Trusted Solaris Audit Administration

Trusted Solaris 8 4/01

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

Part No: 816–1049–10
November 2001

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.

Sun, Sun Microsystems, the Sun logo, docs.sun.com, AnswerBook, AnswerBook2, Trusted Solaris, Solaris Management Console and Solaris are trademarks, registered trademarks, or service marks 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.

Federal Acquisitions: Commercial Software -- Government Users Subject to Standard License Terms and Conditions

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.

Des parties de ce produit pourront être dérivées du système 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, docs.sun.com, AnswerBook, AnswerBook2, Solaris Management Console et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, 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 REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

010928@2471

Contents

Figures

Preface

Auditing is a security feature required for a C2 rating in TCSEC, and is a functional requirement in the *Common Criteria for Information Technology Security Evaluation*, Version 2.1, August 1999 (CCv21), an ISO standard (IS 15408). C2 discretionary-access control and identification and authentication features are provided by the Solaris™ environment. The Trusted Solaris™ 2.5.1 operating environment earned an ITSEC evaluation in the United Kingdom of assurance level E3 and functionality F-B1.

Who Should Use This Book

Trusted Solaris Audit Administration is intended for the system administrator whose duties include setting up and maintaining audit file systems, and for the security administrator whose duties include determining what will be audited and analyzing the audit trail. The system administrator should be familiar with file system administration, such as NFS-mounting, sharing directories, exporting directories, and creating disk partitions. The security administrator should be familiar with the site security policy, and with the help of the system administrator, be able to create and modify shell scripts.

How This Book Is Organized

Chapter 1 explains the system management and configuration of the auditing subsystem. Topics discussed include managing audit trail storage, determining global and per-user preselection, and setting site-specific configuration options.

Chapter 2 covers setting up and maintaining auditing at your site. The latter part of the chapter contains procedures for setting up and maintaining auditing.

Chapter 3 describes how the audit daemon creates the audit trail, and how to manage audit files and read the contents. The latter part of the chapter contains procedures for merging audit files, selecting records, reading the audit trail, and backing up the trail.

Chapter 4 contains procedures for troubleshooting the auditing subsystem.

Appendix A lists audit events by their default audit class and alphabetically. It also connects them to their system calls and user commands.

Appendix B describes in detail the content of the audit records generated, including a description of every audit token.

Appendix C lists and describes the man pages for the auditing subsystem and the security attributes on the auditing subsystem files.

Related Books

All sites should have the following books or information available when setting up auditing:

From Sun Microsystems

- *Trusted Solaris 8 4/01 Release Notes*

  Describes any late-breaking news about auditing, including known problems.
- *Trusted Solaris Administrator's Procedures*

  Describes administration tasks, such as assuming a role, in detail.

From Elsewhere

- *Your site security policy*

  Describes the security policy and security procedures at your site.

Other books on auditing that may be useful include:

- *A Guide to Understanding Audit in Trusted Systems*

- *Auditing in a UNIX System*
- *DoD Trusted Computer System Evaluation Criteria (the Orange Book)*
- *Compartmented Mode Workstation Evaluation Criteria*
- *Guideline for Trusted Facility Management and Audit*, Virgil D. Gligor, 1985
- *Common Criteria for Information Technology Security Evaluation, Version 2.1*, August 1999. For online information, see `http://csrc.ncsl.nist.gov/cc/ccv20/ccv2list.htm`.

Ordering Sun Documents

Fatbrain.com, the Internet's most comprehensive professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at `http://www1.fatbrain.com/documentation/sun`.

Accessing Sun Documentation Online

The docs.sun.com^{SM} Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is `http://docs.sun.com`.

Typographic Conventions

The following table describes the typographic conventions used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.

Shell Prompts in Command Examples

The following table shows the default system prompt and administrative role prompts for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
root role prompt	`#`
other administrative role prompts	`$`

CHAPTER 1

Auditing Basics

This chapter explains how auditing works on one system and on a network of Trusted Solaris systems.

- "Auditing Overview" on page 15
- "The Audit Mechanism" on page 17

Auditing makes it possible to:

- Monitor security-relevant events that take place on a system
- Record the events in a network-wide audit trail
- Detect misuse or unauthorized activity (by analyzing the audit trail)
- Review patterns of access, and see the access histories of individuals and objects
- Discover attempts to bypass the protection mechanisms
- Discover extended use of privilege that occurs when a user assumes an administrative role
- Supply additional assurance that attempts to bypass protection mechanisms are recorded and discovered

Auditing may serve as a deterrent: if users know that their actions are likely to be audited, they may be less likely to attempt malicious activities.

Auditing Overview

Auditing in the Trusted Solaris environment is enabled by default, configurable by the system and security administrators, and extensible. By default, audit records are stored in *system_name*`:/var/audit/`. Events in the audit classes `login/logout` and `non-attribute` are audited for the root user.

The system administrator can provide dedicated partitions for audit records. The audit analyst can collect all records from all systems in a Trusted Solaris network into one audit trail. The auditing records from a network of systems can be viewed as one large file. Record selection (called *pre-selection*) using a variety of criteria is possible.

After audit data is collected into one audit trail, selection (called *post-selection*) and interpretation tools enable the audit reviewer to examine specific parts of the audit trail. For example, records can be selected for individual users or groups, for a host name, for a certain type of event on a specific day, or for a time of day.

To simplify audit administration, Trusted Solaris auditing provides classes of auditable events. When the security administrator specifies a class of events to be audited, all events in that class are audited. User commands or kernel system calls are auditable events. Classes of events to be audited are specified per system. Specific users or roles (like `root`, for example) can be audited specially.

The security administrator can modify and extend the provided event-class mappings. For some events, event details that are not required by site security policy can be omitted from the audit record. Audit classes can be audited for failure, for success, or for both. Selecting which activities to monitor is called *pre-selection*. When the auditing subsystem encounters error conditions, the security administrator can specify email addresses to be notified.

In the Trusted Solaris auditing subsystem, audit records are protected from snooping by the sensitivity label `admin_high`. Audit configuration files are accessible by the appropriate administrative role only, and sending records to the audit queue requires privilege. A special privilege, `proc_audit_appl`, is provided for ISVs and integrators to add their applications' audit records to the audit queue. Audit event numbers from 32768 to 65535 are available for third-party trusted applications.

Successful auditing depends on two other security features: identification and authentication. At login, after a user supplies a user name and password, a unique audit ID is associated with the user's process. The audit ID is inherited by every process started during the login session. Even when users change identity, for example, by assuming an administrative role, all of their actions are tracked with the same audit ID.

The rest of this chapter describes the auditing subsystem. Chapter 2 describes how to set up and administer auditing. The latter part of the chapter contains setup and maintenance procedures. Chapter 3, describes the audit trail, how to manage its files, and how to read them. The latter part of the chapter contains typical procedures for managing and analyzing the audit trail.

The Audit Mechanism

Auditing is enabled by an audit daemon that uses six configurable audit files: `audit_class`(4), `audit_event`(4), `audit_control`(4), `audit_user`(4), `audit_startup`(1M), and `audit_warn`(1M). These files are in the `/etc/security` directory and determine what to audit, where to put the audit logs, and what to do when there is trouble. By default, events in the `lo` (login/logout) audit class are audited for the root role, the audit records are written to the `/var/audit` directory, and no one receives mail when there is trouble.

You can suspend and re-enable auditing without rebooting the system, and you can dynamically change what is being audited.

Audit Startup

Auditing is enabled when the audit daemon starts, usually when the system is booted (see the `auditd`(1M) man page). When troubleshooting, the daemon can be started manually by executing `/usr/sbin/auditd` in an `admin_high` shell in the `secadmin` role.

The existence of a file with the pathname `/etc/security/audit_startup` causes the audit daemon to be run automatically when the system enters multiuser mode. The file is actually an executable script that is invoked as part of the startup sequence just prior to the execution of the audit daemon (see the `audit_startup`(1M) man page). A default `audit_startup` script that automatically configures the event-to-class mappings and sets the audit policies is created during audit package installation.

The security administrator can edit the `audit_startup` script to alter the default audit policy. See "Setting Audit Policies" on page 33 for more information on audit policy.

Audit Classes and Events

Security-relevant actions may be audited. The system actions that are auditable are defined as *audit events* in the `/etc/security/audit_event` file. Each auditable event is defined in the file by a symbolic name, an event number, a set of preselection classes, and a short description (see the `audit_event`(4) man page).

Most events are attributable to an individual user. However, some events are *nonattributable* because they occur at the kernel-interrupt level or before a user is identified and authenticated. Nonattributable events are auditable as well.

Each audit event is also defined as belonging to an audit class or classes. Administrators name an audit class (called an *audit flag*) when specifying for the audit daemon what is to be audited. When naming a class, one simultaneously addresses all of the events in that class. The mapping of audit events to classes is configurable and the classes themselves are configurable. These configuration changes are made in the `audit_event` file. New classes are added to the `audit_class` file.

Whether an auditable event is recorded in the audit trail depends on whether the administrator preselects an audit class that includes the specific event.

Audit Classes

The file `/etc/security/audit_class` stores class definitions. Site-specific definitions can be added and default definitions can be changed. Each entry in the file has the form:

mask:*name*:*description*

Each class is represented as a bit in the mask, which is an unsigned integer, giving 32 different available classes plus two global classes, `all` and `no`. `all` is a conjunction of all allowed classes. `no` is the invalid class. Events mapped to the `no` class are not audited. Events mapped solely to the `no` class are not audited even if the `all` class is turned on. Below is a sample `audit_class` file:

```
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
0x00000100:nt:network
0x00000200:ip:ipc
0x00000400:na:non-attribute
0x00001000:lo:login or logout
0x00002000:ax:x server
0x00004000:ap:application
0x000f0000:ad:administrative
0x00010000:ss:change system state
0x00020000:as:system-wide administration
0x00040000:aa:audit administration
0x00080000:ao:other administration
```

0x00300000:pc:process
0x00100000:ps:process start/stop
0x00200000:pm:process modify
0x20000000:io:ioctl
0x40000000:fn:fcntl
0x80000000:ot:other
0xffffffff:all:all classes

If the `no` class is actually turned on for auditing, the audit trail fills up with records for the audit event `AUE_NULL`.

Kernel Events

Events generated by the kernel (system calls) have event numbers between 1 and 2047. The event names for kernel events begin with `AUE_`, followed by an uppercase mnemonic for the event. For example, the event number for the `creat()` system call is 4 and the event name is `AUE_CREAT`.

Within kernel events there is one pseudo-event defined, `AUE_UPRIV`, which audits use-of-privilege decisions.

When the AUE_UPRIV pseudo-event is preselected, audit information is collected internally *even if* the underlying kernel event is not selected. For example, if the kernel event AUE_OPEN_R is not selected for auditing but the pseudo-event AUE_UPRIV is enabled, the kernel event AUE_OPEN_R will be written to the audit trail if a use-of-privilege decision was part of the AUE_OPEN_R system call.

User-Level Events

Events generated by trusted application software outside the kernel range from 2048 to 65535. The event names begin with `AUE_`, followed by a lowercase mnemonic for the event. The file `/etc/security/audit_event` lists individual events in numerical order. For a listing of events by class, see Appendix A. The following table shows general categories of user-related events.

TABLE 1–1 Audit Event Categories

Number Range	Type of Event
2048–65535	User-level audit events
2048–32767	Reserved for Solaris and Trusted Solaris user-level programs
32768–65536	Available for third-party applications

Non-Attribute Events

Events that are not attributable to a user, such as AUE_ENTERPROM.

Audit Records

Each audit record describes the occurrence of a single audited event and includes such information as who did the action, which files were affected, what action was attempted, and where and when it occurred.

The type of information saved for each audit event is defined as a set of *audit tokens*. The definition and structure of every audit token are described in detail in "Audit Token Structure" on page 120. Each time an audit record is created for an event, the record contains some or all of the tokens defined for it, depending on the nature of the event and the audit policy. The audit record descriptions in "Audit Records" on page 149 list in order the audit tokens defined for each event.

Audit records are stored in audit files. An audit trail is one or more audit files in a distributed system. The construction of the audit trail is shown in Figure 1–1. The audit trail may be converted to a human readable format by the `praudit`(1M) command. Specific audit records can be selectively chosen using the `auditreduce`(1M) command. See "Audit Files Management" on page 74, for details.

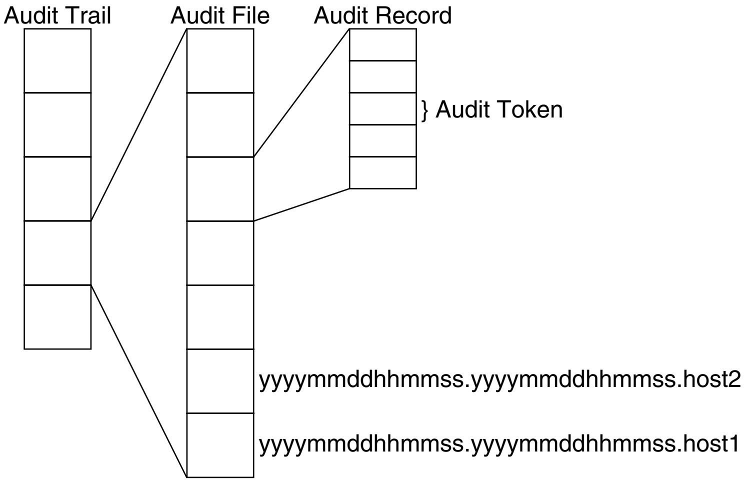

Audit Trail
Audit File
Audit Record
} Audit Token
yyyymmddhhmmss.yyyymmddhhmmss.host2
yyyymmddhhmmss.yyyymmddhhmmss.host1

FIGURE 1–1 From the Audit Token to the Audit Trail

Audit Flags

Audit flags are the short names for the audit classes. Audit flags are used to indicate which classes to audit in the audit_control(4) file, the audit_user(4) file, and as arguments to the auditconfig(1M) command.

The audit_control file is described in "Auditing a System" on page 23. The audit_user file is described in "The audit_user File" on page 26.

Definitions of Audit Flags

Each predefined audit class is listed in Table A–1. The table includes the audit flag (which is the short name that stands for the class), the long name, its audit mask, and a pointer to the list of audit events that by default are in that audit class. The system administrator uses the audit flags in the auditing configuration files to specify which classes of events to audit. Additional classes can be defined and existing classes can be renamed by modifying the audit_class(4) file.

Audit Flag Syntax

Depending on the prefixes, a class of events can be audited whether it succeeds or fails, or only if it succeeds or only if it fails. The format of the audit flag is shown here.

prefixflag

```
-lo        # audit for failure
+lo        # audit for success
lo         # audit for success and failure
```

The audit flag +lo means "all successful attempts to log in and log out". The audit flag –lo means "all failed attempts to log in". (You cannot fail an attempt to logout.). The audit flag lo means "all successful attempts to log in and log out and all failed attempts to log in".

Note – The audit class xs should not be audited for failure. Failures will place a lot of noise in the audit trail. The correct audit flag syntax would be +xs. See the audit_class(4) file for more information on X server audit classes.

For another example, the +all flag refers to all successful attempts of any kind.

Caution – The all flag can generate large amounts of data and fill up audit file systems quickly, so use it only if you have extraordinary reasons to audit everything.

The following table shows prefixes that specify whether the audit class is audited for success or failure or both.

TABLE 1–2 Prefixes Used in Audit Flags

Prefix	Definition
none	Audit for both success and failure
+	Audit for success only
–	Audit for failure only

Prefixes to Modify Previously Set Audit Flags

Use the modification prefixes in any of three ways: in the flags line in the `audit_control`(4) file to modify already-specified flags, as flags in the user's entry in the `audit_user`(4) file, or as arguments to the `auditconfig`(1M) command.

The prefixes in Table 1–3 along with audit flags, turn on or turn off previously specified audit classes. These prefixes turn on or off previously specified flags only.

TABLE 1–3 Prefixes Used to Modify Already-Specified Audit Flags

Prefix	Definition
^–	Turn off for failed attempts
^+	Turn off for successful attempts
^	Turn off for both failed and successful attempts

The ^– prefix is used in the flags line in the following example from an `audit_control` file.

```
flags:lo,ad,–all,^–fc
```

Audit Storage

On every system, the `/etc/security/audit` directory contains subdirectories with all the audit log files. The `/etc/security` directory contains files related to audit configuration. Because the `/etc/security` directory contains the per-system audit_data file, which is used by the audit daemon at boot time, the /etc/security directory must be part of the root file system.

The audit postselection tools look in directories under /etc/security/audit by default. For this reason, the pathname of the mount point for the first audit file system on an audit server is in the form: /etc/security/audit/*server-name* (where *server-name* is the name of the audit server). If more than one audit partition is on an audit server, the name of the second mount point is: /etc/security/audit/*server-name*.1, the third is /etc/security/audit/*server-name*.2, and so forth.

For example, the names of the audit file systems available on the audit file server audubon are /etc/security/audit/audubon and /etc/security/audit/audubon.1.

Each audit file system has a subdirectory named files. This files subdirectory is where the audit files are located and where the auditreduce commands looks for them. For example, the audit file system on audit server audubon has a files subdirectory whose full pathname is: /etc/security/audit/audubon/files.

The local audit_control file on each system directs the audit daemon to put the audit files in the files subdirectory. For example, the dir: line for the audit_control file on a system mounting the audit file system from eagle is:

```
dir: /etc/security/audit/eagle/files
```

The extra level of hierarchy prevents a system's local root file system from filling with audit files when (for whatever reason) the /etc/security/audit/*server-name*[.*suffix*] directory is not available on the audit server. Because the files subdirectory is present on the audit server and the clients use the same naming convention for their local audit log files, /etc/security/audit/*client-name*, audit files cannot be created unintentionally in the local mount-point directory if the mount fails.

Permissions on Audit Directories

In a Trusted Solaris environment, audit directories, such as the /etc/security/audit/*system_name* directory and the files directory directly beneath it, should be protected at the label admin_high. Permissions should be 750.

Auditing a System

Auditing is set per system by the security administrator in the audit_control(4) file. This file on each system is read by the audit daemon. The audit_control file is located in the /etc/security directory.

A system–specific audit_control file is maintained on each system because the dir: lines, and perhaps the minfree: line are specific to the system. In a distributed system, the other lines should be identical.

You specify four kinds of information in four kinds of lines in the audit_control file:

- The *audit flags* line (flags:) contains the audit flags that define what classes of events are audited for all users on the system. The audit flags specified here are referred to as the *system-wide audit flags* or the *system-wide audit preselection mask*. Audit flags are separated by commas, with no spaces.
- The *nonattributable flags* line (naflags:) contains the audit flags that define what classes of events are audited when an action cannot be attributed to a specific user. The flags are separated by commas, with no spaces.
- The *audit threshold* line (minfree:) defines the minimum free-space level for all audit file systems. See "Storing Audit Data" on page 29.

  The minfree percentage must be greater than or equal to 0. The default is 20 percent.
- The *directory definition* lines (dir:) define which audit file systems and directories the system will use to store its audit trail files.

  There may be one or more directory definition lines. The order of the dir: lines is significant, because the auditd command opens audit files in the directories in the order specified (see the audit(1M) man page). The first audit directory specified is the primary audit directory for the system, the second is the secondary audit directory where the audit daemon puts audit trail files when the first one fills, and so forth.

The security administrator modifies the default audit_control file during the configuration process on each system.

After the audit_control file is configured, the system administrator on a distributed system distributes it to the other hosts. After any change in the file, the administrator runs audit -s on every system on the network to instruct the audit daemon to reread its audit_control file.

Note – The audit -s command does not change the preselection mask for existing processes (see "Process Preselection Mask" on page 27). Use auditconfig, setaudit (see the getauid(2) man page), or auditon(2) for existing processes.

Dynamic controls refer to controls put in place by the administrator while processes are running. These persist only while the affected processes (and any of their children) exist, but will not continue in effect at the next login. Dynamic controls apply to one system at a time, since the audit command only applies locally.

Sample audit_control File

Following is a sample audit_control file for the system willet. willet uses two audit file systems on the audit server egret, and a third audit file system mounted from the audit administration server audubon, which is used to store audit records only when the audit file system on egret fills up or is unavailable. The minfree value of 20 percent specifies that the warning script (see the audit_warn(1M) man page) is run when the file systems are 80 percent filled and the audit data for the current system will be stored in the next available audit directory, if any. The flags specify that all logins and administrative operations are to be audited (whether or not they succeed), and that failures of all types except failures to create a file system object are to be audited.

```
flags:lo,ad,-all,^-fc
naflags:lo,nt
minfree:20
dir:/etc/security/audit/egret/files
dir:/etc/security/audit/egret.1/files
#
# Audit filesystem used when egret fills up
#
dir:/etc/security/audit/audubon
```

Note – Successful events and failed events are treated separately, so a process can (for example) generate more audit records when an error occurs than when the event is successful.

Each process has two sets of one-bit flags for audit classes. One set controls whether the process is audited when an event in the class is requested successfully. The other set controls auditing when an event is requested but fails (for any reason). It is common for processes to be more heavily audited for failures than for successes, since this can be used to detect attempts at browsing and other types of attempts at violating system security.

Auditing User Exceptions

The security administrator sets up auditing for the default configuration. You may want all users and administrators to be audited according to the system-wide audit flags you specified in the audit_control file. To fine-tune auditing for individual users, you add user entries to the audit_user file. You may also choose to add audit flags to users' entries at the time you add new users, and you should probably set up auditing for the new user just after you unlock the account and configure the security attributes for that user.

Note – Alterations to a static auditing database (`audit_control`, `audit_startup`, or `audit_warn`) on one system should be copied to all hosts on the network. See "To Distribute Audit Configuration Files" on page 57.

The `audit_user` database is distributed to all hosts by the User Accounts tool in the Solaris Management Console.

In addition to supplying the per-user audit control information in the static databases, you can dynamically adjust the state of auditing while a user's processes are active on a single system.

The `audit_user` File

If it is desirable to audit some users differently from others, the administrator can edit the `audit_user` file to add audit flags for individual users. If specified, these flags are combined with the system-wide flags specified in the audit control file to determine which classes of events to audit for that user. The flags the administrator adds to the user's entry in the `audit_user` file modify the defaults from the `audit_control` file in two ways: by specifying a set of event classes that are never to be audited for this user or by specifying a set of event classes that are always to be audited.

So, what is audited for an individual user is the combination of the system audit flags and the user's always and never audit flags, shown in "Process Preselection Mask" on page 27.

In the `audit_user` file entry for each user, there are three fields. The first field is the *username*, the second field is the *always-audit* field, the third is the *never-audit field*.

The two auditing fields are processed in sequence, so auditing is enabled by the first field and turned off by the second.

Note – Avoid the placing the `all` flag in the *never-audit* field. This causes all auditing to be turned off for that user, overriding the flags set in the *always-audit* field.

Using the *never-audit* flags for a user is not the same as removing classes from the *always-audit* set. For example, suppose (as shown in the examples below), you have a user `katya` for whom you want to audit everything except successful reads of file system objects. (This is a good way to audit almost everything for a user while generating only about three-quarters of the audit data that would be produced if all data reads were also audited.) You also want to apply the system defaults to `katya`. Here are two possible `audit_user` entries.

The correct entry

katya:all,^+fr:

The incorrect entry:

katya:all:+fr

The first example says, “always audit everything except successful file-reads.” The second example says “always audit everything, but never audit successful file-reads.” The second example is incorrect because it overrides the system default. The first example achieves the desired effect: any earlier default applies, as well as what is specified in the `audit_user` entry.

Process Audit Characteristics

The following audit characteristics are set at initial login:

- Process preselection mask
- Audit ID (AUID)
- Audit Session ID
- Terminal ID (port ID, system ID)

Process Preselection Mask

When a user logs in, `login` combines the system-wide audit flags from the `audit_control` file with the user-specific audit flags (if any) from the `audit_user` file, to establish the *process preselection mask* for the user’s processes. The process preselection mask specifies whether events in each audit event class are to generate audit records.

The algorithm for obtaining the process preselection mask is as follows: the audit flags from the `flags:` line in the `audit_control` file are added to the flags from the *always-audit* field in the user’s entry in the `audit_user` file. The flags from the *never-audit* field from the user’s entry in the `audit_user` file are then subtracted from the total.

user’s process preselection mask = (*flags: line + always audit flags*)
- *never audit flags*

Audit ID

A process also acquires its audit ID when the user logs in, and this audit ID is inherited by all child processes started by the user's initial process. The audit ID helps enforce accountability. Even after a user assumes a role, the audit ID remains the same. The audit ID that is saved in each audit record enables the administrator to always trace actions back to the original user that logged in.

Audit Session ID

The audit session ID is assigned at login and inherited by all descendant processes.

Terminal ID

The terminal ID consists of the host name and the Internet address, followed by a unique number that identifies the physical device on which the user logged in. Most of the time the login will be through the console and the number that corresponds to the console device will be 0.

The audit_data File

When auditd starts on each system, it creates the file /etc/security/audit_data. The format of the file consists of a single entry with the two fields separated by a colon (see the audit_data(4) man page). The first field is the audit daemon's process ID, and the second field is the pathname of the audit file to which the audit daemon is currently writing audit records. Here is an example:

```
# cat /etc/security/audit_data
116:/etc/security/audit/egret.1/files/19910320100002.not_terminated.tern
```

In the Trusted Solaris environment, the audit_date file is protected at the label admin_high.

The Audit Daemon's Role

The following list summarizes what the audit daemon, auditd(1M), does.

- auditd opens and closes audit log files in the directories specified in the audit_control file in the order in which they are specified.
- auditd reads audit data from the kernel and writes it to an audit file.

- auditd executes the audit_warn script when the audit directories fill past limits specified in the audit_control file. The script, by default, sends warnings to the audit_warn alias and to the console. Your site should customize audit_warn to suit your needs. The audit_warn script is described in "The audit_warn Script" on page 30.
- With the system default configuration, when all audit directories are full, processes that generate audit records are suspended and auditd writes a message to the console and to the audit_warn alias. (The auditing policy can be reconfigured with the auditconfig command.) At this point only the system administrator could log in to write audit files to tape, delete audit files from the system, or do other cleanup.

When the audit daemon starts as the system is brought up to multiuser mode, or when the audit daemon is instructed by the audit -s command to reread the file after the file has been edited, auditd determines the amount of free space necessary and reads the list of directories from the audit_control file and uses those as possible locations for creating audit files.

The audit daemon maintains a pointer into this list of directories, starting with the first. Every time the audit daemon needs to create an audit file, it puts the file into the first available directory in the list, starting at the audit daemon's current pointer.

Storing Audit Data

A directory is *suitable* for storing audit records if it is accessible to the audit daemon, which means that it must be mounted, that the network connection (if remote) permits successful access, and that the permissions on the directory permit access. Also in order for a directory to be suitable for audit files, it must have sufficient free space remaining. You can edit the minfree: line in the audit_control file to change the default of 20 percent. To give an example of how the minfree percentage is applied, if the default minimum free space of 20 percent is accepted, an email notice is sent to the audit_warn alias whenever a file system becomes more than 80 percent full.

When no directories on the list have enough free space left, the daemon starts over from the beginning of the list and picks the first accessible directory that has any space available until the hard limit is reached. In the default configuration, if no directories are suitable, the daemon stops processing audit records, and they accumulate within the kernel until all processes generating audit records are suspended.

Keeping Audit Files Manageable

To keep audit files at a manageable size, a cron job can be set up that periodically switches audit files (see the cron(1M) man page). Intervals might range from once per hour to twice per day, depending on the amount of audit data being collected. The data can then be filtered to remove unnecessary information and then compressed.

The audit_warn Script

Whenever the audit daemon encounters an unusual condition while writing audit records, it invokes the /etc/security/audit_warn script. See the audit_warn(1M) man page. This script can be customized by your site to warn of conditions that might require manual intervention or to handle them automatically. For all error conditions audit_warn writes a message to the console and sends a message to the audit_warn alias. This alias should be set up by the administrator after enabling auditing.

When the following conditions are detected by the audit daemon, it invokes audit_warn.

- An audit directory has become more full than the minfree value permits. (The minfree or soft limit is a percentage of the space available on an audit file system.)

  The audit_warn script is invoked with the string soft and the name of the directory whose space available has gone below the minimum. The audit daemon switches automatically to the next suitable directory, and writes the audit files there until this new directory reaches its minfree limit. The audit daemon then goes to each of the remaining directories in the order listed in audit_control, and writes audit records until each is at its minfree limit.

- All the audit directories are more full than the minfree threshold.

  The audit_warn script is invoked with the string allsoft as an argument. A message is written to the console and mail is sent to the audit_warn alias.

  When all audit directories listed in audit_control are at their minfree limits, the audit daemon switches back to the first one, and writes audit records until the directory completely fills.

- An audit directory has become completely full with no space remaining.

  The audit_warn script is invoked with the string hard and the name of the directory as arguments. A message is written to the console and mail is sent to the audit_warn alias.

  The audit daemon switches automatically to the next suitable directory with any space available, if any. The audit daemon goes to each of the remaining directories in the order listed in audit_control, and writes audit records until each is full.

- All the audit directories are completely full. The audit_warn script is invoked with the string allhard as an argument.

In the default configuration, a message is written to the console and mail is sent to the audit_warn alias. The processes generating audit records are suspended. The audit daemon goes into a loop waiting for space to become available and resumes processing audit records when that happens. While audit records are not being processed, no auditable activities take place—every process that attempts to generate an audit record is suspended.

- An internal error occurs: another audit daemon process is already running (string ebusy), a temporary file cannot be used (string tmpfile), the auditsvc(2) system call fails (string auditsvc), or a signal was received during auditing shutdown (string postsigterm).

  Mail is sent to the audit_warn alias.
- A problem is discovered with the audit_control file's contents. By default, mail is sent to the audit_warn alias and a message is sent to the console.

Controlling Audit Costs

Because auditing consumes system resources, you must control the degree of detail that is recorded. When you decide what to audit, consider the following three costs of auditing:

- Costs in increased processing time
- Costs of analysis of audit data
- Costs of storage of audit data

The cost in increased processing time is the least significant of the three costs of auditing. The first reason is that auditing generally does not occur during computational-intensive tasks—image processing, complex calculations, and so forth. The other reason that processing cost is usually insignificant is that single-user systems have plenty of extra CPU cycles.

The cost of analysis is roughly proportional to the amount of audit data collected. The cost of analysis includes the time it takes to merge and review audit records, and the time it takes to archive them and keep them in a safe place.

The fewer records you generate the less time it takes to analyze them, so upcoming sections describe how you can reduce the amount of data collected, while still providing enough coverage to achieve your site's security goals.

Storage cost is the most significant cost of auditing. The amount of audit data depends on the following:

- Number of users
- Number of systems

- Amount of use
- Degree of security required

Because the factors vary from one situation to the next, no formula can determine in advance the amount of disk space to set aside for audit data storage.

Full auditing (with the `all` flag) can fill up a disk quickly. Even a simple task like compiling a program of modest size (for example, 5 files, 5000 lines total) in less than a minute could generate thousands of audit records, occupying many megabytes of disk space. Therefore, it is very important to use the preselection features to reduce the volume of records generated. For example, not auditing the `fr` class can reduce the audit volume by more than two-thirds. Efficient audit file management is also important after the audit records are created to reduce the amount of storage required.

Auditing Efficiently

What to audit, when to audit it, and where to store the files are factors to consider when enforcing your site's security goals while auditing more efficiently. For example, you might try:

- Random auditing of only a certain percentage of users at any one time.
- Real-time monitoring of the audit data for unusual behaviors. (You set up procedures to monitor the audit trail as it is generated for certain activities and to trigger higher levels of auditing of particular users or systems when suspicious events occur.) See "To Read a Current Audit File" on page 76 for an example.
- Setting the public object flag on publicly accessible files or directories. This reduces the potential size of the audit trail while not compromising security, because the viewing of publicly accessible files and directories is not generally interesting for audit purposes. Files so marked do not generate audit records for the following audit events, even if the classes for those events are turned on for auditing:

  AUE_ACCESS, AUE_STAT, AUE_LSTAT, AUE_READLINK, AUE_STATFS, AUE_FSTATFS, AUE_PATHCONF, AUE_OPEN_R, AUE_FGETCMWLABEL, AUE_GETCMWFSRANGE, AUE_GETCMWLABEL, AUE_GETFILEPRIV, AUE_LGETCMWLABEL, AUE_GETMLDADORN, AUE_GETSLDNAME, AUE_OSTAT, AUE_FUSERS, AUE_STATVFS, AUE_XSTAT, and AUE_LXSTAT. The list may not be exhaustive.

  See "To Set Public Object Bit" on page 62 for the procedure.
- Reducing the disk-storage requirements for audit files by combining, reducing, and compressing them (see "To Combine Selected Audit Files " on page 79), and developing procedures for storing them offline.

Setting Audit Policies

The `auditconfig` command provides a command line interface to get and set audit configuration information and audit policy. It can be used in the `audit_startup`(1M) script to set audit policies when the audit daemon is started. See the `auditconfig`(1M) man page and "Dynamic Auditing (Tasks)" on page 62, for examples of the use of the `auditconfig` command.

You can use `auditconfig` with the `-setpolicy` option to change the default Trusted Solaris audit policies. Setting audit policies means to add optional audit tokens to the audit record. The `auditconfig` command with the `-lspolicy` argument shows the audit policies that are optional. See "To Determine Current Audit Policy" on page 63 for the audit policies and their short descriptions. The following gives longer descriptions of the less easily understood policy flags.

Caution – To run auditing in an evaluated configuration, you cannot have the `cnt` policy or the `passwd` policy turned on. They *must* be turned off.

