

## **Managing Auditing in Oracle® Solaris 11.3**



**Part No: E54781**  
November 2016



**Part No: E54781**

Copyright © 2002, 2016, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

**Documentation Accessibility**

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

**Référence: E54781**

Copyright © 2002, 2016, Oracle et/ou ses affiliés. Tous droits réservés.

Ce logiciel et la documentation qui l'accompagne sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d'utilisation et de divulgation. Sauf stipulation expresse de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, accorder de licence, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d'interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu'elles soient exemptes d'erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Si ce logiciel, ou la documentation qui l'accompagne, est livré sous licence au Gouvernement des Etats-Unis, ou à quiconque qui aurait souscrit la licence de ce logiciel pour le compte du Gouvernement des Etats-Unis, la notice suivante s'applique:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Ce logiciel ou matériel a été développé pour un usage général dans le cadre d'applications de gestion des informations. Ce logiciel ou matériel n'est pas conçu ni n'est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer des dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d'applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l'utilisation de ce logiciel ou matériel pour ce type d'applications.

Oracle et Java sont des marques déposées d'Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d'autres propriétaires qu'Oracle.

Intel et Intel Xeon sont des marques ou des marques déposées d'Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées de SPARC International, Inc. AMD, Opteron, le logo AMD et le logo AMD Opteron sont des marques ou des marques déposées d'Advanced Micro Devices. UNIX est une marque déposée d'The Open Group.

Ce logiciel ou matériel et la documentation qui l'accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers, sauf mention contraire stipulée dans un contrat entre vous et Oracle. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'accès à des contenus, produits ou services tiers, ou à leur utilisation, sauf mention contraire stipulée dans un contrat entre vous et Oracle.

**Accessibilité de la documentation**

Pour plus d'informations sur l'engagement d'Oracle pour l'accessibilité à la documentation, visitez le site Web Oracle Accessibility Program, à l'adresse <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

**Accès aux services de support Oracle**

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> ou le site <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> si vous êtes malentendant.

# Contents

---

<b>Using This Documentation .....</b>	<b>9</b>
<b>1 About Auditing in Oracle Solaris .....</b>	<b>11</b>
What Is Auditing? .....	11
Audit Terminology and Concepts .....	12
Audit Events .....	14
Audit Classes and Preselection .....	15
Audit Records and Audit Tokens .....	16
Audit Plugin Modules .....	17
Audit Remote Server .....	17
Audit Logs .....	18
Storing and Managing the Audit Trail .....	20
Ensuring Reliable Time Stamps .....	21
Managing a Remote Repository .....	21
How Is Auditing Related to Security? .....	21
How Does Auditing Work? .....	22
How Is Auditing Configured? .....	23
Using Oracle Audit Vault and Database Firewall for Storage and Analysis of Audit Records .....	24
Auditing on a System With Oracle Solaris Zones .....	26
<b>2 Planning for Auditing .....</b>	<b>27</b>
Concepts in Planning Auditing .....	27
Planning a Single System Audit Trail .....	27
Planning Auditing in Zones .....	28
Planning Auditing .....	29
▼ How to Plan Who and What to Audit .....	30
Planning Disk Space for Audit Records .....	32
Preparing to Stream Audit Records to Remote Storage .....	33
Understanding Audit Policy .....	34
Controlling Auditing Costs .....	36

Cost of Increased Processing Time of Audit Data .....	36
Cost of Analysis of Audit Data .....	36
Cost of Storage of Audit Data .....	37
Auditing Efficiently .....	38
<b>3 Managing the Audit Service .....</b>	<b>39</b>
Default Configuration of the Audit Service .....	39
Displaying Audit Service Defaults .....	40
Enabling and Disabling the Audit Service .....	42
Configuring the Audit Service .....	42
▼ How to Preselect Audit Classes .....	44
▼ How to Configure a User's Audit Characteristics .....	45
▼ How to Change Audit Policy .....	48
▼ How to Change Audit Queue Controls .....	51
▼ How to Configure the audit_warn Email Alias .....	53
▼ How to Add an Audit Class .....	54
▼ How to Change an Audit Event's Class Membership .....	55
Customizing What Is Audited .....	56
▼ How to Audit All Commands by Users .....	57
▼ How to Find Audit Records of Changes to Specific Files .....	59
▼ How to Update the Preselection Mask of Logged In Users .....	61
▼ How to Prevent the Auditing of Specific Events .....	62
▼ How to Compress Audit Files on a Dedicated File System .....	63
▼ How to Audit FTP and SFTP File Transfers .....	64
Configuring the Audit Service in Zones .....	65
▼ How to Configure All Zones Identically for Auditing .....	66
▼ How to Configure Per-Zone Auditing .....	68
Example: Configuring Oracle Solaris Auditing .....	69
<b>4 Monitoring System Activities .....</b>	<b>73</b>
Configuring Audit Logs .....	73
Configuring Audit Logs .....	73
▼ How to Create ZFS File Systems for Audit Files .....	74
▼ How to Assign Audit Space for the Audit Trail .....	77
▼ How to Send Audit Files to a Remote Repository .....	80
▼ How to Configure a Remote Repository for Audit Files .....	82
▼ How to Configure syslog Audit Logs .....	86
<b>5 Working With Audit Data .....</b>	<b>89</b>

