describes defaults for each file system. The information is stored in a table with the following column headings:</p> <table border="0"> <tr> <td><i>device to mount</i></td> <td><i>device to fsck</i></td> <td><i>mount point</i></td> <td><i>FS type</i></td> <td><i>fsck pass</i></td> <td><i>mount at boot</i></td> <td><i>mount options</i></td> </tr> </table> <p>The fields in the table are space-separated and show the resource name (<i>device to mount</i>), the raw device to fsck (<i>device to fsck</i>), the default mount directory (<i>mount point</i>), the name of the file system type (<i>FS type</i>), the number used by fsck to decide whether to check the file system automatically (<i>fsck pass</i>), whether the file system should be mounted automatically by mountall (<i>mount at boot</i>), and the file system mount options (<i>mount options</i>). (See respective mount file system man page below in SEE ALSO for <i>mount options</i>.) A '-' is used to indicate no entry in a field. This may be used when a field does not apply to the resource being mounted.</p> <p>The getvfsent(3C) family of routines is used to read and write to /etc/vfstab.</p> <p>/etc/vfstab may be used to specify swap areas. An entry so specified, (which can be a file or a device), will automatically be added as a swap area by the /sbin/swapadd script when the system boots. To specify a swap area, the <i>device-to-mount</i> field contains the name of the swap file or device, the <i>FS-type</i> is "swap", <i>mount-at-boot</i> is "no" and all other fields have no entry.</p> <p>Mount-time security attributes for a file system specified in the vfstab file can be specified either with the -S option on the mount(1M) command line or in an entry created for the file system in the vfstab_adjunct(4TSOL) file. See the DESCRIPTION sections in the mount and the vfstab_adjunct man pages for more about specifying security attributes. The vfstab file should not be edited directly; instead, it should be edited using the Set Mount Points action, which maintains the proper user, group, sensitivity label, and file permissions for the file and audits all changes. The Set Mount Points action resides in the System_Admin folder available in the Application Manager folder in the Front Panel. By default, the administrator (admin) role has the Set Mount Points action in the File System Management execution profile.</p>	<i>device to mount</i>	<i>device to fsck</i>	<i>mount point</i>	<i>FS type</i>	<i>fsck pass</i>	<i>mount at boot</i>	<i>mount options</i>
<i>device to mount</i>	<i>device to fsck</i>	<i>mount point</i>	<i>FS type</i>	<i>fsck pass</i>	<i>mount at boot</i>	<i>mount options</i>		
SUMMARY OF TRUSTED SOLARIS CHANGES	<p>Two new pairs of security-relevant mount options devices nodevices, and priv nopriv can be specified in the vfstab file for filesystems that support them as filesystem-specific options: mount_hfs(1M), mount_nfs(1M), , and mount_ufs(1M).</p> <p>Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). Trusted Solaris security policy applies when mounting. The vfstab file should be edited by using the Set Mount Points action.</p>							
SEE ALSO	<p>fsck(1M), mount(1M), mount_cachefs(1M), mount_hfs(1M), mount_nfs(1M), mount_tmpfs(1M), mount_ufs(1M), setmnt(1M), swap(1M), getvfsent(3C), and vfstab_adjunct(4TSOL), <i>System Administration Guide</i>, <i>Trusted Solaris Administrator's Procedures</i>, <i>Trusted Solaris Administration Overview</i></p>							