`ahlt` Halt the computer if an asynchronous audit event occurs which can not be delivered to the audit queue. The default is not to halt the system.

`cnt` Do not suspend auditable actions when the queue is full. Count how many audit records are dropped. The default is suspend.

Note – To return to the default, remove the `cnt` policy. See "To Set Audit Policy Temporarily" on page 64 for examples of replacing, adding, and removing audit policies.

`path` Add secondary `path` tokens to audit record. These secondary paths are typically the pathnames of dynamically linked shared libraries or command interpreters for shell scripts. By default they are not included.

`seq` Include a sequence number in every audit record. The default is to not include. (The sequence number could be used to analyze a crash dump to find out whether any audit records are lost.)

CHAPTER 2

Auditing Setup

The focus of this chapter is on setting up auditing for a network of Trusted Solaris systems. It also describes how to set up auditing for a non-networked Trusted Solaris system.

Planning Auditing at Your Site

When the system administrator and security administrator configure the first system for the Trusted Solaris operating environment, auditing is enabled and a limited number of audit records are collected to a default audit location, *system_name*:`/var/audit`. The security administrator needs to plan what to audit and whether to customize site-specific event-to-class mappings. The system administrator plans disk space (local and remote) for the audit files, an audit administration server, and the order of installation.

Planning auditing for a non-networked system is a bit simpler. For a single system, customizing event-to-class mappings may not be worth the time. Your most important task is to ensure that auditing does not slow down your work. Planning the size and locations of auditing partitions can prevent work slowdown, and a regular maintenance schedule can automatically back up and free up the audit partition for more audit records.

Planning What to Audit

Trusted Solaris auditing collects user actions and non-attributable (in the class na, non-attribute) events into audit classes. It is these audit classes, each of which holds a number of events, that are audited for success, for failure, or for both.

Before configuring auditing, understand the audit flags and the types of events they flag. Develop a philosophy of auditing for your organization that is based on the amount of security your site requires and the types of users you administer.

Unless the process audit preselection mask is modified dynamically, the audit characteristics in place when a user logs in are inherited by all processes during the login session, and, unless the databases are modified, the process preselection mask applies in all subsequent login sessions.

See "Audit Events Listed by Audit Class" on page 91 for a list of provided audit classes. Each audit class is listed in its own table, where each audit event's corresponding system call or user command points to its audit record format.

The security administrator plans what to audit based on the site security policy. You can configure a system-wide setup and user exceptions/additions.

1. **Decide if non-attributable events should be audited.**

   The audit flag na represents the non-attributable class of events. For example, accessing the PROM, booting, and remote mounting are non-attributable events. See "Events in Audit Class na " on page 106 for a list of the events in the default non-attribute class.

   When you audit a class, you audit all events in that class. If you want to customize the non-attributable class, see "Planning a Site-Specific Event-to-Class Mapping" on page 38.

   To audit non-attributable events, you will enter the na flag on the naflags: line of the audit_control file.

2. **Decide whether to audit them for success, for failure, or for both.**

   To audit non-attributable events for success, the naflags: line of the audit_control file would look like:

   naflags:+na

   To audit non-attributable events for failure:

   naflags:–na

   To audit non-attributable events for both:

   naflags:na

3. **Decide if *all* events will be audited.**

Note – The class `all` includes all auditable events in the Trusted Solaris software environment. While unusual circumstances may dictate use of this class, typically you would avoid auditing all events.

---

4. **If you are not going to audit all events, repeat step 1 and step 2 for the other audit classes as you did for the class `na`.**

   You enter your auditing decisions in the `audit_control` file when establishing auditing on the first system. The `na` flag goes on the `naflags:` line. All other class flags go on the the `flags:` line of the `audit_control` file.

5. **Determine if there are particular users or roles that should be audited slightly differently than the system-wide setup.**

   You will enter user exceptions to the system setup in the `audit_user` file. In the Trusted Solaris 8 4/01 release, the security administrator does not edit the `audit_user` file directly. The Audit tab on a user's account in the Solaris Management Console (SMC) handles each user's audit flags as part of the account. The SMC distributes the user information using the site's name service.

6. **Be consistent.**

   All hosts in a Trusted Solaris network should have identical `naflags:` entries in their `audit_control` files.

   All hosts in a Trusted Solaris network should have identical `flags:` entries in their `audit_control` files.

   All hosts in a Trusted Solaris network should have identical `audit_user` files. The Solaris Management Console will distribute user audit information using the site's name service.

Considerations When Planning What to Audit

What is audited at your site is based on your site policy and the costs of auditing (time, efficiency, disk space), as discussed in "Controlling Audit Costs " on page 31. The following are factors to consider when using auditing as it is implemented in the Trusted Solaris environment.

- Every audit record stands alone, so records can quickly fill up disk space.

  Therefore, you might want to start with a small amount of auditing and see how the audit partitions fill. You can then make more educated estimates of disk requirements and an audit archiving schedule. You can refine audit classes as you get an estimate of the size of the audit trail.

- The number of events in an audit class does not necessarily correlate to how many records are generated.

For example, the `file read` class contains about the same number of events as the `login or logout` class. Enabling the `file read` class for success is likely to generate many more records than enabling the `login or logout` class for success.

- Auditing for failure locates abnormal events; auditing for success monitors system use.

  If site policy requires monitoring of system use, you will want to set aside more space for the audit trail than if you are auditing for abnormal events.

- Auditing for failure may generate many fewer records than auditing for success.

  For example, auditing for failure of `file read` events in a Trusted Solaris system of sophisticated users can generate many fewer records than turning on the `file read` class for success.

- Configuring the audit classes differently, or setting up new audit classes for audit events can more efficiently satisfy your site requirements. By excluding audit events that site policy does not require to be audited, the audit trail is smaller.

  For example, you may want to create a class `de` for devices. When configuring devices, audit the class for success to generate a record of what devices have been set up and tested. When all devices have been configured, you may want to audit the class for failure.

- Configuring some classes to be audited intermittently may satisfy your site requirements.

  For example, you may want to audit the audit class you created, `de`, intermittently. A cron job, or the command `auditconfig`(1M), enable you to turn auditing on and off for particular classes and set other audit flags dynamically.

Planning a Site-Specific Event-to-Class Mapping

Optional: Skip this section if you are using the default event-to-class mappings provided in the Trusted Solaris environment. Do not skip this section if you have decided to rearrange what events are assigned to what classes, or to create new classes or new events.

Trusted Solaris software handles up to 32 audit classes, including the class `all`. Your site may add classes until the total number is 32.

The security administrator plans site-specific mappings. To plan site-specific mappings:

1. **Decide what classes are needed.**

2. **Decide what events belong in what classes.**

   a. **Decide what events should be copied to another class or classes.**

      An audit event can belong to more than one class. For example, the audit event `AUE_RENAME` belongs to the classes `file create` and `file delete` in the default event-to-class mapping.

   b. **Decide what events should be moved to another class or classes.**

   c. **Decide what events should be added to a class or classes.**

3. **For each class, decide whether to audit it for success, for failure, or for both.**

   When new software programs include audit events not provided by Trusted Solaris software, add the events to existing classes or create a new classes for the new events.

Considerations When Changing Event-to-Class Mappings

The following are factors to consider when changing the contents of default audit classes and creating new ones in the Trusted Solaris environment.

- This document, *Trusted Solaris Audit Administration*, reports the default auditing configuration.

  Document your site's modifications to the auditing defaults, and make the document available to the administrators handling audit administration.
- If you are networked, you must change the auditing configuration files on all the systems when you change the files on one system.

  A network of Trusted Solaris systems should behave like one system. When auditing is enabled, it should be enabled on every host, and every host should be audited for the same classes, with the same defaults, the same user exceptions, and the same event-to-class mappings as every other Trusted Solaris host in the network.

Planning Space on a Non-Networked Systems

Storing audit records on a non-networked system involves setting up at least two local partitions dedicated to audit records, one primary and one backup, and planning a maintenance schedule.

On a non-networked system, plan the size of a disk partition to hold audit records. For efficiency, it is best to place the audit records on a separate disk. For safety, you may want to create two audit partitions on that disk, one as the primary storage area and the other as a backup when the first partition gets full. Set filesystem security attributes to set on the audit directory to prevent snooping on the audit trail.

1. **Estimate the volume of auditing between audit record backups.**

   Balance your security needs against the availability of disk space for audit trail storage.

   A rule of thumb is to assign 200 MB of space per system. However, the disk space requirements for the system are based on how much auditing you perform and may be far greater than this figure.

   "Controlling Audit Costs " on page 31 and "Auditing Efficiently" on page 32 provide guidance on how to reduce storage requirements.

2. **Decide at what point the audit file system sends a warning that it is filling up.**

   You will specify what is called the *minfree limit* for audit partitions in the `audit_control` file. This is the percentage of disk space remaining when the audit administrator is sent an email message (by the `audit_warn` alias) that the disk is getting full. The default is to send the warning when there is 20% disk space remaining. This percentage is tunable.

Planning Space on a Network of Hosts

Storing audit records for a network of hosts involves setting up a local (backup) partition dedicated to audit records, plus a network of audit servers with partitions for remote (primary) audit storage, and plus an audit administration server from which the entire *audit trail* can be monitored. The audit trail is every audit file (audit files hold audit records generated on a system) created by every system on the network.

A networked system should include audit servers to store audit files for users' systems, an audit administration server for central audit analysis and backup, and a local audit partition on every host. You may want to set filesystem security attributes on the directories and mount points to prevent snooping on the audit trail. Create a worksheet to record your auditing plan, or use another mechanism that helps you track the auditing network that you set up.

1. **Determine how much auditing your site needs to do.**

   Balance your site's security needs against the availability of disk space for audit trail storage.

   A rule of thumb is to assign 200 MB of space for each host that will be on the distributed system, but remember that the disk space requirements at your site is based on how much auditing you perform and may be far greater than this figure per host. If you are able to dedicate a local and a remote disk for auditing, one way to set up audit partitions is to divide each disk into two partitions.

   "Controlling Audit Costs " on page 31 and "Auditing Efficiently" on page 32 provide guidance on how to reduce storage requirements while still maintaining site security.

2. **Decide at what point each audit file system for the system sends a warning that it is filling up.**

   You will specify what is called the *minfree limit* for audit partitions in the `audit_control` file. This is the percentage of disk space remaining when the audit administrator is sent an email message (by the `audit_warn` alias) that the disk is getting full. The default is to send the warning when there is 20% disk space remaining. This percentage is tunable.

3. **Determine which hosts will be audit servers.**

   The install team will install these systems before installing the audit client systems.

4. **Plan a local audit partition for each system.**

   The local partition provides a backup in cases where the audit server's partitions are full or when the network is unreachable.

5. **Determine which clients will use which audit file systems on which audit server.**

   Lay out the auditing network. The following figure shows an audit server, `egret`, with file systems `/etc/security/audit/egret`*[.n]*`/files` available to store remote hosts' audit records.

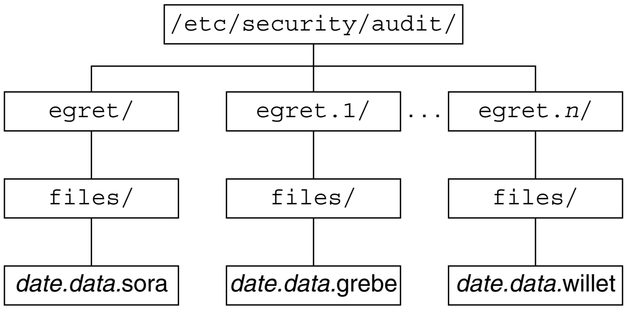


FIGURE 2–1 Audit Server `egret`'s Audit File Systems

6. **Follow the naming conventions for audit file systems.**

   As illustrated in the figure, the convention for naming the audit file systems on a system is:

```
/etc/security/audit/system_name/files
/etc/security/audit/system_name.1/files
/etc/security/audit/system_name.2/files
/etc/security/audit/system_name.3/files ...
```

   For an explanation of the naming scheme, see "Audit Storage" on page 22.

Planning the Rollout

Rolling out the auditing plan to the systems is a job coordinated by the system administrator, who sets up the disks and the network of audit storage, and the security administrator, who decides what is to be audited and enters the information in the audit configuration files. Together, you want to set up an audited network of systems where:

- From one host, the audit analyst is able to read every audit file on every host in the network, and the system operator is able to back up every audit file on every host on the network.

  How: Create an administration server, and mount all audit directories on the server.

- The audit trail is not available for snooping.

  How: Protect audit directories with appropriate discretionary access controls and mandatory access controls. You may want to audit directory access.

- Each host in a Trusted Solaris distributed system is writing records to the audit trail from the first time it is in multiuser mode, and thereafter.

  How: Create audit servers before you create user systems. On all systems, create a dedicated audit partition during installation.

- Every system is audited identically.

  How: Create a central location for all audit configuration files that are not controlled by the Solaris Management Console: `audit_event`, `audit_class`, `audit_control`, `audit_startup`, and `audit_warn`. The examples use the directory `/export/home/tmp` on the NIS+ master. Copy these files to a tape or diskette that is copied to every system.

- When an end user's system is configured, it is able to immediately send its audit records to an audit server.

  How: Create the audit servers and configure them for receiving audit records before the end user systems are set up. Create a procedure to copy the system-wide audit configuration files to each host and to modify the `audit_control` file for the audit storage locations for that host.

- End user's systems are not slowed down by writing audit records.

  How: Regular archiving of the audit trail frees up audit server disk space. Placing the local audit storage on a separate or little-used disk will enable the end user to work quickly when audit records are stored locally.

Rolling Out Auditing at Your Site

To roll out auditing, the system administrator sets up the audit administration server, the audit file servers, the local audit partitions, and what usernames are warned of audit trouble. The security administrator edits the `audit_control`(4) file on the NIS+ root master, and edits other audit configuration files before copying them to a central directory for distribution by tape or floppy. The audit configuration files are copied from the tape to each system as it is configured by the install team. The security administrator edits the `dir:` lines in the `audit_control` file on each system before the system is rebooted.

Note – Administrators should understand that the Trusted Solaris environment only records the security-relevant events that it is configured to record (that is, by preselection). Therefore any subsequent audit can only consider the events recorded. If auditing is not configured to record the security-relevant events for the particular system environment in which it operates, it will not be possible to audit. This may mean that attempts to breach the security of the system go undetected, or that the administrator is unable to detect the user responsible for an attempted breach of security. Administrators should regularly analyze audit trails to check for breaches of security.

System Administrator’s Audit Setup Tasks

TABLE 2–1 Basic Auditing Setup by the System Administrator

Task	For the procedure, see…
Create audit partitions	“To Create Dedicated Audit Partitions” on page 47
Create audit administration server	*Trusted Solaris Installation and Configuration* or *Trusted Solaris Administrator’s Procedures*
Install audit file servers	Plan to install them before audit clients
Create files directory	“To Create an Audit Directory” on page 51
Export audit partitions (networks only)	“To Share an Audit File System” on page 51
Create the audit_warn alias	“To Warn of Audit Trouble” on page 56
Mount audit partitions (networks only)	“To Mount an Audit File System” on page 52

Security Administrator's Audit Setup Tasks - Basic

TABLE 2–2 Basic Auditing Setup by the Security Administrator

Task	For the procedure, see…
On first system	
Edit audit_control file	"To Set Audit Flags" on page 54
	"To Reserve Free Space on an Audit File System" on page 53
	"To Specify the Audit File Storage Locations" on page 53
Set filesystem security attributes	"To Protect an Audit File System" on page 50
	"To Protect an Audit File System" on page 50
Edit audit_startup file	"To Set Audit Policy Permanently" on page 56
Copy for distribution (networks only)	"To Distribute Audit Configuration Files" on page 57
Per user	
Set audit flags in Users Audit tab	"To Set User Exceptions to the Audit Flags" on page 55

Security Administrator's Audit Setup Tasks - Advanced

TABLE 2–3 Advanced Auditing Setup by the Security Administrator

Task	For the procedure, see…
On first system	
Edit audit_event file	"To Add Audit Events" on page 60
	"To Change Event-Class Mappings" on page 61
Edit audit_class file	"To Add Audit Classes" on page 59
Copy for distribution (networks only)	"To Distribute Audit Configuration Files" on page 57

Audit Shutdown and Startup (Tasks)

The following procedures describe how to enable and disable auditing for one or more systems. The commands should be run only on a diskfull computer, and never on a diskless client.

Auditing tasks require commands and actions that are limited to particular roles and particular labels. Read each task for the administrative role that can perform it, and the label required. See “To Execute Commands that Require Privilege” on page 49 for how to assume a role and open a privileged shell.

▼ To Disable Auditing

1. **As role secadmin, at label `admin_low`, open the script** `/etc/init.d/audit` **using the Admin Editor.**

Note – This should be done *only if* auditing is not a site security requirement, or in cases of audit file overflow. The security administrator is responsible.

---

2. **Comment out the start script:**

```
...
# Start the audit daemon
#  if [ -f /etc/security/audit_startup ] ; then
#  echo ”starting audit daemon”
#  /etc/security/audit_startup
#  /usr/sbin/auditd &
#  fi
...
```

3. **Write and quit the file.**

4. **Open the script** `/etc/init.d/drvconfig` **using the Admin Editor.**

5. **Add the following lines to the end of the file:**

```
# Disable auditing
#
/usr/bin/adb -wk /dev/ksyms /dev/mem > /dev/null <<end
audit_active/W 0
end
```

6. **Prevent spurious messages about the audit daemon at shutdown by commenting out the stop script in** `/etc/init.d/audit`**:**

   ```
   ...
   # Stop the audit daemon

   #      if [ -f /etc/security/audit_startup ] ; then
   #             /usr/sbin/audit -t
   #      fi
   ```

7. **Write and quit the file.**

8. **For the changes to take effect, reboot.**

   Note – A user or role requires authorization to shut down the computer.

   ---

   a. **Choose Shut Down from the TP (Trusted Path) menu and confirm the shutdown.**

   b. **Enter** `boot` **at the ok prompt or** `b` **at the > prompt:**

   ```
   Type help for more information
   <#2> ok boot
   Type b (boot), c (continue), or n (new command mode)
   > b
   ```

To Enable Auditing

By default, auditing is enabled. If you have disabled auditing, enable it by reversing the above procedure.

1. **As role secadmin, at label** `admin_low`**, open the script** `/etc/init.d/audit` **using the Admin Editor.**

2. **Remove the comments from the audit start script:**

   ```
   ...
   # Start the audit daemon
     if [ -f /etc/security/audit_startup ] ; then
            echo "starting audit daemon"
            /etc/security/audit_startup
            /usr/sbin/auditd &
     fi
   ...
   ```

3. **Write and quit the file.**

4. **Enable the audit daemon to exit gracefully at shutdown by removing the comments in the stop script in `/etc/init.d/audit`:**

   ```
   ...
   # Stop the audit daemon
     if [ -f /etc/security/audit_startup ] ; then
       /usr/sbin/audit -t
       fi
   ```

5. **Write and quit the file.**

6. **Open the script `/etc/init.d/drvconfig` using the Admin Editor.**

7. **Comment out the Disable auditing lines:**

   ```
   # Disable auditing
   #
   # /usr/bin/adb -wk /dev/ksyms /dev/mem > /dev/null <<end
   # audit_active/W 0
   # end
   ```

8. **Write and quit the file.**

9. **For the changes to take effect, reboot using the Shut Down menu item from the TP (Trusted Path) menu.**

Basic Audit Setup (Tasks)

The following procedures describe how to set up auditing for one or more systems.

▼ To Create Dedicated Audit Partitions

- **During installation, the install team creates dedicated audit partition(s) when formatting the disks.**

  Use the naming convention `/etc/security/audit/`*sytem_name*(.*n*)

  A diskfull computer should have at least one local audit directory, which it can use as a directory of last resort, if unable to communicate with the audit server.

  See “Audit Storage” on page 22 for an explanation of the naming convention.

On an audit file server, most partitions hold audit files, as is shown in the following example of the `egret` audit file server:

Disk	Slice	Mount point	Size
c0t2d0	s0	/etc/security/audit/egret	1.0 GB
	s1	/etc/security/audit/egret.1	.98 GB
	s2	entire disk	1.98 GB
c0t2d1	s0	/etc/security/audit/egret.2	502 MB
	s1	/etc/security/audit/egret.3	500 MB
	s2	entire disk	1002 MB

Note – Another disk holds `egret`'s `/` (root) and `/swap` partitions.

On a diskfull computer, including the audit administration server, at least one partition should be dedicated to local audit files, as is shown in the following example of the system `willet`:

Disk	Slice	Mount point	Size (MB)
c0t3d0	s0	/	70
	s1	swap	180
	s2	entire disk	1002
	s3	/usr	350
	s4	/etc/security/audit/willet	202
	s7	/export/home	200

Hints

A rule of thumb is to assign 200 MB of space for each system. However, the disk space requirements at your site will be based on how much auditing you perform and may be far greater than this figure.

Fewer and large partitions are more efficient than more and smaller ones.

Note – To add a disk to hold audit partitions after installing the system, see the Solaris 8 *System Administration Guide, Volume II*. To protect the disks with Trusted Solaris security attributes, see *Trusted Solaris Administrator's Procedures*.

▼ To Execute Commands that Require Privilege

Most commands for setting up auditing require the use of a profile shell, where commands can run with privilege. Auditing also requires the use of actions in the System_Admin folder and the Solaris Management Console action in the Application Manager.

1. **Log in to the computer as yourself.**

   a. **Enter your user name and press the Return key.**

   If the system is protected against anyone logging in, the Enable Logins dialog is displayed.

   b. **If you are authorized to enable logins, click the Yes button after Login:.**

   If you are not authorized to enable logins, ask the administrator to enable logins.

   c. **Enter your password and click OK.**

   You are presented with the message of the day and a label builder screen. In a single-label system, the screen describes your session label. In a multilabel system, it presents you with a label builder to choose your session clearance.

   d. **Accept the default unless you have a reason not to.**

   Press the Return key or click the OK button and be logged in.

2. **Assume an administrative role that you have been assigned.**

   a. **Click the right mouse button in the middle of the Front Panel.**

   b. **Choose Assume** *administrative* **Role from the menu.**

   c. **At the password prompt, enter the password for that role.**

▼ To Remove Free Space (Optional)

1. **As role admin, at label `admin_low`, unmount the audit partitions from the system by running the `umount`(1M) command in a profile shell.**

   For example, on the audit file server `egret`:

```
egret$ umount /etc/security/audit/egret
egret$ umount /etc/security/audit/egret.1
egret$ umount /etc/security/audit/egret.2
egret$ umount /etc/security/audit/egret.3
```

2. **Reduce reserved filesystem space on each partition to 0% with the command** `tunefs -m 0`**.**

   The security administrator sets the reserved filesystem space (called the minfree limit) in the `audit_control`(4) file.

   For example, on the audit file server `egret`:

   ```
   egret$ tunefs -m 0 /etc/security/audit/egret
   egret$ tunefs -m 0 /etc/security/audit/egret.1
   egret$ tunefs -m 0 /etc/security/audit/egret.2
   egret$ tunefs -m 0 /etc/security/audit/egret.3
   ```

   Similarly, on the system `willet`:

   ```
   willet$ umount /etc/security/audit/willet
   willet$ tunefs -m 0 /etc/security/audit/willet
   ```

   See the `tunefs`(1M) man page for more information on the advantages and disadvantages of tuning a file system.

▼ To Protect an Audit File System

1. **As role secadmin, at label `admin_low`, set the appropriate file permissions on every audit file system while the file system is unmounted.**

   For example, on the audit file server `egret`:

   ```
   egret$ chmod -R 750 /etc/security/audit/egret
   egret$ chmod -R 750 /etc/security/audit/egret.1
   egret$ chmod -R 750 /etc/security/audit/egret.2
   egret$ chmod -R 750 /etc/security/audit/egret.3
   ```

   On the system `willet`:

   ```
   willet$ chmod -R 750 /etc/security/audit/willet
   ```

2. **As role secadmin, at label `admin_high`, set any Trusted Solaris security attribute defaults required by your site security policy on every audit file system while the file system is unmounted.**

   To run the command at the label `admin_high`, you must create an `admin_high` workspace. Follow the procedure in "To Create an Admin_High Workspace" on page 63.

   For example, the following command on the audit file server `egret` should be repeated for all of its audit partitions:

   ```
   egret$ setfsattr -s “[admin_high]” /etc/security/audit/egret
   ```

   On the system `willet`:

```
willet$ setfsattr -s "[admin_high]" /etc/security/audit/willet
```

The -s option sets the partition's default sensitivity label for the audit files. See the setfsattr(1M) man page for more information.

Note – The local audit file systems must already be in the host's /etc/vfstab file.

▼ To Create an Audit Directory

1. **As admin, at label admin_high, remount the local audit file systems.**

   Follow the procedure in "To Create an Admin_High Workspace" on page 63 to get an admin_high process.

   For example, on the audit file server egret:

   ```
   egret$ mount /etc/security/audit/egret
   egret$ mount /etc/security/audit/egret.1
   egret$ mount /etc/security/audit/egret.2
   egret$ mount /etc/security/audit/egret.3
   ```

   Similarly, on the system willet:

   ```
   willet$ mount /etc/security/audit/willet
   ```

2. **Create a directory named files at the top of each mounted audit partition.**

   For example, on the audit file server egret:

   ```
   egret$ mkdir /etc/security/audit/egret/files
   egret$ mkdir /etc/security/audit/egret.1/files
   egret$ mkdir /etc/security/audit/egret.2/files
   egret$ mkdir /etc/security/audit/egret.3/files
   ```

   On the system willet:

   ```
   willet$ mkdir /etc/security/audit/willet/files
   ```

▼ To Share an Audit File System

1. **In the role admin at label admin_low, open the Trusted Solaris Management Console, Scope=files toolbox.**

2. **Navigate to the Storage node, then the Mounts and Shares tool, and double-click the Shares tool.**

3. **Enter every local audit file system in the local host's dfstab(4) file.**

   Follow the online help to share the /etc/security/audit/*hostname* directory.

   For example, the audit file server egret has the following entries:

```
share -F nfs -o ro -d "local audit files" /etc/security/audit/egret
share -F nfs -o rw=willet:audubon -d "audit files" /etc/security/audit/egret.1
share -F nfs -o rw=grebe:audubon -d "audit files" /etc/security/audit/egret.2
share -F nfs -o rw=sora:audubon -d "audit files" /etc/security/audit/egret.3
```

The system `willet` has the following entry:

```
share -F nfs -o ro -d "local audit files" /etc/security/audit/willet
```

▼ To Mount an Audit File System

1. **As role admin at label `admin_low`, on `audubon`, the audit administration server, create a mount point for every audit directory in the Trusted Solaris network.**

   For example, on the audit administration server `audubon`:

   ```
   audubon$ mkdir /etc/security/audit/willet
   audubon$ mkdir /etc/security/audit/egret
   audubon$ mkdir /etc/security/audit/egret.1
   ...
   ```

2. **As role admin, at label `admin_low`, enter every audit partition on the network in the audit administration server's `vfstab`(4) file.**

   Mount audit directories with the read-write (`rw`) option. Mount remote partitions using the `soft` option.

   a. **Click the Application Manager, double-click the System_Admin folder, and double-click the Set Mount Points action.**

   b. **Enter the mount points in the `vfstab`(4) file.**

   The following shows part of the `vfstab` file on audubon:

   ```
   # egret is the main audit file server
   egret:/etc/security/audit/egret - /etc/security/audit/egret nfs - yes bg,soft,nopriv
   egret:/etc/security/audit/egret.1 - /etc/security/audit/egret.1 nfs - yes bg,soft,nopriv
   egret:/etc/security/audit/egret.2 - /etc/security/audit/egret.2 nfs - yes bg,soft,nopriv
   egret:/etc/security/audit/egret.3 - /etc/security/audit/egret.3 nfs - yes bg,soft,nopriv
   willet:/etc/security/audit/willet - /etc/security/audit/willet nfs - yes bg,soft,nopriv
   ...
   ```

3. **On each system, create the mount points for the remote audit file servers' partitions that are used by the system, and enter them in the `vfstab`(4) file. Do this as role admin, at label `admin_low`.**

   For example, to create the mount points on the system `willet`:

   ```
   willet$ mkdir /etc/security/audit/egret
   willet$ mkdir /etc/security/audit/audubon.2
   ```

a. **Click the Application Manager, double-click the System_Admin folder, and double-click the Set Mount Points action.**

b. **Enter the mount points in the `vfstab`(4) file.**

The following shows part of the `vfstab` file on `willet`:

```
# egret is the main audit file server
egret:/etc/security/audit/egret - /etc/security/audit/egret nfs - yes bg,soft,nopriv
# audubon is the audit administration server
audubon:/etc/security/audit/audubon.2 - /etc/security/audit/audubon.2 nfs - yes nopriv
```

To Reserve Free Space on an Audit File System

1. **As role secadmin, at label `admin_low`, enter reserve free space in the `audit_control`(4) file.**

   a. **Open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Control action.**

2. **Enter a value between 10 and 20 on the `minfree:` line.**

```
dir:/var/audit
flags:
minfree:20
naflags:
```

3. **Write the file and quit the editor.**

To Specify the Audit File Storage Locations

1. **As role secadmin, at label `admin_low`, enter audit storage locations in the `audit_control` file.**

   a. **Open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Control action.**

2. **On the first system installed, enter its local audit file system as the value of the `dir:` line.**

   The following shows the `audit_control` file for `grebe`, the NIS+ root master.

```
dir:/etc/security/audit/grebe/files
flags:
minfree:20
```

```
naflags:
```

3. **When the audit file servers have been installed and configured, add their (mounted) filesystem names plus their top-level directory, `files` to the `dir:` entry.**

   The mounted file systems are listed before the system's local file system, as in:

   ```
   dir:/etc/security/audit/egret/files
   dir:/etc/security/audit/egret.1/files
   dir:/etc/security/audit/grebe/files
   flags:
   minfree:20
   naflags:
   ```

4. **Write the file and exit the editor.**

5. **As role secadmin in an `admin_high` profile shell, execute the `audit -s` command to have the audit daemon re-read the `audit_control` file and write audit records to the designated directory.:**

   ```
   $ audit -s
   ```

   By default, the audit records have been stored in `/var/audit`. The audit records will now be stored in the first directory in the `audit_control` file.

▼ To Set Audit Flags

1. **As role secadmin, at label `admin_low`, enter system-wide audit flags in the `audit_control`(4) file.**

   a. **Open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Control action.**

2. **Enter the `na` class in the `naflags:` line if your site is auditing non-attributable events.**

   ```
   dir:/etc/security/audit/egret/files
   dir:/etc/security/audit/egret.1/files
   dir:/etc/security/audit/grebe/files
   flags:
   minfree:20
   naflags:na
   ```

3. **Enter other classes in the `flags:` line if your system is auditing user-level events.**

   ```
   dir:/etc/security/audit/egret/files
   dir:/etc/security/audit/egret.1/files
   dir:/etc/security/audit/grebe/files
   ```

```
flags:lo,ad,-all,^-fc
minfree:20
naflags:na
```

See “Sample audit_control File” on page 25 for an explanation of the syntax of the audit flags’ fields.

4. **Write the file and exit the editor.**

Note – On a distributed system, the audit flags in the `audit_control` file must be identical on every host on the network. See “To Distribute Audit Configuration Files” on page 57 for a process to distribute master copies of files to all hosts on the network.

To Set User Exceptions to the Audit Flags

The security administrator at label `admin_low`, enters user exceptions to system-wide audit flags in the user’s Audit tab.

1. **In the the role secadmin, launch the Solaris Management Console from the Application Manager and choose the toolbox appropriate for your site.**

2. **Under the User Accounts node, select a user.**

3. **In the user’s Audit tab, enter the user exceptions, write the file, and exit the editor.**

   Follow the online help for assistance. The following example shows the format of the `audit_user` file.

   For example, the following `audit_user` entry audits the role root for logins and logouts, and never audits the `fc` class, even if it is being audited for the system. The `jane` entry audits her for all flags specified in the `audit_control` file except for successful `file_read` events. Null events, `no`, are never audited.

```
# User Level Audit User File
#
# File Format
#
#       username:always:never
#
root:lo:no,fc
jane:all,^+fr:no
```

▼ To Warn of Audit Trouble

1. **As role admin, at label `admin_low`, create a mail alias to warn of audit trouble.**

   a. **If you are running a name service, on the master server of the name service, launch the Solaris Management Console from the Application Manager.**

   b. **Choose the toolbox that your site uses for administration, and select the Users node.**

   c. **Double-click the Mailing Lists node.**

   d. **From the Action menu, choose Add mailing list.**

2. **Create an alias called `audit_warn` for notifying its members of audit trouble.**

   For example, this `audit_warn` alias emails the security administrator and the system administrator when the auditing subsystem needs attention.

   Mailing List Name: **`audit_warn`**
   Mailing List Recipients: **`secadmin@grebe,admin@grebe`**

▼ To Set Audit Policy Permanently

1. **As role secadmin, at label `admin_low`, enter permanent audit policy in the `audit_startup`(1M) file.**

   a. **Open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Startup action.**

2. **Create a script that calls the `auditconfig`(1M) command with policy options.**

   The sample `audit_startup`(1M) script below adds ACLs to audit records, halts the computer when its audit file systems are full, and at startup, prints the current audit policy to standard i/o.

```
#!/bin/sh
auditconfig –setpolicy +slabel,+acl
auditconfig –setpolicy +ahlt
auditconfig –getpolicy
```

3. **Write the file and exit the editor**

Caution – To run auditing in an evaluated configuration, the cnt policy cannot be turned on; the ahlt policy (the default) cannot be turned off.