---

Displaying Audit Trail Data .....	89
Displaying Audit Record Definitions .....	89
Selecting Audit Events to Be Displayed .....	91
Viewing the Contents of Binary Audit Files .....	93
Managing Audit Records on Local Systems .....	97
▼ How to Merge Audit Files From the Audit Trail .....	97
▼ How to Clean Up a not_terminated Audit File .....	99
Preventing Audit Trail Overflow .....	100
<b>6 Analyzing and Resolving Audit Service Issues .....</b>	<b>103</b>
Troubleshooting the Audit Service .....	103
Audit Records Are Not Being Logged .....	104
Volume of Audit Records Is Large .....	106
Binary Audit File Sizes Grow Without Limit .....	108
Logins From Other Operating Systems Not Being Audited .....	109
<b>7 Auditing Reference .....</b>	<b>111</b>
Audit Service .....	111
Audit Service Man Pages .....	112
Rights Profiles for Administering Auditing .....	114
Auditing and Oracle Solaris Zones .....	114
Audit Configuration Files and Packaging .....	115
Audit Classes .....	115
Audit Class Syntax .....	115
Audit Plugins .....	116
Audit Remote Server .....	117
Audit Policy .....	117
Audit Policies for Asynchronous and Synchronous Events .....	118
Process Audit Characteristics .....	119
Audit Trail .....	120
Conventions for Binary Audit File Names .....	120
Audit Record Structure .....	120
Audit Record Analysis .....	121
Audit Token Formats .....	121
acl Token .....	123
argument Token .....	123
attribute Token .....	123
cmd Token .....	124
exec_args Token .....	124

exec_env Token .....	124
file Token .....	125
fmri Token .....	125
group Token .....	125
header Token .....	125
ip_address Token .....	126
ip_port Token .....	126
ipc Token .....	126
IPC_perm Token .....	127
path Token .....	127
path_attr Token .....	128
privilege Token .....	128
process Token .....	128
return Token .....	128
sequence Token .....	129
socket Token .....	129
subject Token .....	129
text Token .....	130
trailer Token .....	130
use of authorization Token .....	130
use of privilege Token .....	131
user Token .....	131
xclient Token .....	131
zonename Token .....	131
<b>Glossary</b> .....	<b>133</b>
<b>Index</b> .....	<b>141</b>



## Using This Documentation

---

*Managing Auditing in Oracle® Solaris 11.3* documents the auditing feature of Oracle Solaris.

- **Overview** – Describes how to administer auditing on an Oracle Solaris system or on a network of systems.
- **Audience** – System administrators who must implement security on the enterprise.
- **Required knowledge** – Familiarity with security concepts and terminology.

## Product Documentation Library

Documentation and resources for this product and related products are available at <http://www.oracle.com/pls/topic/lookup?ctx=E53394>.

## Feedback

Provide feedback about this documentation at <http://www.oracle.com/goto/docfeedback>.



## About Auditing in Oracle Solaris

---

The auditing subsystem of Oracle Solaris keeps a record of how the system is being used. The audit service includes tools to assist with the analysis of the auditing data.

This chapter introduces how auditing works in Oracle Solaris:

- “What Is Auditing?” on page 11
- “Audit Terminology and Concepts” on page 12
- “How Is Auditing Related to Security?” on page 21
- “How Does Auditing Work?” on page 22
- “How Is Auditing Configured?” on page 23
- “Using Oracle Audit Vault and Database Firewall for Storage and Analysis of Audit Records” on page 24
- “Auditing on a System With Oracle Solaris Zones” on page 26

For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For procedures to configure auditing at your site, see the following chapters:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 5, “Working With Audit Data”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

For reference information, see [Chapter 7, “Auditing Reference”](#).

## What Is Auditing?

Auditing is the collecting of data about the use of system resources. The audit data provides a record of security-related system events. This data can then be used to assign responsibility for actions that take place on a system.