NAME	vfstab_adjunct – attribute data file for mounting a file system
AVAILABILITY	SUNWtsolr
DESCRIPTION	<p>The vfstab_adjunct file can be used to assign any or all of the following mount-time security attributes to the named file system when appropriate: an ACL, a mode, a user ID, a group ID, an information label, a sensitivity label, forced privilege(s), allowed privilege(s), a file attribute flag, a filesystem label range, or an MLD prefix. If the mount(1MTSOL) command is called with the -S option to specify security attributes, the vfstab_adjunct file is not consulted. Specifying mount-time attributes is meaningful only when mounting file systems that do not support the attributes.</p> <p>If the file system already has security attributes, the attributes specified at mount time are ignored and a message is issued. Otherwise, the mount time attributes are used as the file system-wide security attributes. When access control decisions are made, any security attributes on a file or directory always take precedence over security attributes specified either at the filesystem level or mount time.</p> <p>The vfstab_adjunct file should not be edited directly; instead, it should be edited using the Set Mount Attributes action, which maintains the proper user, group, sensitivity label, and file permissions for the file and audits all changes. The Set Mount Attributes action resides in the System_Admin folder available in the Application Manager folder in the Front Panel. By default, the security administrator (secadmin) role has the Set Mount Attributes action in the File System Security execution profile.</p> <p>Mount-time security attributes can be specified for file systems whose objects do not have any attributes (such as user and group IDs) and for file systems that do not have the Trusted Solaris extended security attributes (such as sensitivity labels). When an appropriate attribute is not specified at mount-time for a fixed attribute file system, a default value is applied. The default values are described later in this section.</p> <p>File systems with the tsol_attr flag have a full set of Trusted Solaris extended security attributes already defined. (See the getfsattr(1MTSOL) man page for how to get attributes on mounted file systems.) Because the attributes can be changed on these file systems <i>after</i> they are mounted, they are called <i>variable</i> file systems. For example, the sensitivity label on a file in a mounted variable file system can be changed by an authorized user. Even though they can be changed after mounting, security attributes on variable file systems cannot be overridden at mount-time. The attributes are obtained directly from the file system.</p> <p>File systems that do not support the Trusted Solaris extended security attributes are called <i>fixed</i> because any attributes assigned to them (either at mount time or by default) cannot be changed. For example, the sensitivity label specified for a mounted fixed-attribute file system cannot be changed on any of the objects in that file system. An object that is moved or copied from the fixed file system to a variable file system can be changed after the move.</p>

Each record in the **vfstab_adjunct** file represents a single file system. An entry consists of the file system's full pathname followed by a semicolon, followed by keyword/value assignments in semicolon-separated fields.

The pathname of the file system is the only portion of the entry that is required and therefore has no keyword associated with it. All keyword fields are optional and follow the format:

keyword=value

where *keyword* is one of the following:

- | | |
|-----------------|---|
| acc_acl | Sets the same ACL on all files or directories in the file system. See aclfromtext(3) for the format. |
| mode | Sets a DAC permission mode for each object in the file system. The only supported mode is the absolute mode, which is specified using octal numbers. See the description for the absolute-mode parameter on the chmod(1MTSOL) man page. (Because the mode is an object-level attribute that has precedence over any mount-time attributes, setting a mode is only meaningful in the rare case when the type of file system being mounted does not support permission bits. In such cases, it is recommended that a value be explicitly specified for the mode.) |
| attr_flg | Sets an attribute flag on all files in the file system. The only supported attr_flag value is public , whose effect is that when certain read operations are performed on any object in the file system on which this flag is set, audit records are not generated even when the operations are part of a preselected audit class, with the following exception. If the audit pseudo event for use of privilege (AUE_UPRIV) is included in a preselected audit class and if the operation involves the use of privilege, then an audit record is always generated. With the previous exception, the read operations for which audit records are not generated when the public flag is set are: access(2TSOL) , fgetcmwlabel(2TSOL) , fgetslidname(2TSOL) , fstatvfs(2TSOL) , getcmwfsrange(2TSOL) , getcmwlabel(2TSOL) , getfpriv(2TSOL) , getmldadorn(2TSOL) , getslidname(2TSOL) , lgetcmwlabel(2TSOL) , lstat(2TSOL) , mldlstat(2TSOL) , mldstat(2TSOL) , open(2TSOL) —read only, pathconf(2TSOL) , preadl(2TSOL) , readl(2TSOL) , readlink(2TSOL) , stat(2TSOL) , and statvfs(2TSOL) . See <i>Trusted Solaris Audit Administration Manual</i> and <i>Trusted Solaris Administrator's Procedures</i> for more details. |
| gid | Sets the group ID for all objects in the file system. (Because the GID is an object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have GIDs on its files or directories. In such cases, it is recommended that a value be explicitly specified for the GID.) |
| uid | Sets the user ID for all objects in the file system. (Because the UID is an |