▼ To Distribute Audit Configuration Files

In the Trusted Solaris 8 4/01 release, the audit_user file can be a NIS map or a NIS+ table, and does not need to be copied to each host. Sites that do not use a name service will want the same audit_user file on every system. If the site modifies the file on any system, it should be copied to all hosts.

1. **During installation, as root, at label admin_low, create a directory on the first installed workstation to hold copies of the audit configuration files customized for your site.**

   For example, on grebe, the first host in a network:

   ```
   # mkdir /export/home/tmp
   ```

2. **Copy the modified files from the /etc/security directory to the /export/home/tmp directory.**

   ```
   # cp /etc/security/audit_control /export/home/tmp/audit_control
   # cp /etc/security/audit_warn /export/home/tmp/audit_warn
   # cp /etc/security/audit_startup /export/home/tmp/audit_startup
   # cp /etc/security/audit_event /export/home/tmp/audit_event
   ```

   The directory would include your customized versions of audit_control, audit_startup, and audit_warn. If you have modified event-to-class mappings, it would include audit_event; if you have created new audit classes, it would include audit_class. It would not include audit_data.

3. **Allocate the tape or diskette device.**

   Follow the procedure in "To Allocate and Deallocate Devices" on page 58.

4. **Run the tar(1) command to copy the contents of the /export/home/tmp directory to a tape or diskette.**

   - To copy to tape:

     ```
     # cd /export/home/tmp
     # tar cv audit_control audit_warn audit_startup audit_event
     ```

   - To copy to diskette:

     ```
     # cd /export/home/tmp
     # tar cvf /dev/diskette \
     audit_control audit_warn audit_startup audit_event
     ```

5. **Deallocate the tape or diskette device and follow the instructions.**

   Follow the procedure in "To Deallocate a Device" on page 59.

6. **As root, at label `admin_low`, as each new host is configured, copy the files from the tape or diskette to the correct directory on the new system.**

   a. **Prepare the directory for the new files.**

   ```
   # cd /etc/security
   # mv audit_control audit_control.orig
   # mv audit_startup audit_startup.orig
   # mv audit_warn audit_warn.orig
   # mv audit_event audit_event.orig
   ```

   b. **Allocate the appropriate device at the label `admin_low`.**

   Follow the procedure in "To Allocate and Deallocate Devices" on page 58.

   c. **Copy the files.**

   - To copy from tape:

     ```
     # tar xv audit_control audit_warn audit_startup audit_event
     ```

   - To copy from diskette:

     ```
     # tar xvf /dev/diskette \
     audit_control audit_warn audit_startup audit_event
     ```

   d. **Deallocate the device.**

   Follow the procedure in "To Deallocate a Device" on page 59.

7. **As role admin, at label `admin_low`, modify the `audit_control` file on each new system with that system's remote and local audit file systems.**

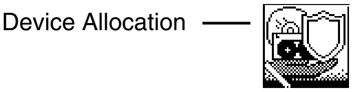
▼ To Allocate and Deallocate Devices

The Device Manager allocates and deallocates devices.

1. **In an administrative role workspace at the label required, click the left mouse button on the triangle above the Style Manager icon on the Front Panel.**

   The Tools subpanel is displayed.

2. **Click the Device Manager icon once.**



3. **Double-click the device to be allocated.**

   mag_tape_0 allocates a tape device. floppy_0 allocates a diskette.

4. **Click OK in the label builder that appears.**

   The file you load will be labeled `admin_low`.

▼ To Deallocate a Device

1. **Go to the workspace where the Device Manager was allocated.**

2. **Double-click the device to deallocate it.**

   A window appears listing devices being deallocated.

3. **When prompted, remove the tape or diskette from the drive and label it appropriately.**

4. **Click the top left button and select Close to close the Device Allocation Manager window.**

Advanced Audit Setup (Tasks)

The following procedures describe how to modify the default audit classes and audit events, and to set a public object bit on files and folders to reduce unnecessary auditing.

▼ To Add Audit Classes

1. **As role secadmin, at label `admin_low`, add audit classes in the** `audit_classes` **file.**

   a. **Open the System_Admin folder from the Application Manager.**

b. **Double-click the Audit Classes action.**

2. **Add the classes you planned in "Planning a Site-Specific Event-to-Class Mapping" on page 38, write the file, and exit the editor.**

Caution – Do not reassign the hexadecimal numbers already in use.

---

3. **As role secadmin, at label `admin_low`, open the Audit Events action to add the new class to each event in the new class.**

   For events in more than one class, use a comma (no space) to delimit the classes.

4. **Write the file and exit the editor.**

5. **Make any changes to `audit_control`(4) and `audit_user`(4) to audit the events in the new classes.**

   See "To Set Audit Flags" on page 54 and "To Set User Exceptions to the Audit Flags" on page 55 for details of the procedures.

Note – On a distributed system, the `audit_class`, `audit_event`, `audit_startup`, and `audit_user` files must be identical on every host on the network. See "To Distribute Audit Configuration Files" on page 57 for a process to distribute master copies of files to all hosts on the network.

---

6. **Reboot, or as secadmin in an `admin_low` profile shell, run the `auditconfig`(1M) command with appropriate options.**

   In the following example, the audit session ID is 159, and the new classes are `gr` (for graphic applications) and `db` (for databases applications).

```
$ auditconfig -setsmask 159 gr,db
```

▼ To Add Audit Events

1. **As role secadmin, at label `admin_low`, add audit events in the `audit_event`(4) file.**

   a. **Open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Events action.**

2. **Add the events you planned in "Planning a Site-Specific Event-to-Class Mapping" on page 38, write the file, and exit the editor.**

   For events in more than one class, use a comma (no space) to delimit the classes.

Note – Third-party applications can use the event numbers 32768 through 65536 only. See for more information about event number assignment.

3. **Make any changes to `audit_control`(4) and `audit_user`(4) to audit the events in the new classes.**

   See "To Set Audit Flags" on page 54 and "To Set User Exceptions to the Audit Flags" on page 55 for details of the procedures.

Note – On a distributed system, the `audit_class`, `audit_event`, `audit_startup`, and `audit_user` files must be identical on every host on the network. See "To Distribute Audit Configuration Files" on page 57 for a process to distribute master copies of files to all hosts on the network.

4. **Reboot, or as secadmin in an `admin_low` profile shell, run the `auditconfig`(1M) command with appropriate options.**

   In the following example, the audit session ID is 159, and the new events are in the classes `gr` (for graphic applications) and `db` (for databases applications).

```
$ auditconfig -setsmask 159 gr,db
```

▼ To Change Event-Class Mappings

1. **Change event-class mappings in the `audit_control`(4) file.**

   a. **As role secadmin, at label `admin_low`, open the System_Admin folder from the Application Manager.**

   b. **Double-click the Audit Events action.**

2. **Edit the file to change the class mapping for each event to be changed, write the file, and exit the editor.**

   If you are changing events above number 2048, this is all you need to do.

Note – On a distributed system, the audit_class, audit_event, audit_startup, and audit_user files must be identical on every host on the network. See "To Distribute Audit Configuration Files" on page 57 for a process to distribute master copies of files to all hosts on the network.

---

3. **If you modify a kernel event mapping (numbers 1 to 2047), restart auditing by doing one of the following:**
   - Reboot the system, or
   - As role secadmin, at label admin_low, change the runtime event-to-class mappings:

     ```
     $ auditconfig -conf
     ```

▼ To Set Public Object Bit

Setting the public object bit can reduce the size of the audit trail when the audit record includes successful accesses of files or directories. Successful viewing, listing, or listing of a file or directory's attributes will not be written to the audit record when the file's public object bit is set.

● **As role secadmin, at label admin_low, set the public object bit on a local directory of publicly accessible files using the setfattrflag(1) command with the -p 1 option.**

The following command sets the public object bit on the /etc directory. A search of the /etc directory, or a read of files in the /etc directory will not result in an audit record.

```
$ setfattrflag -p 1 /etc
$ getfattrflag /etc
 Multilevel directory: no
 Single level directory: no
          Public object: yes
```

Dynamic Auditing (Tasks)

Dynamic controls apply to one system at a time, since the audit command only applies to the current system where you are logged in. Use dynamic controls to test auditing on a system (estimate volume of records, for example), or to add an auditing flag without having to reboot the computer. However, if you make dynamic changes on one system for other than testing purposes, you should make the changes on all systems.

Note – The following procedures work only when auditing is enabled.

▼ To Determine Current Audit Policy

The auditconfig(1M) command enables an appropriately configured role to determine audit policy and to see what policies can be set. If your role is not configured to determine the policy, or if auditing is turned off, the command auditconfig -getpolicy returns an error. The following example was run by the role secadmin, at label admin_low:

```
$ auditconfig -getpolicy
    audit policies = none
$ auditconfig -lspolicy
policy string   description:
    arge    include exec environment args in audit recs
    argv    include exec args in audit recs
    cnt     when no more space, drop recs and keep a count
    group   include supplementary groups in audit recs
    seq     include a sequence number in audit recs
    trail   include trailer tokens in audit recs
    path    allow multiple paths per event
    acl     include ACL information in audit recs
    ahlt    halt machine if we can't record an async event
    slabel  include sensitivity labels in audit recs
    passwd  include cleartext passwords in audit recs
    windata_down include downgraded information in audit recs
    windata_up   include upgraded information in audit recs
    all     all policies
    none    no policies
```

▼ To Create an Admin_High Workspace

To label files admin_high, to move files to an admin_high directory, to reset the audit daemon, and to make other changes in auditing requires an admin_high process. An admin_high process starts from an admin_high workspace.

1. **Click the right button on the Front Panel and choose Assume secadmin Role from the menu.**

   A secadmin role workspace becomes the current workspace.

2. **In the current workspace, click the right button on the workspace name (secadmin) button and choose Change Workspace SL from the menu.**

3. **In the label builder, click the ADMIN_HIGH button, then click OK.**

   The color of the workspace button turns to black, indicating an `admin_high` workspace. An `admin_high` workspace is available only to an administrative role.

To Set Audit Policy Temporarily

The `auditconfig` command enables you to change audit policy, such as whether to include acl information in the audit record. Since the audit policy variable is a dynamic kernel variable, the policy that you set is in effect until the computer next boots. See the `auditconfig`(1M) man page for a list of audit policy parameters.

The security administrator sets or changes audit policy. Policy changes are set at the label `admin_low`.

- **To set policies in one invocation of the command, or to override all current policies, separate the policies with commas (no spaces):**

```
$ auditconfig -setpolicy trail,seq
$ auditconfig -getpolicy
    audit policies = trail,seq
$ auditconfig -setpolicy argv,acl
$ auditconfig -getpolicy
    audit policies = argv,acl
```

- **To add policies to the current policies, preface each added policy with a plus (+):**

```
$ auditconfig -setpolicy trail,seq
$ auditconfig -getpolicy
    audit policies = trail,seq
$ auditconfig -setpolicy +argv
$ auditconfig -setpolicy +acl
$ auditconfig --getpolicy
    audit policies = seq,trail,argv,acl
```

- **To remove policies from the current policies, preface each policy to be removed with a minus (–):**

```
$ auditconfig -setpolicy trail,seq
$ auditconfig -getpolicy
    audit policies = trail,seq
$ auditconfig -setpolicy  -seq
$ auditconfig -getpolicy
    audit policies = trail
```

In the examples above, the `trail` and `seq` tokens are added to debug audit trail discrepancies. To set policies permanently, enter the `auditconfig` command in the `audit_startup`(1M) script. See "To Set Audit Policy Permanently" on page 56 for how to edit the script.

Caution – To run auditing in an evaluated configuration, the `cnt` policy cannot be turned on; the `ahlt` policy (the default) cannot be turned off.

▼ To Change Audit Flags Dynamically

The `auditconfig`(1M) command enables you to change audit flags dynamically, such as adding extra flags to a user, a session, or a process while the user, session, or process is active. Since the flags are added dynamically, they are in effect until the user logs out, the session ends, or the process ends.

The security administrator sets or changes audit policy. Policy changes are set at the label `admin_low`.

- **To set a particular user to be additionally audited for successful file reads:**

```
$ auditconfig -setumask audit_user_id +fr
```

- **To set a particular session to be additionally audited for failed file attribute access:**

```
$ auditconfig -setsmask audit_session_id -fa
```

- **To set a particular process to be additionally audited for successful and unsuccessful file attribute modifications:**

```
$ ps -ef | grep application-to-be-monitored
$ auditconfig -setpmask process_id fm
```

▼ To Stop the Audit Daemon

Only one audit daemon may run at a time. An attempt to start a second one will result in an error message, and the new one will exit. If there is a problem with the audit daemon, terminate the audit daemon gracefully, then restart it manually.

- **To stop the audit daemon in event of trouble, as role secadmin, at label `admin_high`:**

```
$ audit -t
```

This is not recommended. Audit records may be lost.

▼ To Start the Audit Daemon

The audit daemon starts when the computer is brought up to multiuser mode, and restarts when the audit daemon is instructed by the `audit -s` command to reread an audit configuration file.

- **To restart the audit daemon in event of trouble or a change to an audit configuration file, as role secadmin, at label `admin_high`:**

  ```
  $ audit -s
  ```

  The pointer may be reset to the beginning of the list of audit directories when the administrator enters the `audit -s` command.

▼ To Send Audit Records to a New Audit File

- **To change the current audit file for audit records being generated on the system, as role secadmin at label `admin_high`:**

  ```
  $ audit -n filename
  ```

  The new file is created in the same directory as the current file. The directory must be able to contain files labeled `admin_high`.

CHAPTER 3

Audit Trail Management and Analysis

The tools described in this chapter manage the audit files generated on a system or on a distributed system. Managing the audit trail involves file tasks and interpretive tasks. File tasks handle disk space issues, such as combining multiple audit files into one and renaming files. Interpretive tasks cover audit analysis, such as selecting audit records based on audit event, user, host machine, and time of day. Sophisticated postprocessing using shell scripts can create auditing reports.

The chapter includes procedures in the following areas:

- "Using the auditreduce and praudit Commands" on page 75
- "Audit Files Backup and Recovery" on page 76

The Audit Trail

The collection of all audit files in a distributed system is called the *audit trail*. The audit trail may consist of audit files in several audit directories, or an audit directory may contain several audit trails. Most often the audit directories will be separate audit file system partitions. Even though they can be included in other file systems, this is not recommended.

Audit files by default are stored in the *audit root directory*, defined as `/etc/security/audit/*/files`. Once each system has created an audit root directory, and the directories have been mounted (with mount points that follow the naming convention) on the audit administration server, the management tools, `auditreduce` and `praudit`, can examine the entire audit trail. See "Basic Audit Setup (Tasks)" on page 47 for how to set up an audit trail.

Even though it is possible to locate audit directories within other file systems that are not dedicated to auditing, this is not recommended. If other factors dictate placing audit files on a partition not dedicated to auditing, only do so for directories of last resort. Directories of last resort would be directories where audit files would be written only when there is no other suitable directory available. One other scenario where locating audit directories outside of dedicated audit file systems could be acceptable would be in an environment where auditing is optional, and where it is more important to make full use of disk space than to keep an audit trail. Putting audit directories within other file systems is unworkable in a security-conscious production environment.

How the Audit Trail Is Created

The *audit trail* is created by the audit daemon, `auditd`(1M). The audit daemon starts on each system when the system is booted. After `auditd` starts, it is responsible for collecting the audit trail data and writing the audit records into *audit files*, which are also called *audit log files*. See the `audit.log`(4) man page for a description of the file format.

The audit daemon runs as root. All files it creates are owned by root. Even when `auditd` has no classes to audit, `auditd` continuously operates, looking for a place to put audit records. The `auditd` operations continue even if the rest of the system's activities are suspended because the kernel's audit buffers are full. The audit operations can continue because `auditd` is not audited.

Audit Record Format

Audit files consist of self-contained audit records of user-level and kernel-level events that have been preselected for auditing by the security administrator. An *audit record* consists of a sequence of *audit tokens*, each of which describes an attribute of the event being audited. Each auditable event in the system generates a particular type of audit record. The audit record for each event has certain tokens within the record that describe the event. An audit record does not describe the audit event class to which the event belongs; that mapping is determined by an external table, the `/etc/security/audit_event` file.

Each audit token starts with a one-byte token type, followed by one or more data elements in an order determined by the type. The different audit records are distinguished by event type and different sets of tokens within the record. Some tokens, such as the `text` token, contain only a single data element, while others, such as the `process` token, contain several (including the audit user ID, real user ID, and effective user ID).

Audit records are stored and manipulated in binary form; however, the byte order and size of data are predetermined to simplify compatibility between different systems.

"Audit Token Structure" on page 120, gives a detailed description of each data element in each token and shows sample output. "Audit Records" on page 149 lists all the audit records generated by Trusted Solaris 8 4/01 auditing. The records are listed alphabetically by kernel event and by user event. Tables that connect audit events to their audit records are found in "Audit Events Listed by Audit Class" on page 91.

Order of Audit Tokens

Each audit record begins with a `header` token and ends (optionally) with a `trailer` token. One or more tokens between the header and trailer describe the event. For user-level and kernel events, the tokens describe the process that performed the event, the objects on which it was performed, and the objects' attributes, such as the owner or mode.

For example, the AUE_LSTAT kernel event, whose audit record is described in Table B–71, has the following tokens:

- header
- path
- attribute (optional)
- privilege (optional)
- subject
- return

If the `trail` policy has been turned on using the `auditconfig` command, the `trailer` token appears in the audit record after the `return` token.

Human-Readable Audit Record Format

This section provides examples of audit records in text format. Audit records are stored in binary format. Running the binary records through the `praudit` command produces text output, which can be sent to standard output, a printer, or a scripting program to produce reports. For a complete description of `praudit`, see the `praudit`(1M) man page. For an example of a scripting program, see "To Perform Selections Using a praudit Script" on page 80.

Reading an Audit Token

The following examples of a header token show the form that praudit produces by default. Examples are also provided of raw (-r) and short (-s) options.

Every audit record begins with a header token. The header token gives information common to all audit records. When displayed by praudit in default format, a header token looks like the following example from ioctl():
header,240,1,ioctl(2),,Thurs Sept 7 16:11:44 2000, + 270 msec

The fields are:

- A token ID, here in text form, **header**
- The record length in bytes, including the header and trailer tokens, here **240**
- An audit record structure version number, here, version **1**
- An event ID identifying the type of audit event, here in text form, **ioctl(2)**
- An event ID modifier with descriptive information about the event type, here the descriptive field is empty
- The time and date the record was created, here **Thurs Sept 7 16:11:44 2000, + 270 msec**

Using praudit -s, the event description (ioctl(2) in the default praudit example above) is replaced with the event name (AUE_IOCTL), like this:

```
header,240,1,AUE_IOCT
L,,Thurs Sept 7 16:11:44 2000, + 270 msec
```

Using praudit -r, all fields are displayed as numbers (that may be decimal, octal, or hex), where 20 is the header token ID and 158 is the event number for this event.

```
20,240,1,158,,699754304, + 270 msec
```

Note that praudit displays the time to millisecond resolution.

Reading an Audit Record

Every audit record contains at least the header token and one other token. For example, the audit record for the audit event AUE_login contains five tokens. See Table B–262 for a full description of its audit record format.

When displayed by praudit in default format, the audit record for AUE_login looks like this, one token per line:

```
header,90,3,login - local,,Tue Jul 8 15:12:01 1997, +520 msec,
text,emily
```

```
text,successful login
subject,emily,emily,staff,emily,staff,14094,14094,0 0 willet,
return,success,0
sequence,17
trailer,90
```

The tokens are:

- A header token
- A text token (login name)
- A text token (for success or failure)
- A subject token
- A return token

When this audit file collected records, the audit policy tokens `sequence` and `trailer` were turned on, so all audit records including this one contain the following tokens:

- A sequence token
- A trailer token

Note the following features in the audit record:

- Each user's processes is assigned a unique audit ID that stays the same even when the user ID changes (14094)
- Each session has an audit session ID (14094)
- Audit records are self-contained

Because each audit record contains an audit ID that identifies the user who generated the event, and because audit records are self-contained, you can look at individual audit records and get meaningful information without looking back through the audit trail.

Trusted Solaris audit records contain all the relevant information about an event and do not require you to refer to other audit records to interpret what occurred. For example, an audit record describing a file event contains the file's full path name starting at the root directory and a time and date stamp of the file's opening or closing.

Note – You should archive system administration files with audit file archives. Information that is referred to in the audit trail but changes as site personnel and equipment change, such as users and their UIDs, affects your ability to interpret records.

Using `praudit -l`, the audit record displays on one line, like this:

```
header,90,3,login - local,,Tue Jul 8 15:12:01 1997, +520 msec,text,emily,
text,successful login,subject,emily,emily, staff,emily,staff,14094,
14094,0 0 willet,return,success,0, sequence,17,trailer,90
```

Using `praudit -r` the audit record displays like this:

```
20,90,3,6152,0x0000,872028721,520
40,emily
40,successful login
36,6001,6001,10,6001,10,14094,14094,0 0 192.168.110.2
39,0,0
47,17
19,9
```

Audit Files

Each audit file is a self-contained collection of records; the file's name identifies the time span during which the records were generated and the system that generated them. The contents of the audit files are binary, protected at the sensitivity label `admin_high`, and accessible in a profile shell only by an administrative role with the Audit Review profile.

Audit File Naming

Audit files that are complete have names of the following form: *start-time.finish-time.system*, where *start-time* is the time of the first audit record in the audit file, *finish-time* is the time of the last record, and *system* is the name of the system that generated the file. Some examples of these names can be found in "Example of a Closed Audit File Name" on page 73.

If the audit log file is still active, it has a name of the following form: *start-time*`.not_terminated.`*system*

How Audit File Names Are Used

The file name time stamps are used by the `auditreduce` command to locate files containing records for the specific time range that has been requested. This is important because there may be a month's supply or more of audit files online, and searching them all for records generated in the last 24 hours would be expensive.

Time-Stamp Format and Interpretation

The *start-time* and *finish-time* are time stamps with one-second resolution; they are specified in Greenwich mean time. The format is four digits for the year, followed by two for each month, day, hour, minute, and second, as shown here:
YYYYMMDDHHMMSS

The time stamps are in GMT to ensure that they will sort in proper order even across a daylight saving time boundary. Because they are in GMT, the date and hour must be translated to the current time zone to be meaningful; beware of this whenever manipulating these files with standard file commands rather than with `auditreduce`.

Example of a File Name for a Still-Active File

The following shows the format of a file name of a still-active file:
YYYYMMDDHHMMSS`.not_terminated.`*hostname*

Here is an example:

```
19900327225243.not_terminated.patchwork
```

The audit log files are named by the beginning date, so the example above was started in 1997, on March 27, at 10:52:43 PM, GMT. The `not_terminated` in the file name means either that the file is still active or that `auditd` was unexpectedly interrupted. The name `patchwork` at the end is the host name whose audit data is being collected.

Example of a Closed Audit File Name

The following shows the format of the name of a closed audit log file:
YYYYMMDDHHMMSS.YYYYMMDDHHMMSS.hostname

Here is an example:

```
19970320005243.19970327225351.patchwork
```

The example above was started in 1997, on March 20, at 12:52:43 AM, GMT. The file was closed March 27, at 10:53:51 PM, GMT. The name `patchwork` at the end is the host name of the system whose audit data is being collected.

Audit Files Management

Two commands, praudit(1M) and auditreduce(1M), enable the audit reviewer to process audit records. The praudit command makes the records readable, and the auditreduce command enables selecting particular audit records and merging the records into one audit trail.

Note – The auditreduce command can only find records that have been preselected by the security administrator. Events that are not recorded in the audit trail are unavailable to postselection tools.

Merging the Audit Trail

The auditreduce command merges audit records from one or more input audit files to create a single, chronologically ordered output file. On a distributed system, the input audit files originate from different hosts. Therefore, when issued from the audit administration server, the auditreduce command treats the distributed system as if it were one system. This treatment simplifies audit administration. Coupled with backup audit partitions, the distributed system is robust in the face of system failures.

The auditreduce command also includes options for selecting sets of records to examine. For instance, records from the past 24 hours can be selected to generate a daily report; all records generated by a specific user can be selected to examine that user's activities; or all records caused by a specific event type can be selected to see how often that type occurs.

Selecting Records from the Audit Trail

Options to the auditreduce(1M) command enable you to select audit records based on file characteristics and record characteristics, as shown in the following table.

TABLE 3–1 Some Options to the auditreduce Command

Characteristic	Option(s)
Time, date (start, finish)	-d, -a, -f
Host (system) ID	-M, -h, -S
Audit class	-c

TABLE 3–1 Some Options to the auditreduce Command *(Continued)*

Characteristic	Option(s)
Audit event	-m
Audit User ID – AUID	-u
Effective and Real User ID – EUID, RUID	-e, -r
Effective and Real Group ID – EGID, RGID	-f, -g
Process ID – PID	-j
Sensitivity label	-s
Filename	*filename*

Uppercase options select operations or parameters for *files*, and lowercase options select parameters for *records*. When piped through praudit, audit files processed by the auditreduce command are readable. Otherwise, they remain in binary format.

The merging and selecting functions of auditreduce are logically independent. The auditreduce command selects messages from the input files as the records are read, before the files are merged and written to disk.

Using the auditreduce and praudit Commands

This section describes a few common uses of auditreduce and praudit to select and manage data. See the auditreduce(1M) man page for more examples.

Prerequisites for running the auditreduce and praudit commands:

- You are in an administrative role that includes the Audit Review profile. The role admin includes this profile by default.
- You are in an admin_high workspace of that role.

  To create an admin_high workspace, see "To Create an Admin_High Workspace" on page 63 in Chapter 2.
- You have launched a terminal window.

To access the audit trail for a distributed system:

- You issue the auditreduce command from the audit administration server.

Audit Files Backup and Recovery

Audit files occupy disk space. The disk space needs to be freed up in order to make space for subsequent audit files. By default, the role oper handles audit file backup via the profile Media Backup and the role admin handles audit file restore via the profile Media Restore.

Audit Analysis (Tasks)

▼ To Read a Closed Audit File

The praudit command enables you to display audit records interactively and create very basic reports. For multiple files, the input is piped from auditreduce.

- **Specify the audit file as the file argument to the** praudit **command.**

  ```
  $ praudit 19970401000000.19970601000000.grebe
  ```

  This displays audit token per line to standard output.

- **Specify the audit file as the file argument to the** praudit -l **command.**

  ```
  $ praudit -l 19970401000000.19970601000000.grebe
  ```

  This displays one audit record per line to standard output.

▼ To Read a Current Audit File

- **Use the** tail**(1) command to see what is currently being written to an active audit file.**

  ```
  $ praudit | tail -40 19970401000000.not_terminated.grebe
  ```

  This displays the latest 40 tokens that were recorded to standard output.

▼ To Display Several Audit Files as One Audit File

- **To display several audit files in chronological order in the terminal window, pipe the output of** `auditreduce` **into** `praudit`**.**

```
$ auditreduce 19970413000000.19970413235959.willet \
19970413000000.19970413235959.grebe | praudit
```

- **To display the entire audit trail in the terminal window, pipe the output of** `auditreduce` **into** `praudit`**.**

  The `auditreduce` command without options does not disturb open audit files.

```
$ auditreduce | praudit
```

▼ To Print an Audit Log

- **Use** `praudit` **with a pipe to** `lp`**, to send the output of one file to the printer.**

```
$ praudit 19970413000000.19970413235959.audubon | lp
```

- **Use** `auditreduce` **piped through** `praudit` **with a pipe to** `lp`**, to send the output of all closed audit files to the printer.**

```
$ auditreduce | praudit | lp
```

Note – In the Trusted Solaris environment, the printer must be able to accept `admin_high` print jobs.

▼ To Display User Activity on a Selected Date

- **Use the** `-d` **option to the** `auditreduce` **command to see audit information collected during a specified 24-hour period.**

  In the following example, the security administrator checks to see when a user named `doris` logged in and logged out on April 13, 1997, by requesting the `lo` message class. The short-form date is in the form *yymmdd*. (The long form is described in the `auditreduce`(1M) man page.)

```
$ auditreduce -d 970413 - -u  doris  -c  lo | praudit
```

▼ To Print User Activity on a Selected Date

- **Use the** `auditreduce` **command with a pipe through** `praudit` **to** `lp`**, to send selected output to a printer.**

Note – In the Trusted Solaris environment, the printer must be able to accept `admin_high` print jobs.

```
$ auditreduce -d 970413 -u doris -c lo | praudit | lp
```

▼ To Copy Login/Logout Messages to a Single File

In this example, login/logout messages for a particular day are summarized in a file. The target file is written in a directory other than the normal audit root.

```
$ auditreduce --c lo  -d  970413  -O  /usr/audit_summary/logins
```

The `-O` option creates an audit file in the `/usr/audit_summary` directory. The file name has 14-character timestamps for both start-time and end-time, and the suffix `logins`: `/usr/audit_summary/19970413000000.19970413235959.logins`

▼ To Display Audit Records Created Before or After a Designated Date

The *date-time* options `-b` and `-a` allow specifying records before or after a particular day and time. A day begins at *yyyymmdd*00:00:00 and ends at *yyyymmdd*23:59:59. The six parameters of a day are: year, month, day, hour, minute, and second.

The `auditreduce -a` command with the date shown in the following screen example sends all audit records created after midnight on July 15, 1997 through `praudit` to standard output.

```
$ auditreduce -a 97071500:00:00 | praudit
```

If `-a` is not specified, `auditreduce` defaults to 00:00:00, January 1, 1970.

The `auditreduce -b` command with the same date shown above sends all audit records created before midnight on July 15, 1997 through `praudit` to standard output.

```
$ auditreduce -b 97071500:00:00 | praudit
```

If `-b` is not specified, `auditreduce` defaults to the current time of day (GMT). The `-d` option selects a particular 24-hour period, as shown in "To Copy Login/Logout Messages to a Single File " on page 78.

▼ To Find an Audit Event

- **Use the message type selection for auditreduce (-m option) to find a particular audit event.**

  The -m option accepts either numeric message identifiers or AUE_xxxxx event names. The screen example below finds all kernel-level login events in the audit trail and displays them to standard output.

  ```
  $ auditreduce -m AUE_LOGIN | praudit
  ```

  The auditreduce command rejects an incorrect format, but does not describe the correct format.

▼ To Combine Selected Audit Files

Although auditreduce can do this type of combination and deletion automatically (see the -C and -D options in the auditreduce(1M) man page), it is often easier to select the files manually (perhaps with find) and use the auditreduce command to combine just the named set of files.

1. **List the audit files as arguments to the auditreduce command.**

   In the following example, a recurring job that starts a bit before midnight merges the audit files from two days before. The final time on the file is the time the job ended, here just before midnight, Greenwich Mean Time (GMT).

   ```
   $ auditreduce 19970413000000.19970413235959.grebe \
   19970413000000.19970413235959.willet \
   19970413000000.19970413235959.sora
   $ ls *audubon 19970413000000.19970414235959.audubon
   ```

2. **Delete the input files and move the output file to the audit root directory on the administration server.**

   In this example, the auditreduce(1M) command was run on the audit administration server, audubon, and then placed in its audit root directory so that future calls to auditreduce locate the file.

```
$ rm /etc/security/grebe/files/19970413000000.19970413235959.grebe
$ rm /etc/security/willet/files/19970413000000.19970413235959.willet
$ rm /etc/security/sora/files/19970413000000.19970413235959.sora
$ mv 19970413000000.19970414235959.audubon /etc/security/audit/audubon/files/
```

▼ To Reduce Audit Files

The auditreduce program can also reduce the number of records in its output file by eliminating the less interesting ones as it combines the input files.

You might use auditreduce to eliminate all except the login/logout events in audit files over a month old, assuming that if you needed to retrieve the complete audit trail you could recover it from backup tapes. The following example selects just the audit records from April 1997.

```
$ auditreduce -m AUE_LOGIN  -a  19970401000000 \
-b 19970501000000 \
-O /usr/audit_summary/logins_april97
```

The output is a smaller file containing just the April 1997 login/logout records. Note that the end-time stamp is the date (in GMT) that the command was run (June 1, 1997), not the last date of the merged records. You specified the file suffix, logins_april97, on the command line with the directory name.

```
/usr/audit_summary/19970401000000.19970601000000.logins_april97
```

▼ To Change the praudit Field Separator to a Tab

When the praudit command displays an audit token, it separates the data fields with commas. However, if a field (such as a time stamp) contains a comma, this cannot be distinguished from a field-separating comma.

- **Press the Tab key as the value of the -d option to praudit(1M).**

```
$ praudit -d"<press Tab key>" 19970413120429.19970413180433.grebe
```

There is no space between the -d option and the delimiter. Surround the delimiter with double quotes. The delimiter can be up to four characters long.

▼ To Change the praudit Token Separator to a Tab

Audit tokens are separated by newlines by default. When audit records are printed one per line using the -l option, the audit token separator is the same as the audit field separator. In the following screen example, the audit tokens are separated by tabs, as are the audit fields.

```
$ praudit -l -d"<press Tab key>" 19970413120429.19970413180433.grebe
```

▼ To Perform Selections Using a praudit Script

To accomplish more sophisticated display and reports, process the output from praudit with sed or awk, or write programs to interpret and process the binary audit records.