Successful auditing starts with identification and authentication. At each login, after a user supplies a user name and PAM (pluggable authentication module) authentication succeeds, a unique and immutable *audit user ID* is generated and associated with the user, and a unique

audit session ID is generated and associated with the user's process. The audit session ID is inherited by every process that is started during that login session. When a user switches to another user, all user actions are tracked with the same audit user ID. For more details about switching identity, see the [su\(1M\)](#) man page. Note that by default, certain actions such as booting and shutting down the system are always audited.

The audit service enables the following operations:

- Monitoring security-relevant events that take place on the system
- Recording the events in a network-wide audit trail
- Detecting misuse or unauthorized activity
- Reviewing patterns of access and the access histories of individuals and objects
- Discovering attempts to bypass the protection mechanisms
- Discovering extended use of privilege that occurs when a user changes identity

---

**Note** - To maintain security, audited events do not include sensitive information such as passwords. For more details, see [“Audit Records and Audit Tokens” on page 16](#).

---

## Audit Terminology and Concepts

The following terms are used to describe the audit service. Some definitions include pointers to more complete descriptions.

audit class	<p>A grouping of audit events. Audit classes provide a way to select a group of events to be audited.</p> <p>For more information, see <a href="#">“Audit Classes and Preselection” on page 15</a>, and the <a href="#">audit_flags(5)</a>, <a href="#">audit_class(4)</a>, and <a href="#">audit_event(4)</a> man pages.</p>
audit file system	<p>A repository of audit files in binary format.</p> <p>For more information, see <a href="#">“Audit Logs” on page 18</a> and the <a href="#">audit.log(4)</a> man page.</p>
audit event	<p>A security-related system action that is auditable. For ease of selection, events are grouped into audit classes.</p> <p>For more information, see <a href="#">“Audit Events” on page 14</a> and the <a href="#">audit_event(4)</a> man page.</p>
audit flag	<p>An audit class that is supplied as an argument to a command or keyword. Audit flags preselect which audit classes are to be audited for a process.</p> <p>For information about using audit flags, see <a href="#">“Audit Class Syntax” on page 115</a>. See, also, the <a href="#">audit_flags(5)</a> man page.</p>

audit plugin	<p>A module that transfers the audit records in the queue to a specified location. The <code>audit_binfile</code> plugin creates binary audit files. Binary files comprise the audit trail, which is stored on audit file systems. The <code>audit_remote</code> plugin sends binary audit records to a remote repository. The <code>audit_syslog</code> plugin summarizes audit records and writes them to the system log using <code>syslog(3C)</code>.</p> <p>For more information, see <a href="#">“Audit Plugin Modules” on page 17</a> and the module man pages, <a href="#">audit_binfile(5)</a>, <a href="#">audit_remote(5)</a>, and <a href="#">audit_syslog(5)</a>.</p>
audit policy	<p>A set of auditing options that you can enable or disable at your site. You can specify policies such as the following:</p> <ul style="list-style-type: none"> <li>■ Whether to record certain kinds of audit data</li> <li>■ How much information to include in the audit content</li> <li>■ How to handle certain types of files</li> <li>■ How to handle a full audit queue</li> </ul> <p>For more information, see <a href="#">“Understanding Audit Policy” on page 34</a> and the <a href="#">auditconfig(1M)</a> man page.</p>
audit record	<p>Audit data that is collected in the audit queue. An audit record describes a single audit event. Each audit record is composed of audit tokens.</p> <p>For more information, see <a href="#">“Audit Records and Audit Tokens” on page 16</a> and the <a href="#">audit.log(4)</a> man page.</p>
audit token	<p>A field of an audit record. Each audit token describes an attribute of an audit record, such as a user, a group, a program, or other object.</p> <p>For more information, see <a href="#">“Audit Token Formats” on page 121</a> and the <a href="#">audit.log(4)</a> man page.</p>
audit trail	<p>A collection of one or more audit files that store the audit data from all audited systems that use the default plugin, <code>audit_binfile</code>.</p> <p>For more information, see <a href="#">“Audit Trail” on page 120</a>.</p>
local auditing	<p>The collecting of audit records that are generated on the local system. The records can be generated in the global zone or in non-global zones, or both.</p> <p>For more information, see <a href="#">“Audit Plugin Modules” on page 17</a>.</p>
post-selection	<p>The choice of which audit events to examine from the preselected audit trail. The default active plugin, <code>audit_binfile</code>, creates the audit trail. A post-selection tool, the <code>auditreduce</code> command, selects records from the audit trail.</p>