	object-level attribute that has precedence over any mount-time attributes, setting this is only useful in the rare case when the type of file system being mounted does not have UIDs on its files or directories. In such cases, it is recommended that a value be explicitly specified for the UID.)
ilabel	Sets the information label for all objects in the file system. Specify the information label in string (text) or hexadecimal format.
slabel	Sets the sensitivity label for all objects in the file system. Specify the sensitivity label in in string (text) or hexadecimal format.
forced	Specify one or more forced privileges for all executable files in the file system. Specify symbolic privilege name(s) in a comma-separated list (such as: forced=file_audit, file_chown;) or use all to indicate all privileges. Using none or omitting the keyword results in no forced privileges being applied. For example, the assignment of forced=; results in the default of "none" being applied. Any forced privileges must be a subset of the allowed privileges. See priv_desc(4TSOL) for names of privileges.
allowed	Specify one or more allowed privilege(s) for all executable files in the file system. Specify symbolic privilege names in a comma-separated list (such as: allowed=file_audit, file_chown;) or use all to indicate all privileges. Using none or omitting the keyword results in no allowed privileges being applied. See priv_desc(4TSOL) for names of privileges. Any allowed privilege(s) must be a superset of the forced privileges.
low_range	Specify the lower bound of the file system label range as a sensitivity label in string (text) or hexadecimal format.
hi_range	Specify the upper bound of the file system label range as a sensitivity label in string (text) or hexadecimal format.
mld_prefix	Set a prefix to be used in the adorned names of multilevel directories. [See multilevel directories in the DEFINITIONS in Intro(2TSOL) for more about the MLD prefix.] Specify the value in text format (such as: .MLD. or .hidden.). On unlabeled (fixed attribute) file systems, the prefix generally has no useful effect—with the exception that an mld_prefix should be supplied if another file system that has the tsol_attr flag is being mounted on the unlabeled file system and the root of that filesystem is an MLD.

A comment line or entry is terminated by an unescaped newline character. Lines ending with a (\) (backslash) continue the current entry to the next line. Leading and trailing white space characters (blank, tab) surrounding a keyword or an attribute value are ignored. When a keyword value is quoted, spaces can be included within the value. Comments are indicated by a pound sign (#) at the beginning of a line and cause the rest of the line to be ignored.

When a keyword appears without an attribute value or when a keyword is missing, a default value is assigned to that attribute. The default values for fixed attribute file systems are:

acc_acl	None
mode	The mode should always be explicitly set for file systems that do not support access modes, such as MS-DOS (pcfs type) file systems.
attr_flag	None
gid	The GID should always be explicitly set for file systems that do not support group IDs, such as MS-DOS (pcfs type) file systems.
uid	The UID should always be explicitly set for file systems that do not support user IDs, such as MS-DOS (pcfs type) file systems. ADMIN_LOW
slabel	The default sensitivity label of a fixed file system being mounted from a local device (such as a hard disk, floppy, or CD-ROM) is the sensitivity label of the device. For an allocated device, the file system is assigned the sensitivity label at which the device was allocated.
forced	None
allowed	None
low_range	ADMIN_LOW
hi_range	ADMIN_HIGH
mld_prefix	None

EXAMPLES

The following example sets a sensitivity label of PUBLIC on a file system (**/workspaces**) being mounted from an unlabeled host running the Solaris operating environment. For this to work, PUBLIC must be a valid sensitivity label on the local host, the file system must either be automounted or an entry must exist for the file system in the **vfstab(4TSOL)** file. Also, entries for the unlabeled host in the **tnrhdb/tnrhtp(4TSOL)** files must assign a template to the unlabeled host that specifies a matching default sensitivity label of PUBLIC.

```
/workspaces; \  
slabel=PUBLIC;
```