It is sometimes useful to manipulate `praudit` output as lines of text; for example to perform selections that cannot be done with `auditreduce`. A simple shell script can process the output of `praudit`. The following example is called `praudit_grep`:

```
#!/bin/sh
praudit | sed -e ’1,2d’ -e ’$s/^file.*$//’ -e ’s/^header/^aheader/’ \\
| tr ’\\012\\001’ ’\\002\\012’ \\
| grep "$1" \\
| tr ’\\002’ ’\\012’
```

The example script marks the header tokens by prefixing them with Control-A. (Note that the ^a is Control-A, not the two characters ^ and a. Prefixing is necessary to distinguish them from the string header that might appear as text.) The script then combines all the tokens for a record onto one line while preserving the line breaks as Control-A, runs the `grep` command, and restores the original newlines.

To run the script in the Trusted Solaris environment, the following conditions must be met:

- The script exists in an `admin_low` directory (to make it visible to the Profile Manager).
- The security administrator has added the script to the appropriate profile (such as Custom Admin Role), and given it the forced privileges:
- The security administrator has added any commands in the script that are not in the role’s profile to the appropriate profile.
- The admin role runs the script in an `admin_high` profile shell in a directory where the admin role has write access.

To Back Up Audit Files

1. **As the role oper in an `admin_high` workspace, go to the system’s audit files directory.**

   ```
   $ cd /etc/security/audit/system_name[.n]/files
   ```

2. **Allocate, at the label `admin_high`, the tape drive that you are going to use for backup.**

   If you are unfamiliar with device allocation, see “To Allocate and Deallocate Devices” on page 58.

3. **Use the `tar`(1) command to copy the completed audit files and their Trusted Solaris security attributes, such as the label, to the tape.**

   For example,

   ```
   $ tar cvT \
   /etc/security/audit/grebe/files/19980413120429.19980413180433.grebe \
   ```

```
/etc/security/audit/grebe/files/19980502120429.19980502180433.grebe \
/etc/security/audit/grebe/files/19980513120429.19980513180433.grebe
```

4. **Deallocate the tape drive when finished, remove the tape, and label it `admin_high`.**

5. **At the same time, in an `admin_low` workspace, back up system files that capture information about the users, labels, roles, and execution profiles on the system.**

   Store the audit tapes with the current system information tape(s).

6. **As admin, at label `admin_high`, remove the audit files that have been backed up.**

   For example,

   ```
   $ rm \
   /etc/security/audit/grebe/files/19980413120429.19980413180433.grebe \
   /etc/security/audit/grebe/files/19980502120429.19980502180433.grebe \
   /etc/security/audit/grebe/files/19980513120429.19980513180433.grebe
   ```

▼ To Restore Audit Files

1. **As role admin, in an `admin_high` workspace, go to the directory where the audit files are to be placed.**

   ```
   $ cd /etc/security/audit/system_name[.n]/reports
   ```

2. **Allocate, at the label `admin_high`, the tape drive that you are going to use to restore the files.**

   If you are unfamiliar with device allocation, see "To Allocate and Deallocate Devices" on page 58.

3. **Use the `tar`(1) command to copy the audit files and their Trusted Solaris security attributes, such as the label, from the tape.**

   For example,

   ```
   $ tar xvT \
   /etc/security/audit/grebe/files/19980513120429.19980513180433.grebe
   ```

4. **Deallocate the tape drive when finished and follow the Device Manager's instructions.**

5. **Use the restored audit files.**

   You may need to restore or refer to other system information from the audit backup's associated system backup.

6. **As role admin, at label `admin_high`, remove the audit files when you are done.**

```
$ rm /etc/security/audit/system_name/reports/19980513120429.19980513180433.grebe
```

CHAPTER 4

Troubleshooting Auditing

Another auditing task is to handle audit anomalies as they occur. Typical tasks that audit analysts and system administrators face are discussed below.

- "Preventing Audit Trail Overflow" on page 83
- "Cleaning up an Open Audit File" on page 84
- "Using the sequence Token for Debugging" on page 84
- "To Start the Audit Daemon Manually" on page 87
- "To Prevent Computers From Being Audited Differently" on page 88
- "To Find Failed Login Attempts" on page 89

Preventing Audit Trail Overflow

When all audit file systems for a workstation fill up, the `audit_warn` script sends a message to the console that the hard limit has been exceeded on all audit file systems and also sends mail to the alias. By default, the audit daemon remains in a loop sleeping and checking for space until some space is freed. All auditable actions are suspended. The audit policy `ahlt` is in effect.

Site security policy may permit a different solution. There are other candidates: preventing overflow and keeping a count of dropped audit records.

If your security policy requires that overflow be prevented so that no audit data is ever lost, see "To Prevent Audit Trail Overflow by Planning Ahead" on page 85.

Note – The audit system can be configured to discard audit records upon overflow of the kernel audit buffer. Such a configuration does not constitute an evaluated configuration of the system, and the system should be configured to suspend upon overflow of the audit buffer.

If your security policy permits the loss of some audit data rather than suspending system activities due to audit trail overflow. In that case, you can set the `auditconfig` policy to drop or count records. See "To Handle an Audit Filesystem Overflow" on page 86 for how to drop or count records.

If your security policy requires you to handle filesystem overflow by halting the affected workstation, you must enter the workstation in single-user mode. This is not a secure practice. See "To Handle an Audit Filesystem Overflow" on page 86 for the procedure.

Cleaning up an Open Audit File

Occasionally, if an audit daemon dies while its audit file is still open, or a server becomes inaccessible and forces the workstation to switch to a new server, an audit file remains in which the end-time in the file name remains the string `not_terminated`, even though the file is no longer used for audit records.

The `auditreduce`(1M) command processes files marked `not_terminated`, but because such files may contain incomplete records at the end, future processing may generate errors. To avoid errors, clean the incomplete file with the `-O` option of `auditreduce`. This creates a new file containing all the records that were in the old one, but with a proper file name time stamp. This operation loses the previous file pointer that's kept at the beginning of each audit file.

Using the sequence Token for Debugging

When an audit trail created from merging records from several workstations appears to have the records listed out of order, you can debug the audit trail discrepancies using the sequence token. Since the sequence token is not recorded by default, the security administrator adds it to the audit policy. The audit policy must be set identically on all workstations contributing to the audit trail.

When the audit trail has been debugged, the security administrator removes the token.

Troubleshooting (Tasks)

▼ To Prevent Audit Trail Overflow by Planning Ahead

If your security policy requires that all audit data be saved, do the following:

1. **Set up a schedule to regularly archive audit files and to delete the archived audit files from all audit file systems.**

   The schedule must permit files to be deleted from the system before the hard limit of the system is reached. Scripts, including modified `audit_warn` scripts, can automatically move audit files to a separate disk before archiving.

2. **Manually archive audit files by backing them up on tape or moving them to an archive file system.**

3. **Store context-sensitive information that will be needed to interpret audit records along with the audit trail.**

   For example, the current list of users and passwords, the directory listings on the workstations, and other volatile information should be saved.

4. **Keep records of what audit files are moved off line.**

5. **Store the archived tapes appropriately.**

6. **Reduce the volume of audit data you store by creating summary files.**

   You can extract summary files from the audit trail using options to `auditreduce`, so that the summary files contain only records for certain specified types of audit events. An example of this would be a summary file containing only the audit records for all logins and logouts. See “The Audit Trail” on page 67.

▼ To Handle an Audit Filesystem Overflow

- **To set the audit policy that a count of audit records is kept when the audit file systems are full, as role secadmin, at label `admin_low`:**

```
$ auditconfig -setpolicy +cnt
```

Caution – To run auditing in an evaluated configuration, you cannot have the `+cnt` policy turned on. It *must* be turned off.

- **To set the audit policy that the workstation is shut down when its audit file systems are full:**

```
$ auditconfig -setpolicy +ahlt
```

To set one of the above policies permanently, enter the command in the `audit_startup`(1M) script. See "To Set Audit Policy Permanently" on page 56 for how to edit the script.

Note – On a distributed system, the same audit policy should be applied to all workstations.

▼ To Clean Up an Open Audit File

1. **As role admin, at label `admin_high` check the `/etc/security/audit_data` file to determine the current process number of the audit daemon.**

   If that process is still running, and if the file name in `audit_data`(4) is the same as the file in question, do not clean the file.

2. **Issue the command `auditreduce` with the `-O` (capital o) option.**

3. **Provide the workstation name as the argument to `-O`, and the incomplete file name. To delete the original record, use the `-D` option.**

```
$ auditreduce -O workstation 19970413120429.not_terminated.workstation
```

   This creates a new audit file with the correct name, cleans up pointers to other files, and copies all the records to the new file. The end-time is the time when the command was executed; the correct suffix is *workstation*, explicitly specified.

4. **If you did not use the `-D` option, verify that the new file contains the original file's records, then delete the original file.**

```
$ ls -l 19970413120429*.workstation
$ rm 19970413120429.not_terminated*
```

▼ To Add the sequence Token to the Audit Record

1. **To add the `seq` audit policy dynamically, as role secadmin, at label `admin_low`, on the command line:**

   ```
   $ auditconfig -setpolicy +seq
   $ auditconfig -getpolicy
   slabel, seq
   ```

2. **To add the `seq` audit policy permanently, as role secadmin at label `admin_low`, in the `audit_startup` file:**

   ```
   #!/bin/sh
   auditconfig -setpolicy +slabel,seq
   ```

▼ To Prevent the sequence Token from Being Part of Audit Records

1. **To remove the `seq` audit policy dynamically, on the command line, as role secadmin at label `admin_low`:**

   ```
   $ auditconfig -setpolicy -seq
   $ auditconfig -getpolicy
   slabel
   ```

2. **To remove the `seq` audit policy from the `audit_startup` file, as role secadmin at label `admin_low`:**

   ```
   #!/bin/sh
   auditconfig -setpolicy +slabel
   ```

▼ To Start the Audit Daemon Manually

On a distributed system, if many workstations have lost their audit daemon, bring up the audit daemons in order.

- **As role secadmin, execute the command `/usr/sbin/auditd` in an `admin_high` shell on the audit administration server, then on the audit servers, and finally on the audit clients.**

  ```
  $ /usr/sbin/auditd
  ```

  If you are unfamiliar with creating an `admin_high` shell, see "To Create an Admin_High Workspace" on page 63.

▼ To Prevent Computers From Being Audited Differently

If you change audit configuration files on one workstation and fail to copy the files to the other workstations on the network, the workstations will be audited differently.

1. **As role secadmin, at label `admin_low`, copy the audit configuration files from a central location to every workstation.**

   Follow the procedure in "To Distribute Audit Configuration Files" on page 57.

2. **Check that the audit class mappings for attributable and nonattributable events match the kernel cache.**

   See "To Set Audit Class Mappings for Attributable Events" on page 88 and "To Set Audit Class Mappings for Non-Attributable Audit Events" on page 88 for details.

▼ To Set Audit Class Mappings for Attributable Events

1. **First, as role secadmin at label `admin_low`, check to see if the kernel preselection mask matches the class mappings in the `flags:` field of the `audit_control`(4) file by issuing the command:**

```
$ auditconfig -chkconf
```

2. **If the runtime class mappings differ from the kernel cache, issue the command:**

```
$ auditconfig -conf
```

▼ To Set Audit Class Mappings for Non-Attributable Audit Events

1. **First, as role secadmin at label `admin_low`, check to see if the kernel preselection mask matches the nonattributable events in the `naflags:` field of the `audit_control`(4) file by issuing the command:**

```
$ auditconfig -getkmask
```

2. **If they differ, issue the command:**

```
$ auditconfig -setkmaskac
```

▼ To Find Failed Login Attempts

- **As role admin at label `admin_high`, enter `-lo` as the value of the `-c` option to `auditreduce`(1M).**

```
$ auditreduce -c  -lo   -O  /usr/audit_summary/logins_failed
```

The value "`-lo`" is the audit flag for failed (–) login (audit class `lo`) attempts. The command produces a binary file in the `/usr/audit_summary` directory with all failed login attempts on the distributed system. The `/usr/audit_summary` directory is labeled `admin_high`.

```
/usr/audit_summary/19970313120429.19970613120415.logins_failed
```

Note – This command works only if the security administrator has preselected failed logins for the computer, network, or users.

APPENDIX **A**

Event-to-Class Mappings

This appendix lists audit events by audit class. See the file `/etc/security/audit_event` for a list of events by audit event number.

Audit Events Listed by Audit Class

The Trusted Solaris environment provides the audit classes listed alphabetically by Short Name in the following table. The classes are listed in the file `/etc/security/audit_class`.

TABLE A–1 Trusted Solaris Audit Classes (Default)

Short Name	Long Name	Audit Mask	List of Events per Class
aa	Audit administration	0x00040000	Table A–2
ad	Administrative	0x000f0000	Table A–3
ao	Other administration	0x00080000	Table A–4
all	All classes	0xffffffff	
ap	Application	0x00004000	Table A–6
as	System-wide administration	0x00020000	Table A–24
ax	X server	0x00002000	Table A–26
cl	File close	0x00000040	Table A–7
fa	File attribute access	0x00000004	Table A–8
fc	File create	0x00000010	Table A–9

TABLE A–1 Trusted Solaris Audit Classes (Default) *(Continued)*

Short Name	Long Name	Audit Mask	List of Events per Class
fd	File delete	0x00000020	Table A–10
fm	File attribute modify	0x00000008	Table A–11
fn	Fcntl	0x40000000	Table A–12
fr	File read	0x00000001	Table A–13
fw	File write	0x00000002	Table A–14
il	Internal label info	0x00010000	Obsolete.
io	Ioctl	0x20000000	Table A–15
ip	Ipc	0x00000200	Table A–16
lo	Login or logout	0x00001000	Table A–17
na	Non-attribute	0x00000400	Table A–18
no	Invalid class	0x00000000	Table A–19
nt	Network	0x00000100	Table A–20
ot	Other	0x80000000	Table A–21
pc	Process	0x00300000	No defined events.
pm	Process modify	0x00200000	Table A–22
ps	Process start/stop	0x00100000	Table A–23
ss	Change system state	0x00010000	Table A–25
xa	X - Allowed information flows	0x40000000	Table A–27
xc	X - Object create/destroy	0x20000000	Table A–28
xl	X - Client login/logout	0x08000000	Table A–29
xp	X - Privileged operations	0x10000000	Table A–30
xs	X - Operations that fail silently	0x80000000	Table A–31
xx	X - All X events	0xf8000000	See the individual X classes.

For more information about the classes, see the `audit_class`(4) man page.

Events in Audit Class aa

The following table lists in alphabetical order the `audit administration` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–2 Audit Administration Audit Events (Default)

Audit Event Number and Event	Where Described
224 AUE_AUDITON_GETCAR	Table B–9
231 AUE_AUDITON_GETCLASS	Table B–10
229 AUE_AUDITON_GETCOND	Table B–11
223 AUE_AUDITON_GETCWD	Table B–12
221 AUE_AUDITON_GETKMASK	Table B–13
225 AUE_AUDITON_GETSTAT	Table B–14
141 AUE_AUDITON_GPOLICY	Table B–15
145 AUE_AUDITON_GQCTRL	Table B–16
139 AUE_AUDITON_GTERMID	No longer supported.
144 AUE_AUDITON_SESTATE	No longer supported.
232 AUE_AUDITON_SETCLASS	Table B–17
230 AUE_AUDITON_SETCOND	Table B–18
222 AUE_AUDITON_SETKMASK	Table B–19
228 AUE_AUDITON_SETSMASK	Table B–20
226 AUE_AUDITON_SETSTAT	Table B–21
227 AUE_AUDITON_SETUMASK	Table B–22
142 AUE_AUDITON_SPOLICY	Table B–23
146 AUE_AUDITON_SQCTRL	Table B–24
140 AUE_AUDITON_STERMID	No longer supported.
529 AUE_AUDITPSA	Table B–25
150 AUE_AUDITSTAT	Table B–26
136 AUE_AUDITSVC	Table B–27
530 AUE_FAUDITPSA	Table B–40
132 AUE_GETAUDIT	Table B–51
130 AUE_GETAUID	Table B–53
147 AUE_GETKERNSTATE	No longer supported.
149 AUE_GETPORTAUDIT	Table B–63
134 AUE_GETUSERAUDIT	No longer supported.

TABLE A–2 Audit Administration Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
133 `AUE_SETAUDIT`	Table B–138
131 `AUE_SETAUID`	Table B–140
148 `AUE_SETKERNSTATE`	No longer supported.
135 `AUE_SETUSERAUDIT`	No longer supported.
9016 `AUE_audit`	Table B–239
9015 `AUE_auditwrite`	Table B–240

Events in Audit Class ad

The following table lists in alphabetical order the `administrative` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–3 Administrative Audit Events (Default)

Audit Event Number and Event	Where Described
267 `AUE_GETAUDIT_ADDR`	Table B–52
263 `AUE_PROCESSOR_BIND`	Table B–109
266 `AUE_SETAUDIT_ADDR`	Table B–139
268 `AUE_UMOUNT2`	Table B–179
6166 `AUE_init_solaris`	
6167 `AUE_uadmin_solaris`	
6168 `AUE_shutdown_solaris`	
6169 `AUE_poweroff_solaris`	
6170 `AUE_crontab_mod`	
6207 `AUE_create_user`	
6208 `AUE_modify_user`	
6209 `AUE_delete_user`	
6210 `AUE_disable_user`	
6211 `AUE_enable_user`	
6220 `AUE_smserverd`	

TABLE A–3 Administrative Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
6214 AUE_kadmind_auth	
6215 AUE_kadmind_unauth	

Events in Audit Class ao

The following table lists in alphabetical order the `other administration` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–4 Administrative Other Audit Events (Default)

Audit Event Number and Event	Where Described
61 AUE_EXPORTFS	Table B–297
62 AUE_MOUNT	Table B–82
115 AUE_NFSSVC_EXIT	nfssvc(2)
58 AUE_NFS_GETFH	nfs_getfh(2)
53 AUE_NFS_SVC	nfs_svc(2)
60 AUE_QUOTACTL	Table B–115
12 AUE_UMOUNT	Table B–178
56 AUE_UNMOUNT	No longer supported.
233 AUE_UTSSYS	Table B–182
6200 AUE_allocate_succ	Table B–232
6201 AUE_allocate_fail	Table B–233
6144 AUE_at_create	Table B–236
6145 AUE_at_delete	Table B–237
6146 AUE_at_perm	Table B–238
9034 AUE_automountd_mismatch	Table B–241
9033 AUE_automountd_mount	Table B–242
9029 AUE_chroot_cmd	Table B–243
6147 AUE_cron_invoke	Table B–246
6148 AUE_crontab_create	Table B–244

TABLE A–4 Administrative Other Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
6149 AUE_crontab_delete	Table B–245
6150 AUE_crontab_perm	Table B–248
6202 AUE_deallocate_succ	Table B–250
6203 AUE_deallocate_fail	Table B–251
6181 AUE_filesystem_add	Table B–300
6182 AUE_filesystem_delete	Table B–300
6183 AUE_filesystem_modify	Table B–300
9031 AUE_fuser	Table B–255
6205 AUE_listdevice_succ	Table B–234
6206 AUE_listdevice_fail	Table B–235
9044 AUE_lp_cancel	Table B–268
9045 AUE_lp_status	
6184 AUE_network_add	Table B–299
6185 AUE_network_delete	Table B–299
6186 AUE_network_modify	Table B–299
6187 AUE_printer_add	Table B–278
6188 AUE_printer_delete	Table B–278
6189 AUE_printer_modify	Table B–278
6180 AUE_prof_cmd	Table B–273
6173 AUE_role_login	Table B–291
6190 AUE_scheduledjob_add	Table B–302
6191 AUE_scheduledjob_delete	Table B–302
6192 AUE_scheduledjob_modify	Table B–302
6193 AUE_serialport_add	Table B–301
6194 AUE_serialport_delete	Table B–301
6195 AUE_serialport_modify	Table B–301
9013 AUE_sendmail_deliver	Table B–293
9014 AUE_sendmail_defer	Table B–293

TABLE A–4 Administrative Other Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
9012 AUE_sendmail_upgrade	Table B–294
9322 AUE_te_modsysfiles	Table B–231
6199 AUE_uauth	Table B–308
9024 AUE_uname_set	Table B–311
6196 AUE_usermgr_add	Table B–303
6197 AUE_usermgr_delete	Table B–303
6198 AUE_usermgr_modify	Table B–303

TABLE A–5 Administrative Other Audit Events (Obsolete)

Audit Event Number and Event	Where Described
9319 AUE_dm_add	Table B–249
9320 AUE_dm_del	
9321 AUE_dm_mod	
9307 AUE_gm_add_grp	Table B–256
9308 AUE_gm_del_grp	
9309 AUE_gm_mod_grp	
9310 AUE_hm_add_host	Table B–258
9311 AUE_hm_del_host	
9312 AUE_hm_mod_host	
9313 AUE_hm_set_def	
9009 AUE_pfsh_priv	Table B–275
9008 AUE_pfsh_trusted_nopriv	
9007 AUE_pfsh_trusted_priv	
9316 AUE_pm_add	Table B–279
9318 AUE_pm_del_prn	
9317 AUE_pm_mod_prn	
9306 AUE_pm_add_prof	Table B–280
9306 AUE_pm_del_prof	Table B–281
9305 AUE_pm_mod_prof	Table B–282

TABLE A–5 Administrative Other Audit Events (Obsolete) *(Continued)*

Audit Event Number and Event	Where Described
9315 `AUE_sm_del_ser`	Table B–295
9314 `AUE_sm_mod_ser`	
9302 `AUE_um_add_user`	Table B–310
9301 `AUE_um_del_user`	
9300 `AUE_um_mod_user`	
9303 `AUE_um_set_def`	

Events in Audit Class ap

The following table lists in alphabetical order the `application` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–6 Application Audit Events (Default)

Audit Event Number and Event	Where Described
9010 `AUE_pfsh_nopriv`	Table B–275
9035 `AUE_sl_change`	Table B–304

Events in Audit Class cl

The following table lists in alphabetical order the `file close` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–7 File Close Audit Events (Default)

Audit Event Number and Event	Where Described
112 `AUE_CLOSE`	Table B–34
213 `AUE_MUNMAP`	Table B–91

Events in Audit Class fa

The following table lists in alphabetical order the `file attribute access` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–8 File Attribute Access Audit Events (Default)

Audit Event Number and Event	Where Described
14 AUE_ACCESS	Table B–5
220 AUE_AUDITSYS	Placeholder
66 AUE_BSMSYS	Placeholder
543 AUE_FGETCMWLABEL	Table B–55
55 AUE_FSTATFS	Table B–50
545 AUE_GETCMWFSRANGE	Table B–54
546 AUE_GETCMWLABEL	Table B–55
547 AUE_GETFILEPRIV	Table B–57
554 AUE_GETMLDADORN	Table B–58
555 AUE_GETSLDNAME	Table B–66
548 AUE_LGETCMWLABEL	Table B–55
17 AUE_LSTAT	Table B–71
236 AUE_LXSTAT	Table B–72
556 AUE_MLDLSTAT	Obsolete.
557 AUE_MLDSTAT	
64 AUE_MSGSYS	Placeholder
3 AUE_OPEN	Placeholder
199 AUE_OSTAT	No longer supported.
71 AUE_PATHCONF	Table B–105
67 AUE_RFSSYS	Placeholder
63 AUE_SEMSYS	Placeholder
65 AUE_SHMSYS	Placeholder
16 AUE_STAT	Table B–167
54 AUE_STATFS	
234 AUE_STATVFS	
70 AUE_VPIXSYS	Placeholder
235 AUE_XSTAT	Table B–188

Events in Audit Class fc

The following table lists in alphabetical order the `file create` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–9 File Create Audit Events (Default)

Audit Event Number and Event	Where Described
111 AUE_CORE	Table B–111
4 AUE_CREAT	Table B–35
532 AUE_FGETSLDNAME	Table B–46
5 AUE_LINK	Table B–70
47 AUE_MKDIR	Table B–74
9 AUE_MKNOD	Table B–75
73 AUE_OPEN_RC	Table B–94
75 AUE_OPEN_RTC	Table B–95
81 AUE_OPEN_RWC	Table B–98
83 AUE_OPEN_RWTC	Table B–99
77 AUE_OPEN_WC	Table B–102
79 AUE_OPEN_WTC	Table B–103
42 AUE_RENAME	Table B–119
48 AUE_RMDIR	Table B–120
21 AUE_SYMLINK	Table B–169
240 AUE_XMKNOD	Table B–187

Events in Audit Class fd

The following table lists in alphabetical order the `file delete` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–10 File Delete Audit Events (Default)

Audit Event Number and Event	Where Described
44 AUE_FTRUNCATE	No longer supported.
74 AUE_OPEN_RT	Table B–96

TABLE A–10 File Delete Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
75 `AUE_OPEN_RTC`	Table B–95
82 `AUE_OPEN_RWT`	Table B–100
83 `AUE_OPEN_RWTC`	Table B–99
78 `AUE_OPEN_WT`	Table B–104
79 `AUE_OPEN_WTC`	Table B–103
42 `AUE_RENAME`	Table B–119
6 `AUE_UNLINK`	Table B–180

Events in Audit Class fm

The following table lists in alphabetical order the `file attribute modify` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–11 File Attribute Modify Audit Events (Default)

Audit Event Number and Event	Where Described
251 `AUE_ACLSET`	Table B–137
11 `AUE_CHOWN`	Table B–30
252 `AUE_FACLSET`	Table B–137
39 `AUE_FCHMOD`	Table B–42
38 `AUE_FCHOWN`	Table B–43
45 `AUE_FLOCK`	Placeholder
544 `AUE_FSETCMWLABEL`	Table B–142
523 `AUE_FSETFATTRFLAG`	Table B–49
158 `AUE_IOCTL`	Table B–67
237 `AUE_LCHOWN`	Table B–69
525 `AUE_LSETCMWLABEL`	Table B–142
19 `AUE_MCTL`	No longer supported.
524 `AUE_MLDSETFATTRFLAG`	Table B–76
542 `AUE_SETCLEARANCE`	Table B–141

TABLE A–11 File Attribute Modify Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
549 `AUE_SETCMWLABEL`	Table B–142
541 `AUE_SETCMWPLABEL`	Table B–143
522 `AUE_SETFATTRFLAG`	Table B–146
550 `AUE_SETFILEPRIV`	Table B–147
551 `AUE_SETPROCPRIV`	Table B–151
202 `AUE_UTIME`	Table B–181
49 `AUE_UTIMES`	
552 `AUE_WRITEL`	Table B–186
553 `AUE_WRITEVL`	
9037 `AUE_dtfile_copy`	Table B–253
9038 `AUE_dtfile_move`	Table B–253

Events in Audit Class fn

The following table lists in alphabetical order the `fcntl` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–12 Fcntl Audit Events (Default)

Audit Event Number and Event	Where Described
30 `AUE_FCNTL`	Table B–45

Events in Audit Class fr

The following table lists in alphabetical order the `file read` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–13 File Read Audit Events (Default)

Audit Event Number and Event	Where Described
72 `AUE_OPEN_R`	Table B–93
73 `AUE_OPEN_RC`	Table B–94

TABLE A–13 File Read Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
74 AUE_OPEN_RT	Table B–96
75 AUE_OPEN_RTC	Table B–95
80 AUE_OPEN_RW	Table B–97
81 AUE_OPEN_RWC	Table B–98
82 AUE_OPEN_RWT	Table B–100
83 AUE_OPEN_RWTC	Table B–99
22 AUE_READLINK	Table B–117

Events in Audit Class fw

The following table lists in alphabetical order the `file read` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–14 File Write Audit Events (Default)

Audit Event Number and Event	Where Described
80 AUE_OPEN_RW	Table B–97
81 AUE_OPEN_RWC	Table B–98
82 AUE_OPEN_RWT	Table B–100
83 AUE_OPEN_RWTC	Table B–99
76 AUE_OPEN_W	Table B–101
77 AUE_OPEN_WC	Table B–102
78 AUE_OPEN_WT	Table B–104
79 AUE_OPEN_WTC	Table B–103

Events in Audit Class io

The following table lists the audit event in the `ioctl` class provided in the Trusted Solaris 8 4/01 release.

TABLE A–15 Ioctl Audit Events (Default)

Audit Event Number and Event	Where Described
158 AUE_IOCTL	Table B–67

Events in Audit Class ip

The following table lists in alphabetical order the ipc class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–16 IPC Audit Events (Default)

Audit Event Number and Event	Where Described
260 AUE_DOORFS_DOOR_BIND	doorfs(2) - DOOR_BIND
254 AUE_DOORFS_DOOR_CALL	doorfs(2) - DOOR_CALL
256 AUE_DOORFS_DOOR_CREATE	doorfs(2) - DOOR_CREATE
258 AUE_DOORFS_DOOR_INFO	doorfs(2) - DOOR_INFO
255 AUE_DOORFS_DOOR_RETURN	doorfs(2) - DOOR_RETURN
257 AUE_DOORFS_DOOR_REVOKE	doorfs(2) - DOOR_REVOKE
259 AUE_DOORFS_DOOR_CRED	doorfs(2) - DOOR_CRED
261 AUE_DOORFS_DOOR_UNBIND	doorfs(2) - DOOR_UNBIND
514 AUE_GETMSGQCMWLABEL	Table B–61
515 AUE_GETSEMCMWLABEL	Table B–64
516 AUE_GETSHMCMWLABEL	Table B–65
84 AUE_MSGCTL	Illegal command
85 AUE_MSGCTL_RMID	Table B–84
86 AUE_MSGCTL_SET	Table B–85
87 AUE_MSGCTL_STAT	Table B–86
88 AUE_MSGGET	Table B–87
174 AUE_MSGGETL	Table B–88
89 AUE_MSGRCV	Table B–89
175 AUE_MSGRCVL	Table B–89
90 AUE_MSGSND	Table B–90

TABLE A–16 IPC Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
176 AUE_MSGSNDL	Obsolete.
98 AUE_SEMCTL	Illegal command
105 AUE_SEMCTL_GETALL	Table B–122
102 AUE_SEMCTL_GETNCNT	Table B–123
103 AUE_SEMCTL_GETPID	Table B–124
104 AUE_SEMCTL_GETVAL	Table B–125
106 AUE_SEMCTL_GETZCNT	Table B–126
99 AUE_SEMCTL_RMID	Table B–127
100 AUE_SEMCTL_SET	Table B–128
108 AUE_SEMCTL_SETALL	Table B–129
107 AUE_SEMCTL_SETVAL	Table B–130
101 AUE_SEMCTL_STAT	Table B–131
109 AUE_SEMGET	Table B–132
177 AUE_SEMGETL	Table B–133
110 AUE_SEMOP	Table B–134
517 AUE_SEMOPL	Obsolete.
96 AUE_SHMAT	Table B–157
91 AUE_SHMCTL	Placeholder
92 AUE_SHMCTL_RMID	Table B–159
93 AUE_SHMCTL_SET	Table B–160
94 AUE_SHMCTL_STAT	Table B–161
97 AUE_SHMDT	Table B–162
95 AUE_SHMGET	Table B–163
178 AUE_SHMGETL	Table B–164

Events in Audit Class lo

The following table lists in alphabetical order the `login` or `logout` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–17 Login or Logout Audit Events (Default)

Audit Event Number and Event	Where Described
6123 `AUE_admin_authenticate`	Table B–298
6165 `AUE_ftpd`	Table B–260
6152 `AUE_login`	Table B–262
6153 `AUE_logout`	Table B–265
6163 `AUE_passwd`	Table B–272
6164 `AUE_rexd`	Table B–286
6162 `AUE_rexecd`	Table B–287
6155 `AUE_rlogin`	Table B–263
6173 `AUE_role_login`	Table B–291
6158 `AUE_rshd`	Table B–288
6159 `AUE_su`	Table B–305
6154 `AUE_telnet`	Table B–264

Events in Audit Class na

The following table lists in alphabetical order the `non-attribute` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–18 Non-attribute Audit Events (Default)

Audit Event Number and Event	Where Described
153 `AUE_ENTERPROM`	Table B–37
154 `AUE_EXITPROM`	
113 `AUE_SYSTEMBOOT`	Table B–171
6151 `AUE_inetd_connect`	Table B–259
6156 `AUE_mountd_mount`	Table B–270
6157 `AUE_mountd_umount`	Table B–271

Events in Audit Class no

The following table lists in alphabetical order the `invalid class` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–19 Invalid Class Audit Events (Default)

Audit Event Number and Event	Where Described
211 AUE_AUDIT	Table B–8
209 AUE_DUP2	No longer supported.
208 AUE_FSTAT	Table B–50
193 AUE_GETDENTS	Table B–56
13 AUE_JUNK	
194 AUE_LSEEK	Placeholder
518 AUE_MAC	No longer supported.
210 AUE_MMAP	Table B–77
242 AUE_MODCTL	Placeholder
197 AUE_NFS	Placeholder
0 AUE_NULL	Indirect system call
185 AUE_PIPE	Table B–106
527 AUE_PREADL	Table B–107
528 AUE_PWRITEL	Table B–186
192 AUE_READ	Table B–116
558 AUE_READL	
198 AUE_READV	Placeholder
559 AUE_READVL	Table B–116
189 AUE_RECV	Placeholder
187 AUE_SEND	Placeholder
186 AUE_SOCKETPAIR	Placeholder
521 AUE_UPRIV	Table B–189
195 AUE_WRITE	Table B–185
196 AUE_WRITEV	Placeholder

Events in Audit Class nt

The following table lists in alphabetical order the `network` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–20 Network Audit Events (Default)

Audit Event Number and Event	Where Described
33 AUE_ACCEPT	No longer supported.
34 AUE_BIND	No longer supported.
32 AUE_CONNECT	No longer supported.
217 AUE_GETMSG	Table B–59
219 AUE_GETPMSG	Table B–62
173 AUE_ONESIDE	No longer supported.
216 AUE_PUTMSG	Table B–112
218 AUE_PUTPMSG	Table B–114
191 AUE_RECVFROM	Format unavailable.
190 AUE_RECVMSG	Table B–118
188 AUE_SENDMSG	Table B–135
184 AUE_SENDTO	Table B–136
35 AUE_SETSOCKOPT	Table B–155
247 AUE_SOCKACCEPT	Table B–60
248 AUE_SOCKCONNECT	Table B–113
183 AUE_SOCKET	Table B–166
250 AUE_SOCKRECEIVE	Table B–60
249 AUE_SOCKSEND	Table B–113
534 AUE_TNIF	Table B–172
535 AUE_TNRH	
536 AUE_TNRHTP	
537 AUE_TOKMAPPER	Table B–173

Events in Audit Class ot

The following table lists in alphabetical order the other class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–21 Other Audit Events (Default)

Audit Event Number and Event	Where Described
238 AUE_MEMCNTL	Table B–73

Events in Audit Class pm

The following table lists in alphabetical order the process modify class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–22 Process Modify Audit Events (Default)

Audit Event Number and Event	Where Described
8 AUE_CHDIR	Table B–28
24 AUE_CHROOT	Table B–31
1 AUE_EXIT	Table B–39
68 AUE_FCHDIR	Table B–41
69 AUE_FCHROOT	Table B–44
15 AUE_KILL	Table B–68
52 AUE_KILLPG	No longer supported.
203 AUE_NICE	Table B–92
204 AUE_OSETPGRP	No information.
200 AUE_OSETUID	No longer supported.
212 AUE_PRIOCNTLSYS	Table B–108
214 AUE_SETEGID	Table B–144
215 AUE_SETEUID	Table B–145
526 AUE_SETPATTR	Table B–149
205 AUE_SETGID	Table B–144
26 AUE_SETGROUPS	Table B–148
27 AUE_SETPGRP	Table B–150

TABLE A–22 Process Modify Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
31 AUE_SETPRIORITY	No longer supported.
41 AUE_SETREGID	Table B–152
40 AUE_SETREUID	Table B–153
200 AUE_SETUID - event name is AUE_OSETUID	Table B–156
36 AUE_VTRACE	Table B–184

Events in Audit Class ps

The following table lists in alphabetical order the `process start/stop` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–23 Process Start/Stop Audit Events (Default)

Audit Event Number and Event	Where Described
7 AUE_EXEC	Table B–38
23 AUE_EXECVE	
2 AUE_FORK	Table B–47
241 AUE_FORK1	
526 AUE_SETPATTR	Table B–149
25 AUE_VFORK	Table B–183
9027 AUE_psradm	Table B–283

Events in Audit Class as

The following table lists in alphabetical order the `system-wide administration` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–24 System-wide Administration Audit Events (Default)

Audit Event Number and Event	Where Described
18 AUE_ACCT	Table B–6
50 AUE_ADJTIME	Table B–7

TABLE A–24 System-wide Administration Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
57 `AUE_ASYNC_DAEMON`	
114 `AUE_ASYNC_DAEMON_EXIT`	
538 `AUE_CHSTATE`	Table B–32
513 `AUE_CLOCK_SETTIME`	Table B–33
531 `AUE_DRVPOLICY`	Table B–36
264 `AUE_INST_SYNC`	
246 `AUE_MODADDMAJ`	Table B–78
245 `AUE_MODCONFIG`	Table B–79
243 `AUE_MODLOAD`	Table B–80
244 `AUE_MODUNLOAD`	Table B–81
533 `AUE_PRIVENABLE`	Table B–110
540 `AUE_REMOUNT`	Table B–176
59 `AUE_SETDOMAINNAME`	
29 `AUE_SETHOSTNAME`	
51 `AUE_SETRLIMIT`	Table B–154
37 `AUE_SETTIMEOFDAY`	
201 `AUE_STIME`	Table B–168
28 `AUE_SWAPON`	swapon(2)
239 `AUE_SYSINFO`	Table B–170
9018 `AUE_add_drv`	Table B–230
9025 `AUE_dispadmin`	Table B–252
9032 `AUE_eeprom`	Table B–254
9042 `AUE_installf`	Table B–261
9020 `AUE_modload`	Table B–269
9021 `AUE_modunload`	Table B–269
9026 `AUE_pbind`	Table B–274
9040 `AUE_pkginstall`	Table B–276
9041 `AUE_pkgremove`	Table B–277

TABLE A–24 System-wide Administration Audit Events (Default) *(Continued)*

Audit Event Number and Event	Where Described
9027 AUE_psradm	Table B–283
9019 AUE_rem_drv	Table B–289
9043 AUE_removef	Table B–285
9022 AUE_setuname	Table B–296
9030 AUE_swap	Table B–306
9024 AUE_uname_set	Table B–311

Events in Audit Class ss

The following table lists in alphabetical order the `change system state` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–25 Change System State Audit Events (Default)

Audit Event Number and Event	Where Described
539 AUE_FREEZE	Table B–174
561 AUE_REBOOT	Table B–175
560 AUE_SHUTDOWN	Table B–177
6160 AUE_halt_solaris	Table B–257
6161 AUE_reboot_solaris	Table B–284
9028 AUE_run_level_change	Table B–290
9023 AUE_uadmin_cmd	Table B–307

Events in Audit Class ax

The following table lists in alphabetical order the `ax` class of audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–26 X Server Audit Events - Remainder (Default)

Audit Event Number and Event	Where Described
9039 AUE_sel_mgr_xfer	Table B–292

Events in Audit Class xa

The following table lists in alphabetical order the xa class of audit events provided in the Trusted Solaris 8 4/01 release. This class contains X protocols that use "default" client privileges to succeed. These privileges are listed in the file /usr/openwin/server/tsol/config.privs. The security administrator can remove privileges from this file.

TABLE A–27 X - Allowed Information Flows Audit Events (Default)

Audit Event Number and Event		Where Described
9194	AUE_ChangeHosts	Table B–226
9137	AUE_GrabServer	Table B–201
9183	AUE_InstallColormap	Table B–217
9146	AUE_SetFontPath	Table B–207
9138	AUE_UngrabServer	Table B–201

Events in Audit Class xc

The following table lists lists in alphabetical order the xc class of audit events provided in the Trusted Solaris 8 4/01 release. This class contains audit events about the creation and destruction of X server object.

TABLE A–28 X - Object Create/Destroy Operations Audit Events (Default)

Audit Event Number and Event		Where Described
9176	AUE_AllocColor	Table B–216
9178	AUE_AllocColorCells	
9179	AUE_AllocColorPlanes	
9177	AUE_AllocNamedColor	
9120	AUE_ChangeProperty	Table B–193
9170	AUE_CreateColormap	Table B–215
9185	AUE_CreateCursor	Table B–219
9186	AUE_CreateGlyphCursor	Table B–220
9103	AUE_CreateWindow	Table B–192
9121	AUE_DeleteProperty	Table B–193

TABLE A–28 X - Object Create/Destroy Operations Audit Events (Default) *(Continued)*

Audit Event Number and Event		Where Described
9107	AUE_DestroySubwindows	Table B–192
9106	AUE_DestroyWindow	
9171	AUE_FreeColormap	Table B–217
9180	AUE_FreeColors	Table B–216
9187	AUE_FreeCursor	Table B–221
9152	AUE_FreeGC	Table B–210
9147	AUE_FreePixmap	Table B–222
9197	AUE_KillClient	Table B–226

Events in Audit Class xl

The following table lists in alphabetical order the xl audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–29 X - Client Login/Logout Audit Events (Default)

Audit Event Number and Event		Where Described
9101	AUE_ClientConnect	Table B–190
9102	AUE_ClientDisConnect	Table B–191

Events in Audit Class xp

The following table lists in alphabetical order the xp audit events provided in the Trusted Solaris 8 4/01 release.

TABLE A–30 X - Privileged Audit Events (Default)

Audit Event Number and Event		Where Described
9148	AUE_ChangeGc	Table B–208
9120	AUE_ChangeProperty	Table B–193
9108	AUE_ChangeSaveSet	Table B–192
9104	AUE_ChangeWindowAttributes	

TABLE A–30 X - Privileged Audit Events (Default) *(Continued)*

Audit Event Number and Event		Where Described
9115	AUE_CirculateWindow	
9114	AUE_ConfigureWindow	
9172	AUE_CopyColormapAndFree	Table B–217
9149	AUE_CopyGC	Table B–209
9161	AUE_FillPolygon	Table B–213
9199	AUE_ForceScreenSaver	Table B–224
9116	AUE_GetGeometry	Table B–192
9140	AUE_GetMotionEvents	Table B–203
9122	AUE_GetProperty	Table B–193
9105	AUE_GetWindowAttributes	Table B–192
9130	AUE_GrabButton	Table B–196
9135	AUE_GrabKey	Table B–199
9133	AUE_GrabKeyboard	Table B–200
9128	AUE_GrabPointer	Table B–197
9168	AUE_ImageText8	Table B–214
9169	AUE_ImageText16	
9173	AUE_InstallColormap	Table B–217
9123	AUE_ListProperties	Table B–193
9184	AUE_LookupColor	Table B–218
9111	AUE_MapSubwindows	Table B–192
9110	AUE_MapWindow	
9160	AUE_PolyArc	Table B–213
9163	AUE_PolyFillArc	
9162	AUE_PolyFillRectangle	
9157	AUE_PolyLine	
9156	AUE_PolyPoint	
9158	AUE_PolySegment	

TABLE A–30 X - Privileged Audit Events (Default) *(Continued)*

Audit Event Number and Event		Where Described
9166	AUE_PolyText8	Table B–214
9167	AUE_PolyText16	
9164	AUE_PutImage	
9183	AUE_QueryColors	Table B–218
9145	AUE_QueryKeymap	Table B–206
9139	AUE_QueryPointer	Table B–202
9117	AUE_QueryTree	Table B–192
9188	AUE_RecolorCursor	Table B–221
9109	AUE_ReparentWindow	Table B–192
9198	AUE_RotateProperties	Table B–227
9195	AUE_SetAccessControl	Table B–226
9151	AUE_SetClipRectangles	Table B–210
9150	AUE_SetDashes	
9193	AUE_SetScreenSaver	Table B–224
9124	AUE_SetSelectionOwner	Table B–195
9181	AUE_StoreColors	Table B–218
9182	AUE_StoreNamedColor	
9141	AUE_TranslateCoords	Table B–204
9136	AUE_UngrabKey	Table B–200
9174	AUE_UninstallColormap	Table B–217
9113	AUE_UnmapSubwindows	Table B–192
9112	AUE_UnmapWindow	
9202	AUE_XExtensions	Table B–229

Events in Audit Class xs

The following table lists in alphabetical order the xs audit events provided in the Trusted Solaris 8 4/01 release.

Note – These events should be audited for success only, not for failure.

TABLE A–31 X - Fail Silently Audit Events (Default)

Audit Event Number and Event		Where Described
9193	`AUE_Bell`	Table B–223
9132	`AUE_ChangeActivePointerGrab`	Table B–198
9190	`AUE_ChangeKeyboardControl`	Table B–223
9189	`AUE_ChangeKeyboardMapping`	
9192	`AUE_ChangePointerControl`	
9126	`AUE_ConvertSelection`	Table B–195
9154	`AUE_CopyArea`	Table B–212
9155	`AUE_CopyPlane`	
9119	`AUE_GetAtomName`	Table B–194
9165	`AUE_GetImage`	Table B–214
9144	`AUE_GetInputFocus`	Table B–205
9125	`AUE_GetSelectionOwner`	Table B–195
9128	`AUE_GrabPointer`	Table B–197
9118	`AUE_InternAtom`	Table B–194
9175	`AUE_ListInstalledColormap`	Table B–217
9159	`AUE_PolyRectangle`	Table B–213
9127	`AUE_SendEvent`	Table B–203
9196	`AUE_SetCloseDownMode`	Table B–225
9143	`AUE_SetInputFocus`	Table B–205
9201	`AUE_SetModifierMapping`	Table B–228
9200	`AUE_SetPointerMapping`	
9124	`AUE_SetSelectionOwner`	Table B–195
9134	`AUE_UngrabKeyboard`	Table B–199
9129	`AUE_UngrabPointer`	Table B–197
9131	`AUE_UngrabButton`	
9142	`AUE_WarpPointer`	Table B–204

APPENDIX **B**

Audit Record Descriptions

This appendix has two parts. The first part describes each part of an audit record structure and each audit token structure. The second part defines all of the audit records generated in Trusted Solaris 8 4/01 software by event description.

- "Audit Record Structure" on page 119
- "Audit Token Structure" on page 120
- "Kernel-Level Generated Audit Records" on page 150
- "Kernel-Level Pseudo-Events" on page 217
- "X Server Protocol Audit Records" on page 217
- "User-Level Generated Audit Records" on page 233

Audit Record Structure

An audit record is a sequence of audit tokens. Each token contains event information such as user ID, time, and date. A header token begins an audit record, and an optional trailer concludes the record. Other audit tokens contain audit-relevant information. The following figure shows a typical audit record.

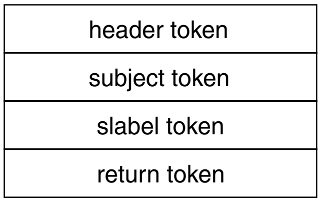


FIGURE B–1 Typical Audit Record

Audit Token Structure

Logically, each token has a token type identifier followed by data specific to the token. Each token type has its own format and structure. The audit tokens are shown in the table below. Those marked TS in the TS8 column are in Trusted Solaris 2.5.1 and later versions only. Those not marked TS are modified versions of audit tokens from the Solaris Basic Security Module. The token scheme can be extended.

TABLE B–1 Trusted Solaris Audit Tokens

Token Name	Description	TS8
"acl Token" on page 122	Access Control List	TS
"arbitrary Token" on page 123	Data with format and type	
"arg Token" on page 124	System call argument value	
"attr Token" on page 124	File attributes	
"clearance Token" on page 125	Clearance	TS
"cmd Token" on page 126	Command execution	
"exec_args Token" on page 126	Exec system call arguments	
"exec_env Token" on page 127	Exec system call environment variables	
"exit Token" on page 127	Program exit	
"file Token" on page 128	Audit file delimiter	

TABLE B–1 Trusted Solaris Audit Tokens *(Continued)*

Token Name	Description	TS8
“groups Token (Obsolete)” on page 128	Process supplementary group (obsolete)	
“header Token” on page 129	Start of audit record	
“host Token” on page 130	Host where audit record was collected	TS
“in_addr Token” on page 131	Internet address	
“ip Token” on page 131	IP header information	
“ipc Token” on page 132	System V IPC information	
“ipc_perm Token” on page 133	System V IPC object tokens	
“iport Token” on page 134	Internet port address	
“liaison Token” on page 134	Liaison information for Trusted Networking	TS
“newgroups Token” on page 135	Process supplementary group information	
“opaque Token” on page 136	Unstructured data (unspecified format)	
“path Token” on page 136	Path (path)	
“upriv Token” on page 144	Use of privilege	TS
“privilege Token” on page 137	Privilege set	TS
“process Token” on page 138	Process information	
“return Token” on page 139	Status of system call	
“return Token” on page 139	Sequence number	
“slabel Token” on page 140	Sensitivity label	TS
“socket Token” on page 141	Socket type and addresses	
“subject Token” on page 142	Subject	
“text Token” on page 143	Character string	
“trailer Token” on page 143	End of audit record	
“uauth Token” on page 144	Use of authorization	
“xatom Token” on page 145	X window atom identification	TS
“xclient Token” on page 145	X client identification	TS
“xcolormap Token” on page 146	X window color information	TS
“xcursor Token” on page 146	X window cursor information	TS

TABLE B–1 Trusted Solaris Audit Tokens *(Continued)*

Token Name	Description	TS8
"xfont Token" on page 147	X window font information	TS
"xgc Token" on page 147	X window graphical context information	TS
"xpixmap Token" on page 147	Xwindow pixel mapping information	TS
"xproperty Token" on page 148	X window property information	TS
"xselect Token" on page 148	X window data information	TS
"xwindow Token" on page 149	X window window information	TS

An audit record always contains a `header` token and may contain a `trailer` token. The `header` token indicates where the audit record begins in the audit trail. The optional `trailer` token allows backward seeks of the audit trail. Every audit record contains a `subject` token, except for audit records from some non-attributable events. In the case of attributable events, these two tokens refer to the values of the process that caused the event. In the case of asynchronous events, the `process` tokens refer to the system. For an example of how to read an audit record, go to "Reading an Audit Record" on page 70.

acl Token

The `acl` token records information about ACLs. It consists of four fixed fields: a token ID that identifies this token as an `acl` token, a field that specifies the ACL type, an ACL ID field, and a field that lists the permissions associated with this ACL. The `acl` token appears as follows:

The following figure shows the token format.

token ID	ACL type	ACL ID	ACL permissions
1 byte	4 bytes	4 bytes	4 bytes

FIGURE B–2 acl Token Format

A list of `acl` tokens is displayed by `praudit`(1M) as follows:

```
acl,user_obj,,rwx
acl,user,bin,---
acl,group_obj,,r-x
acl,class_obj,,r--
acl,other_obj,,r-x
```

arbitrary Token

The arbitrary token encapsulates data for the audit trail. It consists of four fixed fields and an array of data. The item array may have a number of items. The fields are:

- A token ID
- A suggested format, such as decimal
- A size of encapsulated data, such as int
- A count of the data array items
- An item array

The following figure shows the token format.

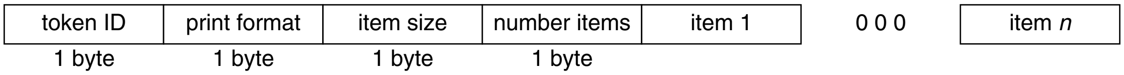


FIGURE B–3 arbitrary Token Format

The print format field can take the values shown in Table B–2.

TABLE B–2 arbitrary Token Print Format Field Values

Value	Action
`AUP_BINARY`	Print date in binary
`AUP_OCTAL`	Print date in octal
`AUP_DECIMAL`	Print date in decimal
`AUP_HEX`	Print date in hex
`AUP_STRING`	Print date as a string

The item size field can take the values shown in Table B–3.

TABLE B–3 arbitrary Token Item Size Field Values

Value	Action
`AUR_BYTE`	Data is in units of bytes (1 byte)
`AUR_SHORT`	Data is in units of shorts (2 bytes)
`AUR_LONG`	Data is in units of longs (4 bytes)
`AUR_LONGLONG`	Data is in units of longlongs (8 bytes)

An arbitrary token is displayed by praudit as follows:

```
arbitrary,decimal,int,1
42
```

arg Token

The arg token contains system call argument information. A 32-bit integer system call argument is allowed in an audit record. The fields are:

- A token ID
- An argument ID of the relevant system call argument
- The argument value
- The length of an optional descriptive text string (does not show)
- An optional text string

The following figure shows the token format.

token ID	argument #	argument value	text value	text
1 byte	1 byte	4 bytes	2 bytes	*n* bytes

FIGURE B–4 arg Token Format

An arg token is displayed by praudit as follows:

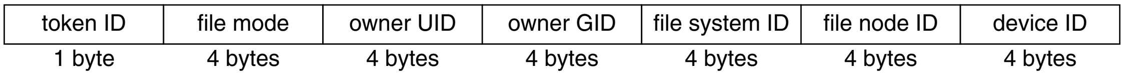
```
argument,2,0x3,cmd
```

attr Token

The attribute token contains file attribute information from the kernel's internal representation of a file or folder. This token usually accompanies a path token and is produced during path searches. In the event of a path-search error, this token is not included as part of the audit record since the file attribute information is not available. The fields are:

- A token ID
- The file access mode and type
- The owner user ID
- The owner group ID
- The file system ID
- The inode ID
- The device ID that the file might represent

See the statvfs(2) man page for further information about the file system ID and the device ID. The following figure shows the token format.



token ID	file mode	owner UID	owner GID	file system ID	file node ID	device ID
1 byte	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes

FIGURE B–5 attr Token Format

An attr token is displayed by praudit as follows:

attribute,100555,root,root,1805,13871,-4288

clearance Token

The clearance token contains Trusted Solaris clearance information. The fields are:

- A token ID
- The CMW clearance, containing
  - A pad ID identifying the label type
  - The clearance's classifications
  - The clearance's compartments

The following figure shows the token format.

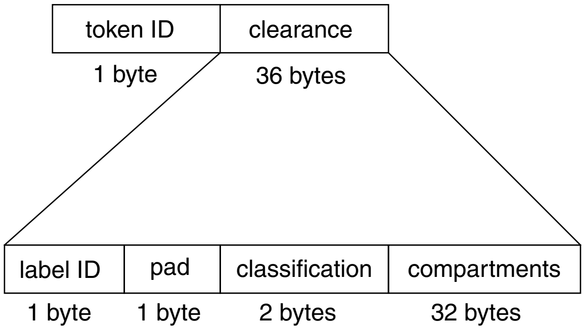


FIGURE B–6 clearance Token Format

A clearance token is displayed by praudit as follows:

clearance,TOP SECRET

cmd Token

The `cmd` token records the arguments and environment in which a command executes. The fields are:

- A token ID
- The number of arguments to the command
- The argument values
- The number of environment variables
- Zero or more names of the variables

A `cmd` token is displayed by `praudit` as follows:

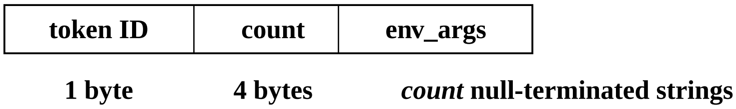
cmd,2,/export/share/tsol8,label_encodings

exec_args Token

The `exec_args` token records the arguments to an `exec()` system call. The fields are:

- A token ID
- A count that represents the number of arguments passed to the `exec` call
- Zero or more null-terminated strings, the arguments of the `exec` call

The following figure shows an `exec_args` token.



token ID	**count**	**env_args**
1 byte	**4 bytes**	***count* null-terminated strings**

FIGURE B–7 exec_args Token Format

Note – The `exec_args` token is output only when the audit policy `argv` is active. See "Dynamic Auditing (Tasks)" on page 62 for more information.

An `exec_args` token is displayed by `praudit` as follows:

exec_args,

exec_env Token

The exec_env token records the current environment variables to an exec() system call. The fields are:

- A token ID
- A count of the current environment variables in the exec call
- Zero or more null-terminated strings, the variables of the exec call

The following figure shows an exec_env token.

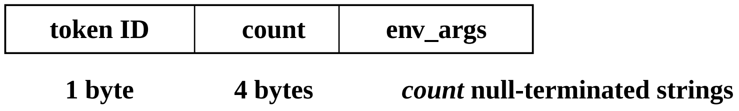


FIGURE B–8 exec_env Token Format

Note – The exec_env token is output only when the audit policy arge is active. See "Dynamic Auditing (Tasks)" on page 62 for more information.

An exec_env token is displayed by praudit as follows:

exec_env,

exit Token

The exit token records the exit status of a program and a return value. The fields are:

- A token ID
- A program exit status as passed to the exit() system call
- A return value that describes the exit status or indicates a system error number

The following figure shows an exit token.

token ID	status	return value
1 byte	4 bytes	4 bytes

FIGURE B–9 exit Token Format

An exit token is displayed by praudit as follows:

exit,Error 0,0

file Token

The file token is a special token generated by the audit daemon to mark the beginning of a new audit trail file and the end of an old file as it is deactivated. The audit daemon builds a special audit record containing this token to link together successive audit files into one audit trail. The fields are:

- A token ID
- A time and date stamp that identifies the time the file was created or closed
- A byte count of the file name including a null terminator (does not show)
- The file null-terminated name

The following figure shows the token format.

token ID	date & time	name length	previous/next file name
1 byte	8 bytes	2 bytes	*n* bytes

FIGURE B–10 file Token Format

A file token is displayed by praudit as follows:

file,Fri Jan 23 13:32:42 1997, + 792 msec,
/etc/security/audit/patchwork/files/19920901202558.19920901203241.patchwork

groups Token (Obsolete)

This token has been replaced by the newgroups token, which provides the same type of information but requires less space. A description of the groups token is provided here for completeness, but the application designer should use the newgroups token. Note that praudit does not distinguish between the two tokens as both token IDs are labelled groups when character output is displayed.

The groups token records the groups entries from the process's credential. The fields are:

- A token ID
- An array of groups entries of size NGROUPS_MAX (16)

The following figure shows a groups token.

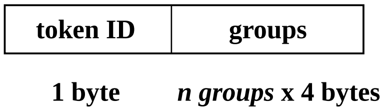


FIGURE B–11 groups Token Format

A groups token is displayed by praudit as follows:

group,staff,wheel,daemon,kmem,bin,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1

Note – The groups token is output only when the audit policy group is active.

---

header Token

The header token is special in that it marks the beginning of an audit record and combines with the trailer token to bracket all the other tokens in the record. The fields are:

- A token ID
- The record length in bytes, including the header and trailer tokens
- An audit record structure version number
- An event ID identifying the type of audit event from the /etc/security/audit_event file:
  - The praudit -l command displays the event description, for example, system booted.
  - The praudit -r command displays the event number, for example, 113.
  - The praudit -s command displays the event ID, for example, AUE_SYSTEMBOOT.
- An event ID modifier with descriptive information about the event type
- For extended headers, an IP address type

- For extended headers, the IP address of the source machine in IPv6 or IPv4 format
- The time and date the record was created

The following figure shows a `header` token.

token ID	byte count	version #	event ID	ID modifier	date and time
1 byte	4 bytes	1 byte	2 bytes	2 bytes	8 bytes

FIGURE B–12 header Token Format

The event modifier field has the following flags defined:

Value	Constant Name	Description
0x0001	PAD_READ	Data read from object
0x0002	PAD_WRITE	Data written to object
0x0080	PAD_SPRIVUSE	Successfully used privilege
0x0100	PAD_FPRIVUSE	Failed use of privilege
0x4000	PAD_NONATTR	Nonattributable event
0x8000	PAD_FAILURE	Failed audit event

For the Trusted Solaris 7 and Trusted Solaris 8 4/01 releases, the `header` token can be displayed with a 64-bit time stamp, in place of the 32-bit time stamp.

For the Trusted Solaris 8 4/01 release, the Internet Address can be displayed as a IPv4 address using 4 bytes, or as an IPv6 address using 16 bytes to describe the type, and 16 bytes to describe the address.

A `header` token is displayed by `praudit` as follows:

header,240,1,ioctl(2),,Tue Sept 7 16:11:44 2000, + 270 msec

host Token

The `host` token contains the machine ID for the system which generated this audit record. The fields are:

- A token ID
- The system ID of the host that generated the audit record

The following figure shows the token format.

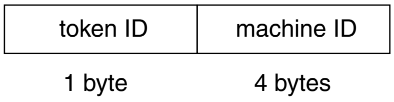


FIGURE B–13 host Token Format

A `host` token is displayed by `praudit` as follows:

host,patchwork

in_addr Token

The `in_addr` token contains an Internet address. This 4-byte value is an Internet Protocol address. The fields are:

- A token ID
- An Internet address

For the Trusted Solaris 8 4/01 release, the Internet Address can be displayed as a IPv4 address using 4 bytes, or as an IPv6 address using 16 bytes to describe the type, and 16 bytes to describe the address.

The following figure shows the token format.

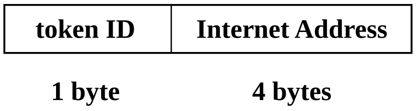


FIGURE B–14 in_addr Token Format

An `in_addr` token is displayed by `praudit` as follows:

ip address,192.168.110.3

ip Token

The `ip` token contains a copy of an Internet Protocol header but does not include any IP options. The IP options may be added by including more of the IP header in the token. The IP header structure is defined in `/usr/include/netinet/ip.h`. The fields are:

- A token ID
- A 20-byte copy of an IP header (all 20 bytes)

The following figure shows the token format.

token ID	IP header
1 byte	20 bytes

FIGURE B–15 ip Token Format

An ip token is displayed by praudit as follows:

ip,0.0.0.0

ipc Token

The ipc token contains the System V IPC message/semaphore/shared-memory handle used by the caller to identify a particular IPC object. The fields are:

- A token ID
- An IPC object type identifier
- The IPC object handle

The following figure shows the token format.

token ID	IPC object type	IPC object ID
1 byte	1 byte	4 bytes

FIGURE B–16 ipc Token Format

An ipc token is displayed by praudit as follows:

IPC,msg,3

Note – The IPC object identifiers violate the context-free nature of the Solaris CMW audit tokens. No global "name" uniquely identifies IPC objects; instead, they are identified by their handles, which are valid only during the time the IPC objects are active. The identification should not be a problem since the System V IPC mechanisms are seldom used and they all share the same audit class.

The IPC object type field may have the values shown in Table B–4. The values are defined in `</usr/include/bsm/audit.h>`.

TABLE B–4 IPC Object Type Field

Name	Value	Description
`AU_IPC_MSG`	1	IPC message object
`AU_IPC_SEM`	2	IPC semaphore object
`AU_IPC_SHM`	3	IPC shared memory object

ipc_perm Token

The `ipc_perm` token contains a copy of the System V IPC access information. Audit records for shared memory, semaphore, and message IPCs have this token added. The fields are:

- A token ID
- The IPC owner's user ID
- The IPC owner's group ID
- The IPC creator's user ID
- The IPC creator's group ID
- The IPC access modes
- The IPC sequence number
- The IPC key value

The values are taken from the ipc_perm structure associated with the IPC object. The following figure shows the token format.

token ID	owner uid	owner gid	creator uid	creator gid	ipc mode	sequence ID	IPC key
1 byte	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes

FIGURE B–17 ipc_perm Token Format

An ipc_perm token is displayed by praudit as follows:

IPC_perm,root,wheel,root,wheel,0,0,0x00000000

iport Token

The iport token contains the TCP (or UDP) port address. The fields are:

- A token ID
- A TCP/UDP address

The following figure shows the token format.

token ID	port ID
1 byte	2 bytes

FIGURE B–18 iport Token Format

An iport token is displayed by praudit as follows:

iport,0xf6d6

liaison Token

The liaison token contains a liaison ID used by the Trusted Networking software. The fields are:

- A token ID
- The liaison ID

The following figure shows the token format.

token ID	liaison ID
1 byte	4 bytes

FIGURE B–19 liaison Token Format

A liaison token is displayed by praudit as follows:

liaison,17

newgroups Token

This token is the replacement for the groups token. Note that praudit does not distinguish between the two tokens as both token IDs are labelled groups when character output is displayed.

The newgroups token records the groups entries from the process's credential. The fields are:

- A token ID field
- A count of the number of groups contained in this audit record.
- Zero or more group entries.

The following figure shows the token format.

token ID	count	groups
1 byte	**2 bytes**	***count** *** 4 bytes**

FIGURE B–20 newgroups Token Format

Note – The newgroups token is output only when the audit policy group is active.

A newgroups token is displayed by praudit as follows:

newgroups,1,analysts

opaque Token

The `opaque` token contains unformatted data as a sequence of bytes. The fields are:

- A token ID
- A byte count of the data array
- An array of byte data

The following figure shows the token format.

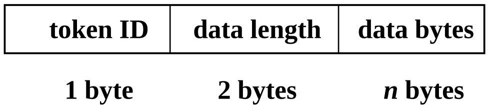


FIGURE B–21 opaque Token Format

An `opaque` token is displayed by `praudit` as follows:

opaque,12,0x4f5041515545204441544100

path Token

The `path` token contains access path information for an object. The fields are:

- A token ID
- A byte count of the path length (does not show)
- An absolute path to the object based on the real root of the system

The following figure shows the token format.

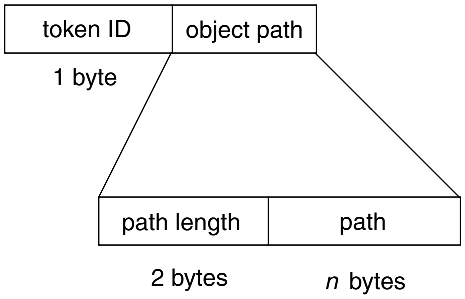


FIGURE B–22 path Token Format

A `path` token is displayed by `praudit` as follows:

path,/etc/security/audit/patchwork

privilege Token

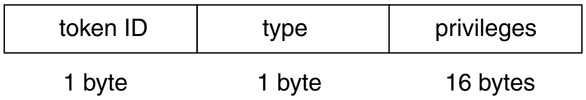
The `privilege` token contains privilege information for an object or a subject. The fields are:

- A token ID
- The type of privilege
- The privilege set

where type is one of the following:

Value	Type
0	Unknown or Undefined
1	Forced
2	Allowed
3	Effective
4	Inheritable
5	Permitted
6	Saved

The following figure shows the token format.



token ID	type	privileges
1 byte	1 byte	16 bytes

FIGURE B–23 privilege Token Format

A `privilege` token is displayed by `praudit` as follows:

privilege,Forced,proc_tcb_audit

process Token

The `process` token contains information describing a process as an object such as the recipient of a signal. The fields are:

- A token ID
- The user audit ID
- The effective user ID
- The effective group ID
- The real user ID
- The real group ID
- The process ID
- The audit session ID
- A terminal ID made up of
  - A device ID
  - A system ID

The audit ID, user ID, group ID, process ID, and session ID are long instead of short.

Note – The `process` token fields for the session ID, the real user ID, or the real group ID might be unavailable. The entry is then set to -1.

For the Trusted Solaris 7 release, the `process` token can be displayed using a 64-bit device ID, in place of the 32-bit value.