The following example is for a DOS filesystem named **/no_attributes** being mounted from a floppy disk. The file system contains an executable that needs the **file_chown** privilege in order to work. The entry sets a **uid** and **gid** of 0, a mode of 02777, a public attribute flag, a forced privilege of **file_chown** and allowed privileges equal to all. It explicitly sets the **low_range** for the file system to **admin_low** and lowers the **hi_range** from the default of **admin_high** to **admin_low**.

```
/no_attributes; \  
mode=02777; \  
attr_flg=public; \  
gid=0; \  
uid=0; \
```

```
ilabel=admin_low; \  
slabel=admin_low; \  
forced=file_chown; \  
allowed=all; \  
low_range=admin_low; \  
hi_range=admin_low;
```

SEE ALSO

setfacl(1), **getfattrflag(1TSOL)**, **getfsattr(1MTSOL)**, **setfsattr(1MTSOL)**,
getmldadorn(1TSOL), **mount(1MTSOL)**, **mount_cacheufs(1M)**, **mount_hufs(1MTSOL)**,
mount_nfs(1MTSOL), **mount_nfs(1MTSOL)**, **mount_tmpfs(1MTSOL)**,
mount_ufs(1MTSOL) **newsecfs(1MTSOL)** **priv_desc(4TSOL)**, *Trusted Solaris Audit
Administration Manual* and *Trusted Solaris Administrator's Procedures*

Index

A

audit — audit control file, 4TSOL-19, 4TSOL-22
audit trail file
 audit, 4TSOL-10
audit.log, 4TSOL-10
audit_class password file, 4TSOL-17
audit_event password file, 4TSOL-23
audit_user password file, 4TSOL-24

C

config.privs — window privileges that override
system checks, 4TSOL-34

D

device_allocate
 device access control file, 4TSOL-36
device_deallocate
 device access control file, 4TSOL-38
device_maps
 device access control file, 4TSOL-40
devices
 access control file — device_allocate,
 4TSOL-36, 4TSOL-38, 4TSOL-40

F

file formats
 — intro, 4TSOL-7
file system
 defaults — vfstab, 4TSOL-137
 mounted— mtab, 4TSOL-57
files used by programs
 /etc/security/device_maps — device_maps
 file, 4TSOL-40
 /etc/security/device_allocate —
 device_allocate file, 4TSOL-37

I

inetd.conf — Internet server database, 4TSOL-45
inittab — script for init, 4TSOL-48
Internet servers database — servers, 4TSOL-45

L

label_encodings —label encodings file, 4TSOL-51

M

man(1), 4TSOL-7
modified configuration files, 4TSOL-7
mounted file system table — mtab, 4TSOL-57
mtab — mounted file system table, 4TSOL-57

N

name servers
configuration file — `resolv.conf`,
4TSOL-101

name service switch
configuration file — `nsswitch.conf`,
4TSOL-58

NFS
remote mounted file systems — `rmtab`,
4TSOL-102

`nsswitch.conf` — configuration file for the name
service switch, 4TSOL-58

P

package security attribute description file
— `tsolinfo`, 4TSOL-129

`proc` — process file system, 4TSOL-79

`proc(4TSOL)`, 4TSOL-7

process file system — `proc`, 4TSOL-79

R

remote mounted file systems
— `rmtab`, 4TSOL-102

`resolv.conf` — configuration file for name server
routines, 4TSOL-101

S

security policy, 4TSOL-7

shared resources, local
— `sharetab`, 4TSOL-105

`sharetab` — shared file system table, 4TSOL-105

Solstice AdminSuite, 4TSOL-7

T

`tndlog` — log of `tnd` debug information,
4TSOL-106

TSOL man page suffix, 4TSOL-7

`tsolinfo` — listing of software package contents,
4TSOL-129

V

`vfstab` — defaults for each file system, 4TSOL-137

`vfstab_adjunct` — file mountable security infor-
mation, 4TSOL-138