For the Trusted Solaris 8 4/01 release, the terminal ID can report an IPv6 address by changing the format to use either 4 or 8 bytes to describe the device, 16 bytes to describe the type, and 16 bytes to describe the address.

The following figure shows the token format.

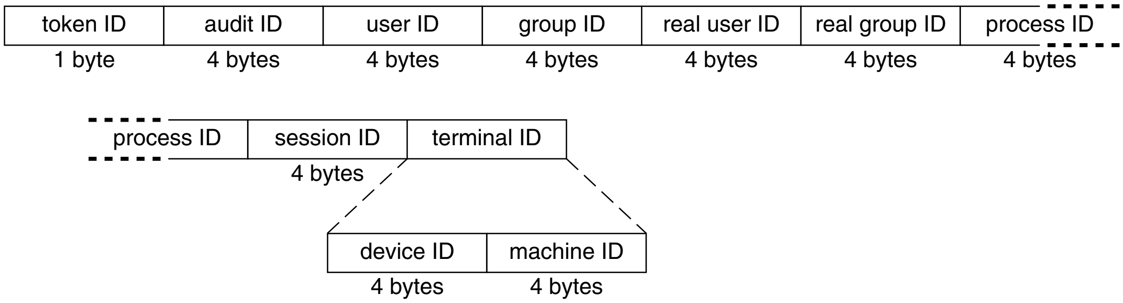


FIGURE B–24 Format for process and subject Tokens

subject

A `process` token is displayed by `praudit` as follows:

process,root,root,wheel,root,wheel,0,0,0,0.0.0.0

return Token

The `return` token contains the return status of the system call (`u_error`) and the process return value (`u_rval1`). The token indicates exit status and other return values in application auditing. This token is always returned as part of kernel-generated audit records for system calls. The fields are:

- A token ID
- The system call error status
- The system call return value

The following figure shows the token format.

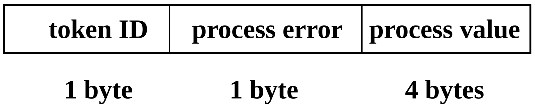


FIGURE B–25 return Token Format

A `return` token is displayed by `praudit` as follows:

return,failure: No such file or directory,-1

seq Token

The `seq` token (`sequence` token) is an optional token that contains an increasing sequence number. This token is for debugging. The token is added to each audit record when the `AUDIT_SEQ` policy is active. The fields are:

- A token ID
- A 32-bit unsigned long-sequence number

The sequence number is incremented every time an audit record is generated and put onto the audit trail. The following figure shows the token format.

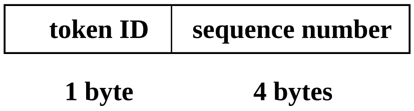


FIGURE B–26 seq Token Format

A `seq` token is displayed by `praudit` as follows:

sequence,1292

slabel Token

The `slabel` token contains a sensitivity label. The fields are:

- A token ID
- A sensitivity label

The following figure shows the token format.

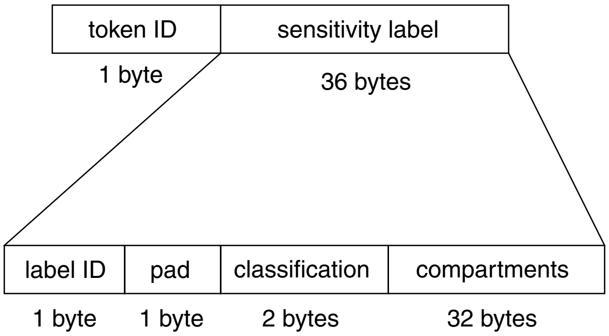


FIGURE B–27 slabel Token Format

An `slabel` token is displayed by `praudit` as follows:

slabel,ADMIN_LOW

socket Token

The `socket` token contains information describing an Internet socket. The fields are:

- A token ID
- A socket type field (TCP/UDP/UNIX)
- The local port address
- The local Internet address
- The remote port address
- The remote Internet address

For the Trusted Solaris 8 4/01 release, the Internet Address can be displayed as a IPv4 address using 4 bytes, or as an IPv6 address using 16 bytes to describe the type, and 16 bytes to descibe the addresses.

The socket type is taken from the designated socket and the port and Internet addresses are taken from the socket's inpcb control structure. The following figure shows the token format.

token ID	socket type	local port	local Internet address	remote port	remote Internet address
1 byte	2 bytes	2 bytes	4 bytes	2 bytes	4 bytes

FIGURE B–28 socket Token Format

A socket token is displayed by praudit as follows:

socket,0x0000,0x0000,0.0.0.0,0x0000,0.0.0.0

socket,0x0002,0x8008,patchwork

subject Token

The subject token describes a subject (process). The structure is the same as the process token:

- A token ID
- The user audit ID
- The effective user ID
- The effective group ID
- The real user ID
- The real group ID
- The process ID
- The session ID
- A terminal ID made up of
  - A device ID
  - A system ID

This token is always returned as part of kernel-generated audit records for system calls. The audit ID, user ID, group ID, process ID, and session ID are long instead of short. Figure B–24 shows the token format.

Note – The subject token fields for the session ID, the real user ID, or the real group ID may be unavailable. The entry is then set to –1.

For the Trusted Solaris 7 release, the process token can be displayed using a 64-bit device ID, in place of the 32-bit value.

For the Trusted Solaris 8 4/01 release, the terminal ID can report an IPv6 address by changing the format to use either 4 or 8 bytes to describe the device, 16 bytes to describe the type, and 16 bytes to describe the address.

A subject token is displayed by praudit as follows:

subject,root,root,staff,root,staff,552,552,24 3 patchwork

text Token

The text token contains a text string. The fields are:

- A token ID
- The length of the text string (does not show)
- A text string

The following figure shows the token format.

token ID	text length	text string
1 byte	1 byte	*n* bytes

FIGURE B–29 text Token Format

A text token is displayed by praudit in 7–bit ASCII with control characters in the form ^*L*, as follows:

text,Enter your name on the next line^JName:

trailer Token

A trailer token it marks the end of an audit record to support backward seeks of the audit trail. It is an optional token that is added as the last token of each record only when the AUDIT_TRAIL audit policy has been set. The fields are:

- A token ID
- A pad number that marks the end of the record (does not show)

- The total number of audit record characters including the `header` and `trailer` tokens

The following figure shows the token format.

token ID	pad number	byte count
1 byte	2 bytes	4 bytes

FIGURE B–30 trailer Token Format

A `trailer` token is displayed by `praudit` as follows:

trailer,136

uauth Token

The `uauth` token contains a text string. The fields are:

- A token ID
- The length of the text string (does not show)
- A text string

The following figure shows the token format.

token ID	text length	text string
1 byte	1 byte	*n* bytes

FIGURE B–31 uauth Token Format

A `uauth` token is displayed by `praudit` as follows:

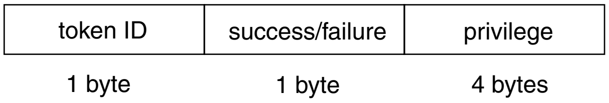
uauth,solaris.device.allocate

upriv Token

The `upriv` token contains use of privilege information. The fields are:

- A token ID
- A success/failure field indicating whether the use of privilege was successful (1 success, 0 failure)
- The privilege being tested

The following figure shows a priv token.



token ID	success/failure	privilege
1 byte	1 byte	4 bytes

FIGURE B–32 upriv Token Format

A upriv token is displayed by praudit as follows:

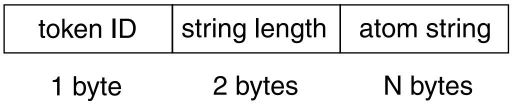
use of privilege,failed use of priv,win_mac_write

xatom Token

The xatom token contains information concerning an X atom. The fields are:

- A token ID
- The string length
- A text string identifying the atom

The following figure shows the token format.



token ID	string length	atom string
1 byte	2 bytes	N bytes

FIGURE B–33 xatom Token Format

An xatom token is displayed by praudit as follows:

X atom,_DT_SAVE_MODE

xclient Token

The xclient token contains information concerning the X client. The fields are:

- A token ID
- The client ID

The following figure shows the token format.

token ID	client ID
1 byte	4 bytes

FIGURE B–34 xclient Token Format

An xclient token is displayed by praudit as follows:

X client,15

xcolormap Token

The xcolormap token contains information about the colormaps. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

The following figure shows the token format.

token ID	XID	creator UID
1 byte	4 bytes	4 bytes

FIGURE B–35 Format for xcolormap, xcursor, xfont, xgc, xpixmap, and xwindow Tokens

An xcolormap token is displayed by praudit as follows:

X color map,0x08c00005,srv

xcursor Token

The xcursor token contains information about the cursors. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

Figure B–35 shows the token format.

An xcursor token is displayed by praudit as follows:

X cursor,0x0f400006,srv

xfont Token

The xfont token contains information about the fonts. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

Figure B–35 shows the token format.

An xfont token is displayed by praudit as follows:

X font,0x08c00001,srv

xgc Token

The xgc token contains information about the xgc. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

Figure B–35 shows the token format.

An xgc token is displayed by praudit as follows:

Xgraphic context,0x002f2ca0,srv

xpixmap Token

The xpixmap token contains information about the pixel mappings. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

Figure B–35 shows the token format.

An xpixmap token is displayed by praudit as follows:

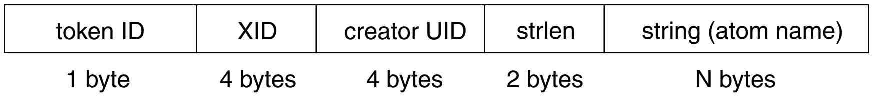
X pixmap,0x08c00005,srv

xproperty Token

The xproperty token contains information about various properties of a window. The fields are:

- A token ID
- The X server identifier
- The creator's user ID
- A string length
- A string (atom name)

The following figure shows an xproperty token format.



token ID	XID	creator UID	strlen	string (atom name)
1 byte	4 bytes	4 bytes	2 bytes	N bytes

FIGURE B–36 xproperty Token Format

An xproperty token is displayed by praudit as follows:

X property,0x000075d5,root,_MOTIF_DEFAULT_BINDINGS

xselect Token

The xselect token contains the data moved between windows. This data is a byte stream with no assumed internal structure, and a property string. The fields are:

- A token ID
- The length of the property string
- The property string
- A length for the property type
- The property type string
- A length field that gives the number of bytes of data
- A byte string containing the data

The following figure shows the token format.

token ID	property length	prop string	prop type len	prop type	data length	window data
1 byte	2 bytes	N bytes	2 bytes	N bytes	2 bytes	N bytes

FIGURE B–37 xselect Token Format

An `xselect` token is displayed by `praudit` as follows:

X selection,

xwindow Token

The `xwindow` token contains information about a window. The fields are:

- A token ID
- The X server identifier
- The creator's user ID

Figure B–35 shows the token format.

An `xwindow` token is displayed by `praudit` as follows:

X window,0x07400001,gww

Audit Records

General Audit Record Structure

The audit records produced by Trusted Solaris auditing software have a sequence of tokens. Certain tokens are optional within an audit record, according to the current audit policy. The `group`, `sequence`, and `trailer` tokens fall into this category. The administrator can determine if these are included in an audit record with the `auditconfig` command `-getpolicy` option.

Kernel-Level Generated Audit Records

These audit records are created by system calls which are used by the kernel. The records are sorted alphabetically by system call. The description of each record includes:

- The name of the system call
- A man page reference (if appropriate)
- The audit event number
- The audit event name
- The audit event class
- The mask for the event class
- The audit record structure

TABLE B–5 `access`(2)

Event Name	Event ID	Event Class	Mask
`AUE_ACCESS`	`14`	`fa`	0x00000004
Format: *header-token* *path-token*[*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–6 `acct`(2)

Event Name	Event ID	Event Class	Mask
`AUE_ACCT`	`18`	`as`	0x00020000
Format (zero path): *header-token* *argument-token* (1, "accounting off", 0) [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (non-zero path): *header-token* *path-token* [*attr-token*] *subject-token* *return-token*			

TABLE B–7 `adjtime`(2)

Event Name	Event ID	Event Class	Mask
`AUE_ADJTIME`	`50`	`as`	0x00000800
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–8 `audit`(2)

Event Name	Event ID	Event Class	Mask
`AUE_AUDIT`	`211`	`no`	0x00000000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–9 `auditon`(2) — get current active root

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETCAR`	`224`	`aa`	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–10 `auditon`(2) — get event class

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETCLASS`	`231`	`aa`	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–11 `auditon`(2) — get audit state

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETCOND`	`229`	`aa`	0x00040000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–12 `auditon`(2) — get current working directory

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETCWD`	`223`	`aa`	0x00040000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–13 `auditon`(2) — get kernel mask

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETKMASK`	`221`	`aa`	0x00040000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–14 `auditon`(2) — get audit statistics

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GETSTAT`	`225`	`aa`	0x00040000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–15 `auditon`(2) — GETPOLICY command

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GPOLICY`	`114`	`aa`	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–16 `auditon`(2) — get audit queue control parameters

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_GQCTRL`	`145`	`aa`	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–17 `auditon`(2) — set event class

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_SETCLASS`	`232`	`aa`	0x00040000
Format: *header-token* [*argument-token*] (2, "setclass:ec_event", event number) [*argument-token*] (3, "setclass:ec_class", class mask) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–18 `auditon`(2) — set audit state

Event Name	Event ID	Event Class	Mask
`AUE_AUDITON_SETCOND`	`230`	`aa`	0x00040000
Format: *header-token* [*argument-token*] (3, "setcond", audit state) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–19 auditon(2) — set kernel mask

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SETKMASK	222	aa	0x00040000
Format: *header-token* [*argument-token*] (2, "setkmask:as_success", kernel mask) [*argument-token*] (2, "setkmask:as_failure", kernel mask) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–20 auditon(2) — set mask per session ID

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SETSMASK	228	aa	0x00040000
Format: *header-token* [*argument-token*] (3, "setsmask:as_success", session ID mask) [*argument-token*] (3, "setsmask:as_failure", session ID mask) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–21 auditon(2) — reset audit statistics

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SETSTAT	226	aa	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–22 auditon(2) — set mask per uid

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SETUMASK	227	aa	0x00040000

TABLE B–22 auditon(2) — set mask per uid *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* [*argument-token*] (3, "setumask:as_success", audit ID mask) [*argument-token*] (3, "setumask:as_failure", audit ID mask) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–23 auditon(2) — SETPOLICY command

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SPOLICY	147	aa	0x00040000
Format: *header-token* [*argument-token*] (1, "policy", audit policy flags) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–24 auditon(2) — set audit queue control parameters

Event Name	Event ID	Event Class	Mask
AUE_AUDITON_SQCTRL	146	aa	0x00040000
Format: *header-token* [*argument-token*] (3,"setqctrl:aq_hiwater",queue control param.) [*argument-token*] (3,"setqctrl:aq_lowater",queue control param.) [*argument-token*] (3,"setqctrl:aq_bufsz",queue control param.) [*argument-token*] (3,"setqctrl:aq_delay",queue control param.) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–25 auditpsa(2)

Event Name	Event ID	Event Class	Mask
AUE_AUDITPSA	529	aa	0x00040000

TABLE B–25 auditpsa(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (valid file descriptor): *header-token* *argument-token* (1, "op", state) *in_addr-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–26 auditstat(2)

Event Name	Event ID	Event Class	Mask
`AUE_AUDITSTAT`	`150`	`aa`	0x00040000
Format: *header-token* [*argument-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–27 `auditsvc`(2)

Event Name	Event ID	Event Class	Mask
`AUE_AUDITSVC`	`136`	`aa`	0x00040000
Format (valid file descriptor): *header-token* [*path-token*] [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (invalid file descriptor): *header-token* *argument-token* (1, "no path: fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–28 `chdir`(2)

Event Name	Event ID	Event Class	Mask
`AUE_CHDIR`	`8`	`pm`	0x00200000

TABLE B–28 chdir(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–29 chmod(2)

Event Name	Event ID	Event Class	Mask
AUE_CHMOD	10	fm	0x00000008
Format: *header-token* *argument-token* (2, "new file mode", mode) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–30 chown(2)

Event Name	Event ID	Event Class	Mask
AUE_CHOWN	11	fm	0x00000008
Format: *header-token* *argument-token* (2, "new file uid", uid) *argument-token* (3, "new file gid", gid) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–31 `chroot`(2)

Event Name	Event ID	Event Class	Mask
`AUE_CHROOT`	`24`	`pm`	0x00200000
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–32 `chstate`(2)

Event Name	Event ID	Event Class	Mask
`AUE_CHSTATE`	`538`	`as`	0x00000800
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–33 `clock_settime`(3R)

Event Name	Event ID	Event Class	Mask
`AUE_CLOCK_SETTIME`	`513`	`as`	0x00000800
Format: *header-token* *slabel-token* *return-token*			

TABLE B–34 `close`(2)

Event Name	Event ID	Event Class	Mask
`AUE_CLOSE`	`112`	`cl`	0x00000040

TABLE B–34 `close`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: <file system object> *header-token* *argument-token* (1, "fd", file descriptor) [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			
Also for files closed on process termination. The *argument-token* is only present with the close() system call. It may be removed in future releases. The *path-token* is present only with valid file descriptors.			

TABLE B–35 `creat`(2)

Event Name	Event ID	Event Class	Mask
`AUE_CREAT`	`4`	`fc`	0x00000010
Format *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–36 `devpolicy`(2)

Event Name	Event ID	Event Class	Mask
`AUE_DRVPOLICY`	`531`	`as`	0x00000800
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–37 `enter prom`, `exit prom`

Event Name	Event ID	Event Class	Mask
`AUE_ENTERPROM`	`153`	`na`	0x00000400
`AUE_EXITPROM`	`154`	`na`	0x00000400
Format: *header-token* *text-token* (addr, "monitor PROM" | "kadb") [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–38 `exec`(2), `execve`(2)

Event Name	Event ID	Event Class	Mask
`AUE_EXEC`	`7`	`ps`	0x00100000
`AUE_EXECVE`	`23`	`ps`	0x00100000
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–39 `exit`(2)

Event Name	Event ID	Event Class	Mask
`AUE_EXIT`	`1`	`pm`	0x00200000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–40 fauditpsa(2)

Event Name	Event ID	Event Class	Mask
`AUE_FAUDITPSA`	`530`	`aa`	0x00040000

TABLE B–40 fauditpsa(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–41 `fchdir`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FCHDIR`	`68`	`pc`	0x00300000
Format: *header-token* [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–42 `fchmod`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FCHMOD`	`39`	`fm`	0x00000008
Format (valid file descriptor): *header-token* *argument-token* (2, "new file mode", mode) [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (invalid file descriptor): *header-token* *argument-token* (2, "new file mode", mode) *argument-token* (1, "no path: fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–43 `fchown`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FCHOWN`	`38`	`fm`	0x00000008
Format (valid file descriptor): *header-token* *argument-token* (2, "new file uid", uid) *argument-token* (3, "new file gid", gid) [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (non-file descriptor): *header-token* *argument-token* (2, "new file uid", uid) *argument-token* (3, "new file gid", gid) *argument-token* (1, "no path: fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–44 `fchroot`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FCHROOT`	`69`	`pm`	0x00200000
Format: *header-token* [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–45 `fcntl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FCNTL (cmd=F_GETLK, F_SETLK,F_SETLKW)`	`30`	`fn`	0x40000000

TABLE B–45 `fcntl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (file descriptor): *header-token* *argument-token* (2, "cmd", cmd) *path-token* *attr-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (bad file descriptor): *header-token* *argument-token* (2, "cmd", cmd) *argument-token* (1, "no path: fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–46 `fgetsldname`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FGETSLDNAME`	`532`	`fc`	0x00000010
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–47 `fork`(2), `fork1`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FORK`	`2`	`ps`	0x00100000
`AUE_FORK1`	`241`	`ps`	0x00100000
Format: *header-token* [*argument-token*] (0, "child PID", pid) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* The `fork()` and `fork1()` return values are undefined since each audit record is produced at the point that the child process is spawned.			

TABLE B–48 `fsetcmwlabel`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FSETCMWLABEL`	`544`	`fm`	0x00000008
Format: *header-token* *argument-token* (3, “flag”, which parts of label to set) [*slabel-token*] (if slabel is being set) *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–49 `fsetfattrflag`(2)

Event Name	Event ID	Event Class	Mask
`AUE_FSETFATTRFLAG`	`523`	`fm`	0x00000008
Format: *header-token* *argument-token* (2, "which", which flags to set) *argument-token* (3, "attrs", flag values) *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–50 fstatfs(2)

Event Name	Event ID	Event Class	Mask
`AUE_FSTATFS`	`55`	`fa`	0x00000004

TABLE B–50 fstatfs(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (file descriptor): *header-token* [*path-token*] [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (non-file descriptor): *header-token* *argument-token* (1, "no path: fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–51 `getaudit`(2)

Event Name	Event ID	Event Class	Mask
`AUE_GETAUDIT`	`132`	`aa`	0x00040000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–52 `getaudit_addr`(2)

Event Name	Event ID	Event Class	Mask
`AUE_GETAUDIT_ADDR`	`267`	`aa`	0x00000800
Format: *header-token* *subject-token* *return-token*			

TABLE B–53 `getauid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_GETAUID`	`130`	`aa`	0x00040000

TABLE B–53 getauid(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–54 getcmwfsrange(2)

Event Name	Event ID	Event Class	Mask
AUE_GETCMWFSRANGE	545	fa	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–55 getcmwlabel(2), fgetcmwlabel(2), lgetcmwlabel(2)

Event Name	Event ID	Event Class	Mask
AUE_GETCMWLABEL	546	fa	0x00000004
AUE_FGETCMWLABEL	118	fa	0x00000004
AUE_LGETCMWLABEL	548	fa	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–56 getdents(2)

Event Name	Event ID	Event Class	Mask
AUE_GETDENTS	193	no	0x00000000

TABLE B–56 getdents(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–57 getfpriv(2)

Event Name	Event ID	Event Class	Mask
AUE_GETFILEPRIV	547	fa	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–58 getmldadorn(2)

Event Name	Event ID	Event Class	Mask
AUE_GETMLDADORN	554	fa	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–59 getmsg(2)

Event Name	Event ID	Event Class	Mask
AUE_GETMSG	217	nt	0x00000100

TABLE B–59 `getmsg`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "fd", file descriptor) *argument-token* (4, "pri", priority) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–60 `getmsg`(2) — accept, receive

Event Name	Event ID	Event Class	Mask
`AUE_SOCKACCEPT`	`247`	`nt`	0x00000100
`AUE_SOCKRECEIVE`	`250`	`nt`	0x00000100
Format: *header-token* *socket-inet-token* *argument-token* (1, "fd", file descriptor) *argument-token* (4, "pri", priority) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–61 `getmsgqcmwlabel`(2)

Event Name	Event ID	Event Class	Mask
`AUE_GETMSGQCMWLABEL`	`514`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "msg ID", message ID) [*argument-token*] [*ipc_perm-token*] (of the IPC) [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			
The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.			

TABLE B–62 getpmsg(2)

Event Name	Event ID	Event Class	Mask
AUE_GETPMSG	219	nt	0x00000100

Format:
header-token
argument-token (1, "fd", file descriptor)
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–63 getportaudit(2)

Event Name	Event ID	Event Class	Mask
AUE_GETPORTAUDIT	149	aa	0x00040000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–64 getsemcmwlabel(2)

Event Name	Event ID	Event Class	Mask
AUE_GETSEMCMWLABEL	515	ip	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*] (of the IPC)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the sem ID is invalid.

TABLE B–65 getshmcmwlabel(2)

Event Name	Event ID	Event Class	Mask
AUE_GETSHMCMWLABEL	516	ip	0x00000200

TABLE B–65 `getshmcmwlabel`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "shm ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*] (of the IPC)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the shm ID is invalid.

TABLE B–66 `getsldname`(2)

Event Name	Event ID	Event Class	Mask
`AUE_GETSLDNAME`	`555`	`fa`	0x00000004

Format:
header-token
path-token
[*attr-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–67 `ioctl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_IOCTL`	`158`	`io`	0x20000000

TABLE B–67 `ioctl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (good file descriptor): *header-token* *path-token* [*attr-token*] *argument-token* (2, "cmd" ioctl cmd) *argument-token* (3, "arg" ioctl arg) [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (socket): *header-token* [*socket-token*] *argument-token* (2, "cmd" ioctl cmd) *argument-token* (3, "arg" ioctl arg) [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (non-file file descriptor): *header-token* *argument-token* (1, "fd", file descriptor) *argument-token* (2, "cmd" ioctl cmd) *argument-token* (3, "arg" ioctl arg) [*priv-token*] (if privilege used or required) *subject-token* *return-token* Format (bad file name): *header-token* *argument-token* (1, "no path: fd", file descriptor) *argument-token* (2, "cmd" ioctl cmd) *argument-token* (3, "arg" ioctl arg) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–68 `kill`(2)

Event Name	Event ID	Event Class	Mask
`AUE_KILL`	`15`	`pm`	0x00200000

TABLE B–68 `kill`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (valid process): *header-token* *argument-token* (2, "signal", signo) [*process-token*] [*slabel-token*] (process) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			
Format (zero or negative process): *header-token* *argument-token* (2, "signal", signo) *argument-token* (1, "process", pid) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–69 `lchown`(2)

Event Name	Event ID	Event Class	Mask
`AUE_LCHOWN`	`237`	`fm`	0x00000008
Format: *header-token* *argument-token* (2, "new file uid", uid) *argument-token* (3, "new file gid", gid) *path-token* [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–70 `link`(2)

Event Name	Event ID	Event Class	Mask
`AUE_LINK`	`5`	`fc`	0x00000010

TABLE B–70 `link`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* (from path) [*attr-token*] (from path) [*slabel-token*] (from path) *path-token* (to path) [*attr-token*] (to path) [*slabel-token*] (to path) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–71 `lstat`(2)

Event Name	Event ID	Event Class	Mask
`AUE_LSTAT`	`17`	`fa`	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–72 lxstat(2)

Event Name	Event ID	Event Class	Mask
`AUE_LXSTAT`	`236`	`fa`	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–73 `memcntl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MEMCNTL`	`238`	`ot`	0x80000000

TABLE B–73 memcntl(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "base", base address) *argument-token* (2, "len", length) *argument-token* (3, "cmd", command) *argument-token* (4, "arg", command args) *argument-token* (5, "attr", command attributes) *argument-token* (6, "mask", 0) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–74 mkdir(2)

Event Name	Event ID	Event Class	Mask
AUE_MKDIR	47	fc	0x00000010
Format: *header-token* *argument-token* (2, "mode", mode) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–75 mknod(2)

Event Name	Event ID	Event Class	Mask
AUE_MKNOD	9	fc	0x00000010
Format: *header-token* *argument-token* (2, "mode", mode) *argument-token* (3, "dev", dev) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–76 `mldsetfattrflag`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MLDSETFATTRFLAG`	`524`	`fm`	0x00000008

Format:
header-token
argument-token (2, “which”, which flags to set)
argument-token (3, “attrs”, flag values)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–77 `mmap`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MMAP`	`210`	`no`	0x00000000

Format (valid file descriptor):
header-token
argument-token (1, "addr", segment address)
argument-token (2, "len", segment length)
[*path-token*]
[*attr-token*]
[*priv-token*] (if privilege used or required)
subject-token
return-token
Format (invalid file descriptor):
header-token
argument-token (1, "addr", segment address)
argument-token (2, "len", segment length)
argument-token (1, "no path: fd", file descriptor)
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–78 modctl(2) — bind module

Event Name	Event ID	Event Class	Mask
`AUE_MODADDMAJ`	`246`	`as`	0x00000800

TABLE B–78 modctl(2) — bind module *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* [*text-token*] (driver major number) [*text-token*] (driver name) *text-token* (root dir. | "no rootdir") *text-token* (driver major number | "no drvname") *argument-token* (5, "", number of aliases) (0..n)[*text-token*] (aliases) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–79 modctl(2) — configure module

Event Name	Event ID	Event Class	Mask
`AUE_MODCONFIG`	`245`	`as`	0x00000800
Format: *header-token* *text-token* (root dir. | "no rootdir") *text-token* (driver major number | "no drvname") [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–80 modctl(2) — load module

Event Name	Event ID	Event Class	Mask
`AUE_MODLOAD`	`243`	`as`	0x00020000
Format: *header-token* [*text-token*] (default path) *text-token* (filename path) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–81 modctl(2) — unload module

Event Name	Event ID	Event Class	Mask
`AUE_MODUNLOAD`	`244`	`as`	0x00020000

TABLE B–81 modctl(2) — unload module *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "id", module ID) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–82 `mount`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MOUNT`	`62`	`ao`	0x00080000
Format (UNIX file system): *header-token* *argument-token* (3, "flags", flags) *text-token* (filesystem type) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (NFS file system): *header-token* *argument-token* (3, "flags", flags) *text-token* (filesystem type) *text-token* (host name) *argument-token* (3, "internal flags", flags) *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–83 `msgctl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MSGCTL`	`84`	`ip`	0x00000200

TABLE B–83 `msgctl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "msg ID", message ID)
[*ipc-token*]
subject-token
return-token

The `ipc` and `ipc_perm` tokens are not included if the msg ID is not valid.

TABLE B–84 `msgctl`(2) — IPC_RMID command

Event Name	Event ID	Event Class	Mask
`AUE_MSGCTL_RMID`	`85`	`ip`	0x00000200

Format:
header-token
argument-token (1, "msg ID", message ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.

TABLE B–85 `msgctl`(2) — IPC_SET command

Event Name	Event ID	Event Class	Mask
`AUE_MSGCTL_SET`	`86`	`ip`	0x00000200

TABLE B–85 `msgctl`(2) — IPC_SET command *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "msg ID", message ID)
[*argument-token*]
[*ipc_perm-token*] (of the IPC's old values)
[*slabel-token*]
[*ipc_perm-token*] (of the IPC's new values)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
subject-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.

TABLE B–86 `msgctl`(2) — IPC_STAT command

Event Name	Event ID	Event Class	Mask
`AUE_MSGCTL_STAT`	`87`	`ip`	0x00000200

Format:
header-token
argument-token (1, "msg ID", message ID)
[*argument-token*]
[*ipc_perm-token*] (of the IPC)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.

TABLE B–87 `msgget`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MSGGET`	`88`	`ip`	0x00000200

TABLE B–87 `msgget`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "msg key", message key) *argument-token* (2, "msg flag", message flags) [*ipc_perm-token*] (of the IPC object) [*slabel-token*] [*argument-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.			

TABLE B–88 `msggetl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MSGGETL`	`174`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "msg key", message key) *argument-token* (2, "msg flag", message flags) *slabel-token* (desired SL) [*ipc_perm-token*] (of the IPC object) [*slabel-token*] [*argument-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.			

TABLE B–89 `msgrcv`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MSGRCV`	`89`	`ip`	0x00000200
`AUE_MSGRCVL`	`175`	`ip`	0x00000200

TABLE B–89 `msgrcv`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "msg ID", message ID) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			
The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.			

TABLE B–90 `msgsnd`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MSGSND`	`90`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "msg ID", message ID) [*argument-token*] [*ipc_perm-token*] (of the IPC's new values) [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			
The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the msg ID is invalid.			

TABLE B–91 `munmap`(2)

Event Name	Event ID	Event Class	Mask
`AUE_MUNMAP`	`214`	`cl`	0x00000040
Format: *header-token* *argument-token* (1, "addr", address of memory) *argument-token* (2, "len", memory segment size) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–92 old `nice`(2)

Event Name	Event ID	Event Class	Mask
`AUE_NICE`	`203`	`pc`	0x00300000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–93 `open`(2) — read

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_R`	`72`	`fr`	0x00000001
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–94 `open`(2) — read,creat

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RC`	`73`	`fc,fr`	0x00000011
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–95 `open`(2) — read,trunc,creat

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RTC`	`75`	`fc,fd,fr`	0x00000031

TABLE B–95 `open`(2) — read,trunc,creat *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–96 `open`(2) — read,trunc

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RT`	`74`	`fd,fr`	0x00000021
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–97 `open`(2) — read,write

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RW`	`80`	`fr,fw`	0x00000003
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–98 `open`(2) — read,write,creat

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RWC`	`81`	`fr,fw,fc`	0x00000013

TABLE B–98 open(2) — read,write,creat *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–99 open(2) — read,write,trunc,creat

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RWTC`	`83`	`fr,fw,fc,fd`	0x00000033
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–100 open(2) — read,write,trunc

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_RWT`	`82`	`fr,fw,fd`	0x00000023
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–101 open(2) — write

Event Name	Event ID	Event Class	Mask
`AUE_OPEN_W`	`76`	`fw`	0x00000002

TABLE B–101 open(2) — write *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–102 open(2) — write,creat

Event Name	Event ID	Event Class	Mask
AUE_OPEN_WC	77	fw,fc	0x00000012
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–103 open(2) — write,trunc,creat

Event Name	Event ID	Event Class	Mask
AUE_OPEN_WTC	79	fw,fc,fd	0x00000032
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–104 open(2) — write,trunc

Event Name	Event ID	Event Class	Mask
AUE_OPEN_WT	78	fw,fd	0x00000022

TABLE B–104 `open`(2) — write,trunc *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–105 `pathconf`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PATHCONF`	`71`	`fa`	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–106 `pipe`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PIPE`	`185`	`no`	0x00000000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–107 `preadl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PREADL`	`527`	`no`	0x00000000

TABLE B–107 `preadl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–108 `priocntl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PRIOCNTLSYS`	`212`	`pm`	0x00200000
Format: *header-token* *argument-token* (1, "pc_version", priocntl version num.) *argument-token* (3,"cmd", command) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–109 `processor_bind`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PROCESSOR_BIND`	`263`	`ao`	0x00080000
Format: *header-token* *slabel-token* *return-token*			

TABLE B–110 privilege enable

Event Name	Event ID	Event Class	Mask
`AUE_PRIVENABLE`	`533`	`as`	0x00020000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–111 process dumped core

Event Name	Event ID	Event Class	Mask
`AUE_CORE`	`111`	`fc`	0x0000010
Format: *header-token* *path-token* [*attr-token*] *argument-token* (1, "signal", signal) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–112 `putmsg`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PUTMSG`	`216`	`nt`	0x00000100
Format: *header-token* *argument-token* (1, "fd", file descriptor) *argument-token* (4, "pri", priority) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–113 `putmsg`(2) - connect, send

Event Name	Event ID	Event Class	Mask
`AUE_SOCKCONNECT`	`248`	`nt`	0x00000100
`AUE_SOCKSEND`	`249`	`nt`	0x00000100
Format: *header-token* *socket-inet-token* *argument-token* (1, "fd", file descriptor) *argument-token* (4, "pri", priority) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–114 `putpmsg`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PUTPMSG`	`218`	`nt`	0x00000100

TABLE B–114 `putpmsg`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "fd", file descriptor) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–115 `quotactl`(7I)

Event Name	Event ID	Event Class	Mask
`AUE_QUOTACTL`	`60`	`ao`	0x00080000
Format: *header-token* *subject-token* *return-token*			

TABLE B–116 `read`(2), `readl`(2), `readvl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_READ`	`192`	`no`	0x00000000
`AUE_READL`	`558`		
`AUE_READVL`	`559`		
Format: *header-token* *path-token*) [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–117 `readlink`(2)

Event Name	Event ID	Event Class	Mask
`AUE_READLINK`	`22`	`fr`	0x00000001

TABLE B–117 `readlink`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–118 `recvmsg`(3SOCKET)

Event Name	Event ID	Event Class	Mask
`AUE_RECVMSG`	`190`	`nt`	0x00000100
Format: *header-token* *sock-inet-token* *argument-token* (3, "flags", message flags) *sock-inet-token* (from address) *subject-token* *return-token*			
The `sock_inet` token for a bad socket is reported as: *argument-token* (1, "fd", socket descriptor)			

TABLE B–119 `rename`(2)

Event Name	Event ID	Event Class	Mask
`AUE_RENAME`	`42`	`fc,fd`	0x00000030
Format: *header-token* *path-token* (from name) [*attr-token*] (from name) [*slabel-token*] (from name) [*path-token*] (to name) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–120 `rmdir`(2)

Event Name	Event ID	Event Class	Mask
`AUE_RMDIR`	`48`	`fd`	0x00000020

TABLE B–120 `rmdir`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–121 `semctl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL`	`98`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "sem ID", semaphore ID) [*ipc-token*] *subject-token* *return-token* The `ipc` and `ipc_perm` tokens are not included if the semaphore ID is not valid.			

TABLE B–122 `semctl`(2) — getall

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_GETALL`	`105`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "sem ID", semaphore ID) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.			

TABLE B–123 `semctl`(2) — GETNCNT command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_GETNCNT`	`102`	`ip`	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–124 `semctl`(2) — GETPID command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_GETPID`	`103`	`ip`	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–125 `semctl`(2) — GETVAL command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_GETVAL`	`104`	`ip`	0x00000200

TABLE B–125 `semctl`(2) — GETVAL command *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–126 `semctl`(2) — GETZCNT command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_GETZCNT`	`106`	`ip`	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–127 `semctl`(2) — IPC_RMID command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_RMID`	`99`	`ip`	0x00000200

TABLE B–127 `semctl`(2) — IPC_RMID command *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–128 `semctl`(2) — IPC_SET command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_SET`	`100`	`ip`	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*] (of the IPC’s old values)
[*slabel-token*]
[*ipc_perm-token*] (of the IPC’s new values)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–129 `semctl`(2) — SETALL command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_SETALL`	`108`	`ip`	0x00000200

TABLE B–129 `semctl`(2) — SETALL command *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–130 `semctl`(2) — SETVAL command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_SETVAL`	`107`	`ip`	0x00000200

Format:
header-token
argument-token (1, "sem ID", semaphore ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.

TABLE B–131 `semctl`(2) — IPC_STAT command

Event Name	Event ID	Event Class	Mask
`AUE_SEMCTL_STAT`	`101`	`ip`	0x00000200

TABLE B–131 `semctl`(2) — IPC_STAT command *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "sem ID", semaphore ID) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–132 `semget`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SEMGET`	`109`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "sem key", semaphore key) *argument-token* (3, "sem flags", semaphore flags) [*ipc_perm-token*] [*slabel-token*] [*argument-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.			

TABLE B–133 `semgetl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SEMGETL`	`177`	`ip`	0x00000200

TABLE B–133 `semgetl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "sem key", semaphore key) *argument-token* (3, "sem flags", semaphore flags) *slabel-token* [*ipc_perm-token*] [*slabel-token*] [*argument-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the system call failed.			

TABLE B–134 `semop`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SEMOP`	`110`	`ip`	0x00000200
Format: *header-token* *argument-token* (1, "sem ID", semaphore ID) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and the *slabel* of the ipc tokens are not included if the semaphore ID is invalid.			

TABLE B–135 `sendmsg`(3SOCKET)

Event Name	Event ID	Event Class	Mask
`AUE_SENDMSG`	`188`	`nt`	0x00000100

TABLE B–135 `sendmsg`(3SOCKET) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
sock-inet-token
sock-inet-token (to address)
argument-token (3, "flags", message flags)
subject-token
return-token

The `sock_inet` token for a bad socket is reported as: *argument-token* (1, "fd", socket descriptor)

TABLE B–136 `sendto`(3SOCKET)

Event Name	Event ID	Event Class	Mask
`AUE_SENDTO`	`184`	`nt`	0x00000100

Format:
header-token
sock-inet-token
argument-token (3, "len", message length)
[*argument-token*] (4, "flags", flags)
argument-token (6, "tolen", address length)
sock-inet-token (to address)
subject-token
return-token

The `sock_inet` token for a bad socket is reported as: *argument-token* (1, "fd", socket descriptor)

TABLE B–137 setacl(1), `setfacl`(1)

Event Name	Event ID	Event Class	Mask
`AUE_ACLSET`	`251`	`fm`	0x00000008
`AUE_FACLSET`	`252`	`fm`	0x00000008

Format:
header-token
argument-token (2,"cmd", command)
argument-token (3,"*n*_entries", number of acl entries)
acl-token ... (token repeated "*n*_entries" times)
path-token
[attr-token]
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–138 `setaudit`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETAUDIT`	`133`	`aa`	0x00040000

Format (valid program stack address):
header-token
argument-token (1, "setaudit:auid", audit user ID)
argument-token (1, "setaudit:port", terminal ID)
argument-token (1, "setaudit:machine", terminal ID)
argument-token (1, "setaudit:as_success", preselection mask)
argument-token (1, "setaudit:as_failure", preselection mask)
argument-token (1, "setaudit:asid", audit session ID)
[*priv-token*] (if privilege used or required)
subject-token
return-token

Format (invalid program stack address):
header-token
subject-token
return-token

TABLE B–139 `setaudit_addr`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETAUDIT_ADDR`	`266`	`aa`	0x00000800

Format:
header-token
argument-token (1, "auid", audit user ID)
argument-token (1, "port", terminal ID)
argument-token (1, "type", machine address type)
argument-token (1, "as_success", preselection mask)
argument-token (1, "as_failure", preselection mask)
argument-token (1, "asid", audit session ID)
subject-token
return-token

TABLE B–140 `setauid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETAUID`	`131`	`aa`	0x00040000

TABLE B–140 `setauid`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (2, "setauid", audit user ID)
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–141 `setclearance`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETCLEARANCE`	`542`	`fm`	0x00000008

Format:
header-token
clearance-token (specified)
clearance-token (old)
clearance-token (new)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–142 `setcmwlabel`(2), `lsetcmwlabel`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETCMWLABEL`	`549`	`fm`	0x00000008
`AUE_LSETCMWLABEL`	`525`	`fm`	0x00000008

Format:
header-token
argument-token (3, “flag”, which parts of label to set)
[*slabel-token*] (if slabel is being set)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–143 `setcmwplabel`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETCMWPLABEL`	`541`	`fm`	0x00000008

TABLE B–143 `setcmwplabel`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format (setting flag == SETCL_ALL): *header-token* *slabel-token* (SL from input argument) *slabel-token* (original SL) *argument-token* (2, “flag”, value) *slabel-token* (new SL) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (setting flag == SETCL_SL): *header-token* *slabel-token* (SL from input argument) *slabel-token* (SL of subject before) *argument-token* (2, “flag”, value) *slabel-token* (SL of subject after) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* Format (setting flag == SETCL_IL): *header-token* *argument-token* (2, “flag”, value) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–144 `setegid`(2), old `setgid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETEGID`	`214`	`pm`	0x00200000
`AUE_SETGID`	`205`	`pm`	0x00200000
Format: *header-token* *argument-token* (1, "gid", group ID) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–145 `seteuid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETEUID`	`215`	`pm`	0x00200000

Format:
header-token
argument-token (1, "gid", user ID)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–146 `setfattrflag`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETFATTRFLAG`	`522`	`fm`	0x00000008

Format:
header-token
argument-token (2, "which", which flags to set)
argument-token (3, "attrs", flag values)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–147 `setfpriv`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETFILEPRIV`	`550`	`fm`	0x00000008

Format:
header-token
argument-token (4, "privilege type", privilege set type)
privilege-token
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–148 `setgroups`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETGROUPS`	`26`	`pm`	0x00200000

TABLE B–148 `setgroups`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* [*argument-token*] (1, "setgroups", group ID) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* One *argument-token* for each group set.			

TABLE B–149 `setpattr`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETPATTR`	`526`	`ps`	0x00100000
Format: *header-token* *argument-token* (1, "type", type of attribute to set) *argument-token* (2, "value", value of attribute) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–150 `setpgrp`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETPGRP`	`27`	`pm`	0x00200000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–151 `setppriv`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETPROCPRIV`	`127`	`fm`	0x00000008

TABLE B–151 `setppriv`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (3, “type”, privilege set type) *argument-token* (4, “op”, operation to perform) *privilege-token* (specified) *privilege-token* (old) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–152 `setregid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETREGID`	`41`	`pm`	0x00200000
Format: *header-token* *argument-token* (1, "rgid", real group ID) *argument-token* (1, "egid", effective group ID) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–153 `setreuid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETREUID`	`40`	`pm`	0x00200000
Format: *header-token* *argument-token* (1, "ruid", real user ID) *argument-token* (1, "euid", effective user ID) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–154 `setrlimit`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SETRLIMIT`	`51`	`as`	0x00020000

TABLE B–154 `setrlimit`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–155 `setsockopt`(3SOCKET)

Event Name	Event ID	Event Class	Mask
`AUE_SETSOCKOPT`	`35`	`nt`	0x00000100

Format:
header-token
sock-inet-token
argument-token (2, "level", protocol level)
[*argument-token*] (3, "optname", option name)
argument-token (4, "val", option value)
argument-token (5, "optlen", option length)
subject-token
return-token

The `sock_inet` token for a non-socket operation is reported as: *argument-token* (1, "fd", file descriptor)

TABLE B–156 old `setuid`(2)

Event Name	Event ID	Event Class	Mask
`AUE_OSETUID`	`200`	`pm`	0x00200000

Format:
header-token
argument-token (1, "uid", user ID)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–157 `shmat`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SHMAT`	`96`	`ip`	0x00000200

TABLE B–157 shmat(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, "shm ID", shared memory ID) *argument-token* (2, "shm adr", shared mem addr) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and *slabel* tokens are not included if the shared memory segment ID is invalid.			

TABLE B–158 shmctl(2)

Event Name	Event ID	Event Class	Mask
AUE_SHMCTL	91	ip	0x00000200
Format: *header-token* *argument-token* (1, "shmid", shared memory ID) [*ipc-token*] *subject-token* *return-token* The ipc and ipc_perm tokens are not included if the shared memory segment ID is not valid.			

TABLE B–159 shmctl(2) — IPC_RMID command

Event Name	Event ID	Event Class	Mask
AUE_SHMCTL_RMID	92	ip	0x00000200
Format: *header-token* *argument-token* (1, "shm ID", shared memory ID) [*argument-token*] [*ipc_perm-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token* The *ipc*, *ipc_perm*, and *slabel* tokens are not included if the shared memory segment ID is invalid.			

TABLE B–160 `shmctl`(2) — IPC_SET command

Event Name	Event ID	Event Class	Mask
`AUE_SHMCTL_SET`	`93`	`ip`	0x00000200

Format:
header-token
argument-token (1, "shm ID", shared memory ID)
[*argument-token*]
[ipc_perm-token] (of the IPC’s old values)
[*slabel-token*]
[ipc_perm-token] (of the IPC’s new values)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and *slabel* tokens are not included if the shared memory segment ID is invalid.

TABLE B–161 `shmctl`(2) — IPC_STAT command

Event Name	Event ID	Event Class	Mask
`AUE_SHMCTL_STAT`	`94`	`ip`	0x00000200

Format:
header-token
argument-token (1, "shm ID", shared memory ID)
[*argument-token*]
[*ipc_perm-token*]
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

The *ipc*, *ipc_perm*, and *slabel* tokens are not included if the shared memory segment ID is invalid.

TABLE B–162 `shmdt`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SHMDT`	`97`	`ip`	0x00000200

TABLE B–162 `shmdt`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask

Format:
header-token
argument-token (1, "shm adr", shared mem addr)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–163 `shmget`(2)

Event Name	Event ID	EventClass	Mask
`AUE_SHMGET`	`95`	`ip`	0x00000200

Format:
header-token
argument-token (1, "shm ID", shared memory ID)
argument-token (3, "shm flag", shared memory flags)
[*argument-token*]
[*slabel-token*]
[*ipc_perm-token*] (of the IPC's old values)
[*slabel-token*]
[*ipc_perm-token*] (of the IPC's new values)
[*slabel-token*]
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
subject-token

The *ipc*, *ipc_perm*, and *slabel* tokens are not included for failed events.

TABLE B–164 `shmgetl`(2)

Event Name	Event ID	EventClass	Mask
`AUE_SHMGETL`	`178`	`ip`	0x00000200

TABLE B–164 `shmgetl`(2) *(Continued)*

Event Name	Event ID	EventClass	Mask
Format: *header-token* *argument-token* (1, "shm ID", shared memory ID) *argument-token* (3, "shm flag", shared memory flags) *slabel-token* [*ipc_perm-token*] (of the IPC’s old values) [*slabel-token*] [*ipc_perm-token*] (of the IPC’s new values) [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *subject-token* The *ipc*, *ipc_perm*, and *slabel* tokens are not included for failed events.			

TABLE B–165 `sockconfig()`

Event Name	Event ID	Event Class	Mask
`AUE_SOCKCONFIG`	`265`	`nt`	0x00000100
Format: *header-token* *argument-token* (1, "domain", socket domain) [*argument-token*] (2, "type", socket type) *argument-token* (3, "protocol", socket protocol) *text-token* *subject-token* *return-token*			

TABLE B–166 `socket`(3SOCKET)

Event Name	Event ID	Event Class	Mask
`AUE_SOCKET`	`183`	`nt`	0x00000100
Format: *header-token* *argument-token* (1, "domain", socket domain) [*argument-token*] (2, "type", socket type) *argument-token* (3, "protocol", socket protocol) *subject-token* *return-token*			

TABLE B–167 `stat`(2), statfs(2), `statvfs`(2)

Event Name	Event ID	Event Class	Mask
`AUE_STAT`	`16`	`fa`	0x00000004
`AUE_STATFS`	`54`	`fa`	0x00000004
`AUE_STATVFS`	`234`	`fa`	0x00000004

Format:
header-token
path-token
[*attr-token*]
[*slabel-token*] (object)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–168 `stime`(2)

Event Name	Event ID	Event Class	Mask
`AUE_STIME`	`201`	`as`	0x00020000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
return-token

TABLE B–169 `symlink`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SYMLINK`	`21`	`fc`	0x00000010

Format:
header-token
text-token (symbolic link string)
path-token
[*attr-token*]
[*slabel-token*] (object)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–170 `sysinfo`(2)

Event Name	Event ID	Event Class	Mask
`AUE_SYSINFO`	`39`	`as`	0x00020000
Format: *header-token* *argument-token* (1, "cmd", command) *text-token* (name) [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–171 `system booted`

Event Name	Event ID	Event Class	Mask
`AUE_SYSTEMBOOT`	`113`	`na`	0x00000400
Format: *header-token* *text-token* ("booting kernel") *return-token*			

TABLE B–172 tnif(2), tnrh(2), tnrhtp(2)

Event Name	Event ID	Event Class	Mask
`AUE_TNIF`	`534`	`nt`	0x00000100
`AUE_TNRH`	`535`		
`AUE_TNRHTP`	`536`		
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–173 `tokmapper`(2)

Event Name	Event ID	Event Class	Mask
`AUE_TOKMAPPER`	`537`	`nt`	0x00000100

TABLE B–173 `tokmapper`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (1, “op”, state) *in_addr-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–174 `uadmin`(2) - system freeze

Event Name	Event ID	Event Class	Mask
`AUE_FREEZE`	`539`	`ss`	0x00010000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–175 `uadmin`(2) - system reboot

Event Name	Event ID	Event Class	Mask
`AUE_REBOOT`	`561`	`ss`	0x00010000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–176 `uadmin`(2) - system remount

Event Name	Event ID	Event Class	Mask
`AUE_REMOUNT`	`540`	`as`	0x00020000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–177 `uadmin`(2) - system shutdown

Event Name	Event ID	Event Class	Mask
`AUE_SHUTDOWN`	`560`	`ss`	0x00010000

Format:
header-token
[*priv-token*] (if privilege used or required)
subject-token
slabel-token
return-token

TABLE B–178 `umount`(2) — old version

Event Name	Event ID	Event Class	Mask
`AUE_UMOUNT`	`12`	`ao`	0x00080000

Format:
header-token
path-token
[*attr-token*]
[*slabel-token*] (object)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–179 `umount`(2)

Event Name	Event ID	Event Class	Mask
`AUE_UMOUNT2`	`268`	`ao`	0x00080000

Format:
header-token
path-token
[*attr-token*]
[*slabel-token*] (object)
[*priv-token*] (if privilege used or required)
subject-token
slabel-token (subject)
return-token

TABLE B–180 `unlink`(2)

Event Name	Event ID	Event Class	Mask
`AUE_UNLINK`	`6`	`fd`	0x00000020

TABLE B–180 `unlink`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–181 old `utime`(2), `utimes`(2)

Event Name	Event ID	Event Class	Mask
`AUE_UTIME`	`202`	`fm`	0x00000008
`AUE_UTIMES`	`49`	`fm`	0x00000008
Format: *header-token* *path-token* [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–182 utssys(2) — fusers

Event Name	Event ID	Event Class	Mask
`AUE_UTSSYS`	`233`	`ao`	0x00080000
Format: *header-token* *path-token* [*attr-token*] [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–183 `vfork`(2)

Event Name	Event ID	Event Class	Mask
`AUE_VFORK`	`25`	`ps`	0x00100000

TABLE B–183 `vfork`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *argument-token* (0, "child PID", pid) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token* The `fork` return values are undefined since the audit record is produced at the point that the child process is spawned.			

TABLE B–184 vtrace(2)

Event Name	Event ID	Event Class	Mask
`AUE_VTRACE`	`36`	`pm`	0x00200000
Format: *header-token* [*priv-token*] (if privilege used or required) *subject-token* *return-token*			

TABLE B–185 `write`(2)

Event Name	Event ID	Event Class	Mask
`AUE_WRITE`	`195`	`no`	0x00000000
Format: *header-token* *slabel-token* (from label specified in syscall args) *path-token*) [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–186 `writel`(2), `pwritel`(2), `writevl`(2)

Event Name	Event ID	Event Class	Mask
`AUE_PWRITEL`	`528`	`no`	0x00000000
`AUE_WRITEL`	`552`	`fm`	0x00000008
`AUE_WRITEVL`	`553`	`fm`	0x00000008

TABLE B–186 `writel`(2), `pwritel`(2), `writevl`(2) *(Continued)*

Event Name	Event ID	Event Class	Mask
Format: *header-token* *slabel-token* (from label specified in syscall args) *path-token*) [*attr-token*] [*slabel-token*] [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* *return-token*			

TABLE B–187 xmknod(2)

Event Name	Event ID	Event Class	Mask
`AUE_XMKNOD`	`240`	`fc`	0x00000010
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

TABLE B–188 xstat(2)

Event Name	Event ID	Event Class	Mask
`AUE_XSTAT`	`235`	`fa`	0x00000004
Format: *header-token* *path-token* [*attr-token*] [*slabel-token*] (object) [*priv-token*] (if privilege used or required) *subject-token* *slabel-token* (subject) *return-token*			

Kernel-Level Pseudo-Events

Pseudo-events do have their own audit record structure. They create audit records for the event that uses privilege. When the pseudo-event AUE_UPRIV is in a class that is being audited, any use of privilege will be audited, including uses of privilege for events that are otherwise not being audited.

TABLE B–189 Use of privilege

Event Name	Event ID	Event Class	Mask
`AUE_UPRIV`	`521`	`no`	0x00000000

X Server Protocol Audit Records

These audit records are created by X windows calls and use of the X server. The records are sorted alphabetically by protocol; where possible, records with identical structure are listed together. The description of each record includes:

- The name of the protocol
- The audit event number
- The audit event name
- The audit record structure

TABLE B–190 XClientConnect

Event Name	Message	Event ID	Event Class	Mask
`AUE_ClientConnect`	Client connection to Xserver	`9101`	`xl`	0x08000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *xclient-token* *inaddr-token* (IP address of client) *iport-token* (port on server) *return-token*				

TABLE B–191 XClientDisconnect

Event Name	Message	Event ID	Event Class	Mask
`AUE_ClientDisconnect`	Client logout from Xserver	`9102`	`xl`	0x08000000

TABLE B–191 XClientDisconnect *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *xclient-token* *return-token*				

TABLE B–192 X Server Protocols - window operations

Event Name	Message	Event ID	Event Class	Mask
`AUE_ChangeSaveSet`	Change the saved set	`9108`	`xp`	0x10000000
`AUE_ChangeWindowAttributes`	Change window attributes	`9104`		
`AUE_CirculateWindow`	Circulate the window	`9115`		
`AUE_ConfigureWindow`	Configure the window	`9114`		
`AUE_CreateWindow`	Create window	`9103`		
`AUE_DestroySubwindows`	Destroy subwindows	`9107`		
`AUE_DestroyWindow`	Destroy window	`9106`		
`AUE_GetGeometry`	Get window geometry	`9116`		
`AUE_GetWindowAttributes`	Get window attributes	`9105`		
`AUE_MapSubwindows`	Map the subwindows	`9111`		
`AUE_MapWindow`	Map the window	`9110`		
`AUE_QueryTree`	Query window tree	`9117`		
`AUE_ReparentWindow`	Reparent the window	`9109`		
`AUE_UnmapSubwindows`	Unmap the subwindows	`9113`		
`AUE_UnmapWindow`	Unmap the window	`9112`		

TABLE B–192 X Server Protocols - window operations *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xwindow-token* *return-token*				

TABLE B–193 X Server Protocols - window properties

Event Name	Message	Event ID	Event Class	Mask
`AUE_ChangeProperty`	Change window property	`9120`	`xc`	0x20000000
`AUE_DeleteProperty`	Delete window property	`9121`	`xc`	0x20000000
`AUE_GetProperty`	Get window property	`9122`	`xp`	0x10000000
`AUE_ListProperties`	List window properties	`9123`	`xp`	0x10000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xwindow-token* *xproperty-token* *return-token*				

TABLE B–194 XGetAtomName, XInternAtom

Event Name	Message	Event ID	Event Class	Mask
`AUE_GetAtomName`	Get atom name	`9119`	`xs`	0x80000000
`AUE_InternAtom`	Fetch atom	`9118`	`xs`	0x80000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xatom-token* (atom string) *return-token*				

TABLE B–195 XConvertSelection, XGetSelectionOwner, XSetSelectionOwner

Event Name	Message	Event ID	Event Class	Mask
`AUE_ConvertSelection`	Convert selection	`9126`	`xs`	0x80000000
`AUE_GetSelectionOwner`	Get selection owner	`9125`	`xs`	0x80000000
`AUE_SetSelectionOwner`	Set selection owner	`9124`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xclient-token
return-token

TABLE B–196 XGrabButton

Event Name	Message	Event ID	Event Class	Mask
`AUE_GrabButton`	Grab window button	`9130`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token (grabbing window id)
[xwindow-token] (current device focus)
xcursor-token
return-token

TABLE B–197 XGrabPointer, XUngrabPointer, XUngrabButton

Event Name	Message	Event ID	Event Class	Mask
`AUE_GrabPointer`	Grab pointer	`9128`	`xs`	0x80000000
`AUE_UngrabButton`	Release window button	`9131`	`xs`	0x80000000
`AUE_UngrabPointer`	Release pointer	`9129`	`xs`	0x80000000

TABLE B–197 XGrabPointer, XUngrabPointer, XUngrabButton *(Continued)*

Event Name	Message	Event ID	Event Class	Mask

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token (grabbing window id)
[xwindow-token] (current device focus)
xcursor-token
return-token

TABLE B–198 XChangeActivePointerGrab

Event Name	Message	Event ID	Event Class	Mask
`AUE_ChangeActivePointerGrab`	Change active pointer grab	`9132`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xcursor-token
return-token

TABLE B–199 XGrabKey, XUngrabKeyboard

Event Name	Message	Event ID	Event Class	Mask
`AUE_GrabKey`	Grab key	`9135`	`xs`	0x80000000
`AUE_UngrabKeyboard`	Release keyboard	`9134`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–200 XGrabKeyboard, XUngrabKey

Event Name	Message	Event ID	Event Class	Mask
`AUE_GrabKeyboard`	Grab keyboard	`9133`	`xp`	0x10000000
`AUE_UngrabKey`	Release key	`9135`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–201 XGrabServer, XUngrabServer

Event Name	Message	Event ID	Event Class	Mask
`AUE_GrabServer`	Grab the server	`9137`	`xa`	0x40000000
`AUE_UngrabServer`	Release the server	`9138`	`xa`	0x40000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xclient-token
return-token

TABLE B–202 XQueryPointer

Event Name	Message	Event ID	Event Class	Mask
`AUE_QueryPointer`	Query pointer	`9139`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token (querying window id)
[xwindow-token] (pointer’s window id)
return-token

TABLE B–203 XGetMotionEvents, XSendEvent

Event Name	Message	Event ID	Event Class	Mask
`AUE_GetMotionEvents`	Get motion events	`9140`	`xp`	0x10000000
`AUE_SendEvent`	Send window event	`9127`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–204 XTranslateCoords, XWarpPointer

Event Name	Message	Event ID	Event Class	Mask
`AUE_TranslateCoords`	Translate coordinates	`9141`	`xp`	0x10000000
`AUE_WarpPointer`	Warp the pointer	`9142`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token (source window id)
[xwindow-token] (destination window id)
return-token

TABLE B–205 XGetInputFocus, XSetInputFocus

Event Name	Message	Event ID	Event Class	Mask
`AUE_GetInputFocus`	Get input focus	`9144`	`xs`	0x80000000
`AUE_SetInputFocus`	Set input focus	`9143`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–206 XQueryKeymap

Event Name	Message	Event ID	Event Class	Mask
`AUE_QueryKeymap`	Query keymap	`9145`	`xp`	0x10000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

TABLE B–207 XSetFontPath

Event Name	Message	Event ID	Event Class	Mask
`AUE_SetFontPath`	Set font path	`9146`	`xa`	0x40000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *[xwindow-token]* *xfont-token* *return-token*				

TABLE B–208 XChangeGC

Event Name	Message	Event ID	Event Class	Mask
`AUE_ChangeGC`	Change graphical context	`9148`	`xp`	0x10000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xfont-token* *xpixmap-token* *xgc-token* *return-token*				

TABLE B–209 XCopyGC

Event Name	Message	Event ID	Event Class	Mask
`AUE_CopyGC`	Copy graphical context	`9149`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xgc-token (source gc ID)
[xgc-token] (destination gc ID)
return-token

TABLE B–210 XFreeGC, XSetClipRectangles, XSetDashes

Event Name	Message	Event ID	Event Class	Mask
`AUE_FreeGC`	Free graphical context	`9152`	`xc`	0x20000000
`AUE_SetClipRectangles`	Set clip rectangles	`9151`	`xp`	0x10000000
`AUE_SetDashes`	Set dashes	`9150`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
[xpixmap-token]
[xfont-token]
[xgc-token]
return-token

TABLE B–211 XClearArea

Event Name	Message	Event ID	Event Class	Mask
`AUE_ClearArea`	Clear area	`9153`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–212 XCopyArea, XCopyPlane

Event Name	Message	Event ID	Event Class	Mask
`AUE_CopyArea`	Copy area	`9154`	`xs`	0x80000000
`AUE_CopyPlane`	Copy plane	`9155`	`xs`	0x80000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xpixmap-token (source pixmap ID)
xpixmap-token (destination pixmap ID)
xgc-token
return-token

TABLE B–213 XFillPolygon, XPolyArc, XPolyFillArc, XPolyFillRectangle, XPolyLine, XPolyPoint, XPolyRectangle, XPolySegment

Event Name	Message	Event ID	Event Class	Mask
`AUE_FillPolygon`	Fill polygon	`9161`	`xp`	0x10000000
`AUE_PolyArc`	Polyarc	`9160`	`xp`	0x10000000
`AUE_PolyFillArc`	Fill polyarc	`9163`	`xp`	0x10000000
`AUE_PolyFillRectangle`	Fill polyrectangle	`9162`	`xp`	0x10000000
`AUE_PolyLine`	Polyline	`9157`	`xp`	0x10000000
`AUE_PolyPoint`	Polypoint	`9156`	`xp`	0x10000000
`AUE_PolyRectangle`	Polyrectangle	`9159`	`xs`	0x80000000
`AUE_PolySegment`	Polysegment	`9158`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
xpixmap-token
xgc-token
return-token

TABLE B–214 XGetImage, XImageText8, XImageText16, XPolyText8, XPolyText16, XPutImage

Event Name	Message	Event ID	Event Class	Mask
`AUE_GetImage`	Get image	`9165`	`xs`	0x80000000
`AUE_ImageText8`	Imagetext (8-bit)	`9168`	`xp`	0x10000000
`AUE_ImageText16`	Imagetext (16-bit)	`9169`	`xp`	0x10000000
`AUE_PolyText8`	Polytext (8-bit)	`9166`	`xp`	0x10000000
`AUE_PolyText16`	Polytext (16-bit)	`9167`	`xp`	0x10000000
`AUE_PutImage`	Put image	`9164`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
xpixmap-token
xgc-token
return-token

TABLE B–215 XCreateColormap

Event Name	Message	Event ID	Event Class	Mask
`AUE_CreateColormap`	Create colormap	`9170`	`xc`	0x20000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
return-token

TABLE B–216 XAllocColor, XAllocColorCells, XAllocColorPlanes, XAllocNamedColor, XFreeColors

Event Name	Message	Event ID	Event Class	Mask
`AUE_AllocColor`	Allocate color	`9176`	`xc`	0x20000000
`AUE_AllocColorCells`	Allocate color cells	`9178`		
`AUE_AllocColorPlanes`	Allocate color planes	`9179`		

TABLE B–216 XAllocColor, XAllocColorCells, XAllocColorPlanes, XAllocNamedColor, XFreeColors *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
`AUE_AllocNamedColor`	Allocate named color	`9177`		
`AUE_FreeColors`	Free colors	`9180`		
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xcolormap-token* *return-token*				

TABLE B–217 XCopyColormapAndFree, XFreeColormap, XInstallColormap, XListInstalledColormap, XUninstallColormap

Event Name	Message	Event ID	Event Class	Mask
`AUE_CopyColormapAndFree`	Copy and free colormap	`9172`	`xp`	0x10000000
`AUE_FreeColormap`	Free colormap	`9171`	`xp`	0x10000000
`AUE_InstallColormap`	Install colormap	`9173`	`xa`	0x40000000
`AUE_ListInstalledColormap`	List installed colormap	`9175`	`xs`	0x80000000
`AUE_UninstallColormap`	Uninstall colormap	`9174`	`xp`	0x10000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xcolormap-token* *return-token*				

TABLE B–218 XLookupColor, XQueryColors, XStoreColors, XStoreNamedColor

Event Name	Message	Event ID	Event Class	Mask
`AUE_LookupColor`	Look up colors	`9184`	`xp`	0x10000000
`AUE_QueryColors`	Query colors	`9183`	`xp`	0x10000000
`AUE_StoreColors`	Store colors	`9181`	`xp`	0x10000000

TABLE B–218 XLookupColor, XQueryColors, XStoreColors, XStoreNamedColor *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
`AUE_StoreNamedColor`	Store named colors	`9182`	`xp`	0x10000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xcolormap-token* *return-token*				

TABLE B–219 XCreateCursor

Event Name	Message	Event ID	Event Class	Mask
`AUE_CreateCursor`	Create cursor	`9185`	`xc`	0x20000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xpixmap-token* (source pixmap ID) *xpixmap-token* (mask pixmap ID) *xcursor-token* *return-token*				

TABLE B–220 XCreateGlyphCursor

Event Name	Message	Event ID	Event Class	Mask
`AUE_CreateGlyphCursor`	Create glyph cursor	`9186`	`xc`	0x20000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xfont-token* (source font ID) *xfont-token* (mask font ID) *xcursor-token* *return-token*				

TABLE B–221 XFreeCursor, XRecolorCursor

Event Name	Message	Event ID	Event Class	Mask
AUE_FreeCursor	Free cursor	9187	xc	0x20000000
AUE_RecolorCursor	Recolor cursor	9188	xp	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xcursor-token
return-token

TABLE B–222 XFreePixmap

Event Name	Message	Event ID	Event Class	Mask
AUE_FreePixmap	Free pixmap	9147	xc	0x20000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xpixmap-token
return-token

TABLE B–223 XBell, XChangeKeyboardControl, XChangeKeyboardMapping, XChangePointerControl

Event Name	Message	Event ID	Event Class	Mask
AUE_Bell	Bell	9193	xs	0x80000000
AUE_ChangeKeyboardControl	Change keyboard control	9190		
AUE_ChangeKeyboardMapping	Change keyboard mapping	9189		
AUE_ChangePointerControl	Change pointer control	9192		

TABLE B–223 XBell, XChangeKeyboardControl, XChangeKeyboardMapping, XChangePointerControl *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

TABLE B–224 XForceScreenSaver, XSetScreenSaver

Event Name	Message	Event ID	Event Class	Mask
`AUE_ForceScreenSaver`	Cover screen	`9199`	`xp`	0x10000000
`AUE_SetScreenSaver`	Set screensaver	`9193`		
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

TABLE B–225 XSetCloseDownMode

Event Name	Message	Event ID	Event Class	Mask
`AUE_SetCloseDownMode`	Set closedown mode	`9196`	`xs`	0x80000000
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

TABLE B–226 XChangeHosts, XKillClient, XSetAccessControl

Event Name	Message	Event ID	Event Class	Mask
`AUE_ChangeHosts`	Change hosts	`9194`	`xa`	0x40000000
`AUE_KillClient`	Kill client	`9197`	`xc`	0x20000000
`AUE_SetAccessControl`	Set access control	`9195`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xclient-token
return-token

TABLE B–227 XRotateProperties

Event Name	Message	Event ID	Event Class	Mask
`AUE_RotateProperties`	Rotate properties	`9198`	`xp`	0x10000000

Format:
header-token
subject-token
newgroups-token
slabel-token
[priv-token] (if privilege used or required)
xwindow-token
xproperty-token
return-token

TABLE B–228 XSetModifierMapping, XSetPointerMapping

Event Name	Message	Event ID	Event Class	Mask
`AUE_SetModifierMapping`	Set modifier mapping	`9201`	`xs`	0x80000000
`AUE_SetPointerMapping`	Set pointer mapping	`9200`	`xs`	0x80000000

TABLE B–228 XSetModifierMapping, XSetPointerMapping *(Continued)*

Event Name	Message	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

TABLE B–229 X Server Extensions

Event Name	Message	Event ID	Event Class	Mask
`AUE_XExtensions`	X extension protocols	`9202`	`xp`	
Format: *header-token* *subject-token* *newgroups-token* *slabel-token* *[priv-token]* (if privilege used or required) *xclient-token* *return-token*				

The `AUE_XExtensions` audit record format is used when auditing extensions to the X11 library, such as `XTSOLMakeTPWindow`.

User-Level Generated Audit Records

These audit records are created by programs that operate outside the kernel. The records are sorted alphabetically by program. The description of each record includes:

- The name of the program
- A man page reference (if appropriate)
- The audit event number
- The audit event name
- The audit record structure

TABLE B–230 `add_drv`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_add_drv`	/usr/sbin/add_drv	`9018`	`as`	0x00020000

TABLE B–230 `add_drv`(1M) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *text-token* (driver name) *text-token* (base directory) *text-token* (class name) *text-token* (aliases)				

TABLE B–231 Admin Editor Action - Modify System Files

Event Name	Program	Event ID	Event Class	Mask
`AUE_te_modsysfiles`	trusted editor	`9322`	`ao`	0x00080000
Format: *header-token* *path-token* (filename) *text-token* (changes) *host-token* *return-token* *subject-token* *slabel-token*				

TABLE B–232 `allocate`(1) - device success

Event Name	Program	Event ID	Event Class	Mask
`AUE_allocate_succ`	/usr/sbin/allocate	`6200`	`ao`	0x00080000
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–233 `allocate`(1) - device failure

Event Name	Program	Event ID	Event Class	Mask
`AUE_allocate_fail`	/usr/sbin/allocate	`6201`	`ao`	0x00080000

TABLE B–233 `allocate`(1) - device failure *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–234 `allocate`(1) - list devices success

Event Name	Program	Event ID	Event Class	Mask
`AUE_listdevice_succ`	/usr/sbin/allocate	`6205`	`ao`	0x00080000
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–235 `allocate`(1) - list devices failure

Event Name	Program	Event ID	Event Class	Mask
`AUE_listdevice_fail`	/usr/sbin/allocate	`6206`	`ao`	0x00080000
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–236 `at`(1) - create atjob

Event Name	Program	Event ID	Event Class	Mask
`AUE_at_create`	/usr/bin/at	`6144`	`ao`	0x00080000
Format: *header-token* *subject-token* *return-token* *exec_args-token* *text-token* (user name) *text-token* (job queue)				

TABLE B–237 `at`(1) - delete atjob file (at or atrm)

Event Name	Program	Event ID	Event Class	Mask
`AUE_at_delete`	/usr/bin/at /usr/bin/atrm	`6145`	`ao`	0x00080000

Format:
header-token
subject-token
return-token
exec_args-token
text-token (user name)
text-token (job queue)

TABLE B–238 `at`(1) - permission

Event Name	Program	Event ID	Event Class	Mask
`AUE_at_perm`	/usr/bin/at	`6146`	`ao`	0x00080000

Format:
header-token
subject-token
[group-token]
exit-token

TABLE B–239 `auditd`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_audit`	/usr/sbin/audit	`9016`	`aa`	0x00040000

Format:
header-token
text-token ("new audit file" | "reread audit_control" |
"terminate auditd" | "unknown option")
return-token
subject-token
slabel-token

TABLE B–240 `auditwrite`(3TSOL)

Event Name	Program	Event ID	Event Class	Mask
`AUE_auditwrite`	`auditwrite()`	`9015`	`aa`	0x00040000

Format:
header-token
text-token (error description)
subject-token
return-token

TABLE B–241 `automountd`(1M) – mismatch

Event Name	Program	Event ID	Event Class	Mask
`AUE_automountd_mismatch`	/usr/lib/fs/autofs/automount	`9034`	`ao`	0x00080000

Format:
header-token
path-token (mount dir)
slabel-token (auto* file slabel)
slabel-token (remote host template slabel)
text-token (remote host server)
return-token

TABLE B–242 `automountd`(1M) – mount

Event Name	Program	Event ID	Event Class	Mask
`AUE_automountd_mount`	/usr/lib/fs/autofs/automount	`9033`	`ao`	0x00080000

Format:
header-token
subject-token
slabel-token (subject slabel)
path-token (mount dir)
return-token
host-token (machine name)

TABLE B–243 `chroot`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_chroot`	/usr/sbin/chroot	`9029`	`ao`	0x00080000

Format:
header-token
subject-token
groups-token
slabel-token
return-token
exec_args-token (command-line arguments)
path-token (new root directory)
path-token (command to execute)

TABLE B–244 `crontab`(1) - crontab created

Event Name	Program	Event ID	Event Class	Mask
`AUE_crontab_create`	/usr/bin/crontab	`6148`	`ao`	0x00080000

TABLE B–244 `crontab`(1) - crontab created *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *return-token* *exec_args-token* *text-token* (user name)				

TABLE B–245 `crontab`(1) - crontab deleted

Event Name	Program	Event ID	Event Class	Mask
`AUE_crontab_delete`	/usr/bin/crontab	`6149`	`ao`	0x00080000
Format: *header-token* *subject-token* *return-token* *exec_args-token* *text-token* (user name)				

TABLE B–246 `crontab`(1) - invoke atjob or crontab

Event Name	Program	Event ID	Event Class	Mask
`AUE_cron_invoke`	/usr/bin/crontab	`6147`	`ao`	0x00080000
Format: *header-token* *subject-token* *return-token* *exec_args-token* *text-token* (user name) *text-token* (one of: at-job; batch-job, crontab-job, queue-job #; or unknown job type #) *text-token* (cron command or at job name)				

TABLE B–247 `crontab`(1) – modify

Event Name	Program	Event ID	Event Class	Mask
`AUE_crontab_mod`	`/usr/bin/crontab`	`6170`	`ad`	0x00000800

TABLE B–247 `crontab`(1) – modify *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* [*group-token*] *exit-token*				

TABLE B–248 `crontab`(1) - permission

Event Name	Program	Event ID	Event Class	Mask
`AUE_crontab_perm`	/usr/bin/crontab	`6150`	`ao`	0x00080000
Format: *header-token* *subject-token* *[group-token]* *exit-token*				

TABLE B–249 dbmgr (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_dm_add`		`9319`	`ao`	0x00080000
`AUE_dm_del`		`9320`		
`AUE_dm_mod`		`9321`		
Format: *header-token* *text-token* (database info) *text-token* (database type) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–250 `deallocate`(1) - device success

Event Name	Program	Event ID	Event Class	Mask
`AUE_deallocate_succ`	/usr/sbin/deallocate	`6202`	`ao`	0x00080000

TABLE B–250 `deallocate`(1) - device success *(Continued)*

			Event	
Event Name	**Program**	**Event ID**	**Class**	**Mask**
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–251 `deallocate`(1) — device failure

			Event	
Event Name	**Program**	**Event ID**	**Class**	**Mask**
`AUE_deallocate_fail`	/usr/sbin/deallocate	`6203`	`ao`	0x00080000
Format: *header-token* *subject-token* [*slabel-token*] (subject) *newgroups-token* *exit-token*				

TABLE B–252 `dispadmin`(1M)

Event Name	**Program**	**Event ID**	**Event Class**	**Mask**
`AUE_dispadmin`	/usr/sbin/dispadmin	`9025`	`as`	0x00020000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *text-token* (scheduler class) *path-token* (input file)				

TABLE B–253 dtfile(1) - copy and move

Event Name	**Program**	**Event ID**	**Event Class**	**Mask**
`AUE_dtfile_copy`	/usr/dt/bin/dtfile	`9037`	`fm`	0x00000008
`AUE_dtfile_move`		`9038`		

TABLE B–253 dtfile(1) - copy and move *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *return-token* *path-token* (target path) *slabel-token* (slabel of target) *path-token* (source path) *slabel-token* (slabel of source) *host-token*				

TABLE B–254 `eeprom`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_eeprom`	/usr/sbin/eeprom	`9032`	`as`	0x00020000
Format: *header-token* *return-token* *path-token* (prom device) *text-token* (variable=old value) *text-token* (variable=new value)				

TABLE B–255 `fuser`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_fuser`	/usr/sbin/fuser	`9031`	`ao`	0x00080000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *path-token* (file name) *arg-token* (1, “PID”, process-id)				

TABLE B–256 `groupmgr` (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_gm_add_grp`		`9307`	`ao`	0x00080000
`AUE_gm_del_grp`		`9308`	`ao`	0x00080000
`AUE_gm_mod_grp`		`9309`	`ao`	0x00080000

TABLE B–256 groupmgr (Obsolete) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *text-token* (group info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–257 halt(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_halt_solaris	/usr/sbin/halt	6160	ss	0x00010000
Format: *header-token* *subject-token* *slabel-token* *return-token*				

TABLE B–258 hostmgr (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
AUE_hm_add_host		9310	ao	0x00080000
AUE_hm_del_host		9311		
AUE_hm_mod_host		9312		
AUE_hm_set_def		9313		
Format: *header-token* *text-token* (host info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–259 inetd(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_inetd_connect	/usr/sbin/inetd	6151	na	0x00000400

TABLE B–259 `inetd`(1M) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *text-token* (service name) *ip-address-token* *ip-port-token* *return-token*				

TABLE B–260 `in.ftpd`(1M) - ftp access

Event Name	Program	Event ID	Event Class	Mask
`AUE_ftpd`	/usr/sbin/in.ftpd	`6165`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (error message, failure only) *return-token*				

TABLE B–261 `installf`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_installf`	/usr/sbin/installf	`9042`	`as`	0x00020000
Format: *header-token* *return-token* *argument-token* (package name) *subject-token* *slabel-token*				

TABLE B–262 `login`(1) — local

Event Name	Program	Event ID	Event Class	Mask
`AUE_login`	/usr/bin/login	`6152`	`lo`	0x00001000
Format: *header-token* *text-token* *text-token* (message - success or failure) *subject-token* *return-token*				

TABLE B–263 `login`(1) — rlogin

Event Name	Program	Event ID	Event Class	Mask
`AUE_rlogin`	/usr/bin/login	`6155`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (error message) *return-token*				

TABLE B–264 `login`(1) — telnet

Event Name	Program	Event ID	Event Class	Mask
`AUE_telnet`	/usr/bin/login	`6154`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (error message) *return-token*				

TABLE B–265 `logout`(1)

Event Name	Program	Event ID	Event Class	Mask
`AUE_logout`	/usr/bin/login	`6153`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* *return-token*				

TABLE B–266 `lpadmin`(1M) - authorization

Event Name	Program	Event ID	Event Class	Mask
`AUE_uauth`	/usr/lib/lpadmin	`6196`	`ao`	0x00000800
Format: *header-token* *text-token* (authorization used) *return-token* *text-token* (admin command line) *subject-token* *slabel-token* *host-token*				

TABLE B–267 `lpsched`(1M) - authorization

Event Name	Program	Event ID	Event Class	Mask
`AUE_uauth`	/usr/lib/lpsched	`6196`	`ad`	0x00000800

Format:
header-token
text-token (“ print without banners |
print without labels |print a PostScript file”)
return-token
text-token (hostname/jobnumber-filenumber)
slabel-token (label of print job)
subject-token
slabel-token
host-token

TABLE B–268 `lpsched`(1M) - privilege

Event Name	Program	Event ID	Event Class	Mask
`AUE_lp_cancel`	/usr/lib/lpsched	`9044`	`ao`	0x00080000
`AUE_lp_status`		`9045`		

Format:
header-token
return-token
privilege-token
text-token (hostname/jobnumber-filenumber)
slabel-token (print job label)
subject-token
slabel-token
host-token (error message)

TABLE B–269 `modload`(1M), `modunload`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_modload`	/usr/sbin/modload	`9020`	`as`	0x00020000
`AUE_modunload`	/usr/sbin/modunload	`9021`		

Format:
header-token
subject-token
groups-token
slabel-token
return-token
exec_args-token (command-line arguments)
text-token (module pathname)

TABLE B–270 mountd(1M) – NFS mount

Event Name	Program	Event ID	Event Class	Mask
AUE_mountd_mount	/usr/lib/nfs/mountd	6156	na	0x00000400

Format:
header-token
argument-token
slabel-token (subject slabel)
text-token (remote client hostname)
path-token (mount dir)
slabel-token (slabel of the directory)
text-token (error message, failure only)
attribute-token
subject-token
return-token

TABLE B–271 mountd(1M) – NFS unmount

Event Name	Program	Event ID	Event Class	Mask
AUE_mountd_umount	/usr/lib/nfs/mountd	6157	na	0x00000400

Format:
header-token
slabel-token (subject slabel)
text-token (remote client hostname)
path-token (mount dir)
slabel-token (slabel of the directory)
text-token (error message, failure only)
attribute-token
subject-token
return-token

TABLE B–272 passwd(1)

Event Name	Program	Event ID	Event Class	Mask
AUE_passwd	/usr/bin/passwd	6163	lo	0x00001000

Format:
header-token
subject-token
text-token (error message)
return-token

TABLE B–273 `pfexec`(1)

Event Name	Program	Event ID	Event Class	Mask
`AUE_prof_cmd`	/usr/bin/pfexec	`6180`	`ao`	0x00080000

Format:
header-token
subject-token
slabel-token
clearance-token
path-token (for pfexec)
path-token (for invoking command)
cmd-token
process-token
clearance-token
slabel-token
privilege-token
return-token

TABLE B–274 `pbind`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_pbind`	/usr/sbin/pbind	`9026`	`as`	0x00020000

Format:
header-token
subject-token
groups-token
slabel-token
return-token
exec_args-token (command-line arguments)
text-token (action: “BIND” | “UNBIND”)
arg-token (1, “CPU”, processor id)
arg-token (2, ”PID”, process-id)

TABLE B–275 pfsh — Obsolete

Event Names	Program	Event IDs	Event Class	Mask
`AUE_pfsh_trusted_priv`	/usr/bin/pfsh	9007	`ao`	0x00080000
`AUE_pfsh_trusted_nopriv`		9008		
`AUE_pfsh_priv`		`9009`		
`AUE_pfsh_nopriv`		`9010`	`ap`	0x00004000

TABLE B–275 pfsh — Obsolete *(Continued)*

Event Names	Program	Event IDs	Event Class	Mask

Format:
header-token
path-token (of the executable)
exec_args-token
path-token (of current directory)
privilege-token
return-token
exec_env-token (if AUDIT_ARGE is on)
subject-token
slabel-token

TABLE B–276 pkgadd(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_pkginstall	/usr/sbin/pkgadd	9040	as	0x00020000

Format:
header-token
return-token
argument-token (package name)
subject-token
slabel-token

TABLE B–277 pkgrm(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_pkgremove	/usr/sbin/pkgrm	9041	as	0x00020000

Format:
header-token
return-token
argument-token (package name)
subject-token
slabel-token

TABLE B–278 Print Manager

Event Name	Program	Event ID	Event Class	Mask
AUE_printer_add		6187	ad	0x00000800
AUE_printer_delete		6188		
AUE_printer_delete		6189		

TABLE B–278 Print Manager *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *text-token* (printer info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–279 printmgr (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_pm_add_prn`		`9316`	`ao`	0x00080000
`AUE_pm_del_prn`		`9318`	`ao`	0x00080000
`AUE_pm_mod_prn`		`9317`	`ao`	0x00080000
Format: *header-token* *text-token* (printer info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–280 `profmgr - add profile (Obsolete)`

Event Name	Program	Event ID	Event Class	Mask
`AUE_pm_add_prof`		`9306`	`ao`	0x00080000
Format: *header-token* *text-token* (new profile info) *text-token* (error message) *return-token* *subject-token* *slabel-token* See Table B–303 for the current Rights profile audit records.				

TABLE B–281 `profmgr - delete profile (Obsolete)`

Event Name	Program	Event ID	Event Class	Mask
`AUE_pm_del_prof`		`9304`	`ao`	0x00080000

TABLE B–281 `profmgr - delete profile (Obsolete)` *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *text-token* (profile info) *text-token* (error message) *return-token* *subject-token* *slabel-token* See Table B–303 for the current Rights profile audit records.				

TABLE B–282 `profmgr - modify profile (Obsolete)`

Event Name	Program	Event ID	Event Class	Mask
`AUE_pm_mod_prof`		`9305`	`ao`	0x00080000
Format: *header-token* *text-token* (old profile info) *text-token* (new profile info) *text-token* (error message) *return-token* *subject-token* *slabel-token* See Table B–303 for the current Rights profile audit records.				

TABLE B–283 `psradm`(1m)

Event Name	Program	Event ID	Event Class	Mask
`AUE_psradm`	/usr/sbin/psradm	`9027`	`ps`	0x00100000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *text-token* (action: "ON" | "OFF") *arg-token* (1, "PID", processor id)				

TABLE B–284 `reboot`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_reboot_solaris`	/usr/sbin/reboot	`6161`	`ss`	0x00010000

TABLE B–284 `reboot`(1M) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *return-token*				

TABLE B–285 `removef`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_removef`	/usr/sbin/removef	`9043`	`as`	0x00020000
Format: *header-token* *return-token* *argument-token* (package name) *subject-token* *slabel-token*				

TABLE B–286 `rpc.rexd`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_rexd`	/usr/sbin/rpc.rexd	`6164`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (error message, failure only) *text-token* (hostname) *text-token* (username) *text-token* (command to be executed) *exit-token*				

TABLE B–287 `in.rexecd`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_rexecd`	/usr/sbin/in.rexecd	`6162`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (error message, failure only) *text-token* (hostname) *text-token* (username) *text-token* (command to be executed) *exit-token*				

TABLE B–288 `in.rshd`(1M) - rsh access

Event Name	Program	Event ID	Event Class	Mask
`AUE_rshd`	/usr/sbin/in.rshd	`6158`	`lo`	0x00001000
Format: *header-token* *subject-token* *text-token* (command string) *text-token* (local user) *text-token* (remote user) *return-token*				

TABLE B–289 `rem_drv`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_rem_drv`	/usr/sbin/rem_drv	`9019`	`as`	0x00020000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *text-token* (driver name) [*text-token*] (base directory)				

TABLE B–290 `init`(1M) - run level change

Event Name	Program	Event ID	Event Class	Mask
`AUE_run_level_change`	/usr/sbin/init	`9024`	`ss`	0x00010000
Format: *header-token* *text-token* (new run level) *subject-token* *slabel-token* (if slabel policy on) *return-token*				

TABLE B–291 role login

Event Name	Program	Event ID	Event Class	Mask
`AUE_role_login`		`6173`	`lo`	0x00001000

TABLE B–291 role login *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *slabel-token* (if slabel policy on) *return-token* *host-token*				

TABLE B–292 Selection Manager Transfer

Event Name	Program	Event ID	Event Class	Mask
AUE_sel_mgr_xfer		9039	ax	0x00002000
Format: *header-token* *subject-token* *slabel-token* *return-token*				

TABLE B–293 sendmail(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_sendmail_deliver	/usr/lib/sendmail	9013	ao	0x00080000
AUE_sendmail_defer		9014		
Format: *header-token* *text-token* (message about status) *text-token* (to) *text-token* (message ID) *text-token* (from) *text-token* (from host) *text-token* (to user) *text-token* (to host) *return-token* *slabel-token*				

TABLE B–294 sendmail(1M) - upgrade

Event Name	Program	Event ID	Event Class	Mask
AUE_sendmail_upgrade	/usr/lib/sendmail	9012	ao	0x00080000

TABLE B–294 `sendmail`(1M) - upgrade *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *text-token* (message ID) *slabel-token* (old label) *slabel-token* (new label) *subject-token* *slabel-token*				

TABLE B–295 `serialmgr` (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_sm_del_ser`		`9315`	`ao`	0x00080000
`AUE_sm_mod_ser`		`9314`		
Format: *header-token* *text-token* (port info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–296 `setuname`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_setuname`	/usr/bin/setuname	`9022`	`as`	0x00020000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token*(command-line arguments) *text-token* (action: “ADD” | “DELETE”) *path-token* (swapname)				

TABLE B–297 `share`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_EXPORTFS`	/usr/lib/fs.d/nfs/share	61	`ao`	0x00080000

TABLE B–297 `share`(1M) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask

Format:
header-token
subject-token
slabel-token (subject slabel)
path-token (export directory)
slabel-token (slabel of the directory)
text-token (export options)
return-token

TABLE B–298 `Solaris Management Console - authentication`

Event Name	Program	Event ID	Event Class	Mask
`AUE_admin_authenticate`	SMC — authentication	`6123`	`ao`	0x00080000

Format:
header-token
subject-token
slabel-token
return-token
host-token

TABLE B–299 `Solaris Management Console - Computers and Networks`

Event Name	Program	Event ID	Event Class	Mask
`AUE_network_add`	SMC Computers and Networks	`6184`	`ao`	0x00080000
`AUE_network_delete`		`6185`		
`AUE_network_modify`		`6186`		

Format:
header-token
subject-token
slabel-token
text-token (a file, such as: `hosts`, `tnrhtp`, `tnrhdb`, `networks`, `tnidb`)
text-token (name service)
uauth-token
text-token (attributes in key-value pair format)
return-token
host-token

TABLE B–300 Solaris Management Console - Mounts and Shares

Event Name	Program	Event ID	Event Class	Mask
AUE_filesystem_add	SMC Mounts and Shares	6181	ao	0x00080000
AUE_filesystem_delete		6182		
AUE_filesystem_modify		6183		
Format: *header-token* *subject-token* *slabel-token* *text-token* (SMC object) *text-token* (name service) *uauth-token* *text-token* (attributes in key-value pair format) *return-token* *host-token*				

TABLE B–301 Solaris Management Console - Serial Ports

Event Name	Program	Event ID	Event Class	Mask
AUE_serialport_add	SMC Serial Ports	6193	ao	0x00080000
AUE_serialport_delete		6194		
AUE_serialport_modify		6195		
Format: *header-token* *subject-token* *slabel-token* *text-token* (SMC object) *text-token* (name service) *uauth-token* *text-token* (attributes in key-value pair format) *return-token* *host-token*				

TABLE B–302 Solaris Management Console - Scheduled Jobs

Event Name	Program	Event ID	Event Class	Mask
AUE_scheduledjob_add	SMC Scheduled Jobs	6190	ao	0x00080000
AUE_scheduledjob_delete		6191		
AUE_scheduledjob_modify		6192		

TABLE B–302 `Solaris Management Console - Scheduled Jobs` *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *slabel-token* *text-token* (SMC object) *text-token* (name service) [*uauth-token*] (when required) *text-token* (attributes in key-value pair format) *return-token* *host-token*				

TABLE B–303 `Solaris Management Console - User Accounts and Rights`

Event Name	Program	Event ID	Event Class	Mask
`AUE_usermgr_add`	SMC User Accounts	`6196`	`ad`	0x00000800
`AUE_usermgr_delete`		`6197`		
`AUE_usermgr_modify`		`6198`		
Format: *header-token* *subject-token* *slabel-token* *text-token* (SMC object) [*text-token*] (domain name) *text-token* (name service) *uauth-token* *text-token* (attributes in key-value pair format) *return-token* *host-token* Adding a user generates three records, one for each SMC object.				

TABLE B–304 `Workspace Label Change`

Event Name	Program	Event ID	Event Class	Mask
`AUE_sl_change`		`9035`	`ap`	0x00004000
Format: *header-token* *subject-token* *slabel-token* (original SL) *slabel-token* (new SL) *return-token* *host-token*				

TABLE B–305 su(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_su	/usr/bin/su	6159	lo	0x00001000
Format: *header-token* *subject-token* *text-token* (error message) *return-token*				

TABLE B–306 swap(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_swap	/usr/sbin/swap	9030	as	0x00020000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* *text-token* (new node name | "*none*") *text-token* (new systemname | "*none*")				

TABLE B–307 uadmin(1M)

Event Name	Program	Event ID	Event Class	Mask
AUE_uadmin_cmd	/usr/sbin/uadmin	9023	ss	0x00010000
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *argument-token* (1, "cmd", command code) *argument-token* (2, "fcn", function code)				

TABLE B–308 uauth - Use of Authorization

Event Name	Program	Event ID	Event Class	Mask
AUE_uauth	use of authorization	6199	ao	0x00080000

TABLE B–308 `uauth - Use of Authorization` *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
(See Table B–267 for use of authorization with printing) Format: *header-token* *subject-token* *slabel-token* *uauth-token* *text-token* (SMC object) *return-token* *host-token*				

TABLE B–309 `uautho` (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_uauth`	use of authorization	`9017`	`ao`	0x00080000
Format: *header-token* *text-token* (user name) *text-token* (authorization) *subject-token* *return-token*				

TABLE B–310 `usermgr` (Obsolete)

Event Name	Program	Event ID	Event Class	Mask
`AUE_um_add_user`		`9302`	`ao`	0x00080000
`AUE_um_del_user`		`9301`		
`AUE_um_mod_user`		`9300`		
`AUE_um_set_def`		`9303`		
Format: *header-token* *text-token* (user info) *text-token* (error message) *return-token* *subject-token* *slabel-token*				

TABLE B–311 `uname`(1)

Event Name	Program	Event ID	Event Class	Mask
`AUE_uname_set`	/usr/bin/uname	`9024`	`as`	0x00020000

TABLE B–311 `uname`(1) *(Continued)*

Event Name	Program	Event ID	Event Class	Mask
Format: *header-token* *subject-token* *groups-token* *slabel-token* *return-token* *exec_args-token* (command-line arguments) *text-token* (new node name)				

TABLE B–312 `unshare`(1M)

Event Name	Program	Event ID	Event Class	Mask
`AUE_exportfs`	/usr/lib/fs.d/nfs/share		`na`	0x00000400
Format: *header-token* *subject-token* *slabel-token* (subject slabel) *path-token* (export directory) *return-token*				

APPENDIX **C**

Audit Reference

Auditing brings a number of utilities to the Trusted Solaris operating environment. The utilities are listed here in four tables, that are ordered by man page section number. Each table gives utility names and a short description of the task performed by each utility. The fifth table gives the file system security attributes of files in the auditing subsystem.

TABLE C–1 Section 1M — Maintenance Commands

Command	Task
`audit`(1M)	Control the audit daemon
`audit_startup`(1M)	Initialize the audit subsystem
`audit_warn`(1M)	Run the audit daemon warning script
`auditconfig`(1M)	Configure auditing
`auditd`(1M)	Control audit trail files
`auditreduce`(1M)	Merge and select audit records from audit trail files
`auditstat`(1M)	Display kernel audit statistics
`praudit`(1M)	Print contents of an audit trail file
`/etc/init.d/audit stop`	Halt auditing [a script; see `init.d`(4)]
`/etc/init.d/audit start`	Restart auditing [a script; see `init.d`(4)]

TABLE C–2 Section 2 — System Calls

System Call	System Parameter	Task
`audit`(2)		Write a record to the audit log
`auditon`(2)		Manipulate auditing:

TABLE C–2 Section 2 — System Calls *(Continued)*

System Call	System Parameter	Task
	A_GETPOLICY	Get audit policy flags
	A_SETPOLICY	Set audit policy flags
	A_GETKMASK	Get asynchronous audit event preselection mask
	A_SETKMASK	Set asynchronous audit event preselection mask
	A_GETQCTRL	Get the kernel audit queue control parameters
	A_SETQCTRL	Set the kernel audit queue control parameters
	A_GETSTAT	Get the audit system statistics
	A_SETSTAT	Reset the audit system statistics
	A_GETCOND	Determine if auditing is on/off/disabled
	A_SETCOND	Set auditing to on/off
	A_GETFSIZE	Get the size limit for an audit trail file
	A_GETCLASS	Return the event to class mapping for the designated event
	A_SETCLASS	Set the event to class mapping for the designated audit event
	A_GETPINFO	Get the audit information for the specified process
	A_SETPMASK	Set the preselection mask for a specified process
	A_SETUMASK	Set the process mask for all processes of a specified audit ID
	A_SETSMASK	Set the process mask for all processes of a specified session ID
	A_GETCWD	Get the current working directory for this process
	A_GETCAR	Get the current active root for this process
`auditsvc`(2)		Write audit log to specified file descriptor
`getaudit`(2)		Get process audit information
`setaudit`(2)		Set process audit information
`getauid`(2)		Get user audit identity

TABLE C–2 Section 2 — System Calls *(Continued)*

System Call	System Parameter	Task
setauid(2)		Set user audit identity

TABLE C–3 Section 3 — C Library Functions

Library Call	Task
au_preselect(3BSM)	Preselect an audit event
au_user_mask(3BSM)	Get user's binary preselection mask
getacdir(3BSM), getacmin(3BSM), getacflg(3BSM), getacna(3BSM), setac(3BSM), endac(3BSM)	Get audit_control(4) file information
getauclassnam(3BSM), getauclassnam_r(3BSM), getauclassent(3BSM), getauclassent_r(3BSM), setauclass(3BSM), endauclass(3BSM)	Get audit_class(4) entries
getauditflagsbin(3BSM), getauditflagschar(3BSM)	Convert audit flag specifications
getauevent(3BSM), getauevent_r(3BSM), getauevnam(3BSM), getauevnam_r(3BSM), getauevnum(3BSM), getauevnum_r(3BSM), getauevnonam(3BSM), setauevent(3BSM), endauevent(3BSM)	Get audit_event(4)entries
getauusernam(3BSM), getauuserent(3BSM), setauuser(3BSM), endauuser(3BSM)	Get audit_user(4) entries
getfauditflags(3BSM)	Generate the process audit state

TABLE C–4 Section 4 — Headers, Tables, and Macros

Files	Task
audit.log(4)	Gives format for an audit trail file
audit_class(4)	Gives audit class definitions
audit_control(4)	Controls information for system audit daemon
audit_data(4)	Holds current information on the audit daemon
audit_event(4)	Holds audit event definition and class mapping
audit_user(4)	Holds per-user auditing information

TABLE C–5 File System Security Attributes for the Audit Subsystem

Name	Label	DAC	Owner	Group
audit(1M)	[ADMIN_LOW]	555	bin	bin
auditd(1M)				
auditconfig(1M)				
auditstat(1M)				
auditreduce(1M)				
praudit(1M)				
/etc/init.d/audit*	[ADMIN_LOW]	400	root	sys
audit_warn(1M)	[ADMIN_LOW]	640	root	sys
audit_startup(1M)	[ADMIN_LOW]	750	root	sys
audit.log(4)	[ADMIN_HIGH]	400	root	root
audit_class(4)	[ADMIN_LOW]	400	root	sys
audit_control(4)	[ADMIN_LOW]	400	root	sys
audit_data(4)	[ADMIN_HIGH]	660	root	root
audit_event(4)	[ADMIN_LOW]	400	root	sys
audit_user(4)	[ADMIN_LOW]	400	root	sys

Index