	For more information, see the <a href="#">auditreduce(1M)</a> and <a href="#">praudit(1M)</a> man pages.
preselection	The initial choice of which audit classes to monitor. The audit events of preselected audit classes are collected in the audit queue. Audit classes that are not preselected are not audited, so their events do not appear in the queue.
<hr/> <b>Note</b> - Only audit events that are preselected are available for further post-selection review using the <code>auditreduce</code> command.	
	For more information, see “ <a href="#">Audit Classes and Preselection</a> ” on page 15 and the <a href="#">audit_flags(5)</a> and <a href="#">auditconfig(1M)</a> man pages.
public object	A file that is owned by the root user and readable by the world. For example, some files in the <code>/etc</code> directory and the <code>/usr/bin</code> directory are public objects. Public objects are not audited for read-only events. For example, even if the <code>file_read (fr)</code> audit class is preselected, the reading of public objects is not audited. You can override the default by changing the public audit policy option.
remote auditing	<p>The audit remote server (ARS) that receives and stores audit records from a system that is being audited and is configured with an active <code>audit_remote</code> plugin. To distinguish an audited system from an ARS, the audited system can be referred to as the “locally audited system.”</p> <p>For more information, see the <code>-setremote</code> options on the <a href="#">auditconfig(1M)</a> man page, the <code>ars(5)</code> man page, and “<a href="#">Audit Remote Server</a>” on page 117.</p>

## Audit Events

Audit events represent auditable actions on a system. Audit events are listed in the `/etc/security/audit_event` file. Each audit event is connected to a system call or user command, and is assigned to one or more audit classes. For a description of the format of the `audit_event` file, see the [audit\\_event\(4\)](#) man page.

For example, the `AUE_EXECVE` audit event audits the `execve()` system call. The command `auditrecord -e execve` displays this entry:

```
# auditrecord -e execve
execve
system call execve                See execve(2)
```

event ID	23	AUE_EXECVE
class	ps,ex	(0x0000000080100000)
header		
path		
[attribute]		omitted on error
[exec_arguments]		output if argv policy is set
[exec_environment]		output if arge policy is set
subject		
[use_of_privilege]		
return		

When you preselect either the audit class ps or the audit class ex, then every `execve()` system call is recorded in the audit queue.

Auditing handles *attributable* and *non-attributable* events. Audit policy divides events into *synchronous* and *asynchronous* events, as follows:

- **Attributable events** – Events that can be attributed to a user. The `execve()` system call can be attributed to a user, so the call is considered an attributable event. All attributable events are synchronous events.
- **Non-attributable events** – Events that occur at the kernel-interrupt level or before a user is authenticated. The na audit class handles audit events that are non-attributable. For example, booting the system is a non-attributable event. Most non-attributable events are asynchronous events. However, non-attributable events that have associated processes, such as a failed login, are synchronous events.
- **Synchronous events** – Events that are associated with a process in the system. Synchronous events are the majority of system events. If synchronous events cannot be queued, the process is blocked until they can be queued.
- **Asynchronous events** – Events that are not associated with any process, so no process is available to be blocked and later started. Initial system boot and PROM enter and exit events are examples of asynchronous events.

In addition to the audit events that are defined by the audit service, third-party applications can generate audit events. Audit event numbers from 32768 to 65535 are available for third-party applications. Vendors need to contact their Oracle Solaris representative to reserve event numbers and obtain access to the audit interfaces.

## Audit Classes and Preselection

Each audit event belongs to an *audit class*. Audit classes are convenient containers for large numbers of audit events. When you *preselect* a class to be audited, all the events in that class are recorded in the audit queue. For example, when you preselect the ps audit class, `execve()`, `fork()`, and other system calls are recorded.

You can preselect for events on a system and for events initiated by a particular user.

- **System-wide preselection** – Specify the system-wide defaults for auditing by using the `-setflags` and `-setnaflags` options to the `auditconfig` command.

---

**Note** - If the `perzone` policy is set, default audit classes can be specified in every zone. For `perzone` auditing, the defaults are zone-wide, not system-wide.

---

- **User-specific preselection** – Specify additional audit flags to audit for individual users to be included along with the system-wide auditing defaults. The `useradd`, `roleadd`, `usermod`, and `rolemo` commands place the `audit_flags` security attribute in the `user_attr` database. The `profiles` command places audit flags for rights profiles in the `prof_attr` database.

The audit preselection mask determines which classes of events are audited for a user. For a description of the user preselection mask, see [“Process Audit Characteristics” on page 119](#). For the configured audit flags that are used, see [“Order of Search for Assigned Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

Audit classes are defined in the `/etc/security/audit_class` file. Each entry contains the audit mask for the class, the name for the class, and a descriptive name for the class. For example, the `lo` and `ps` class definitions appear in the `audit_class` file as follows:

```
0x0000000000001000:lo:login or logout
0x0000000000100000:ps:process start/stop
```

The audit classes include the two global classes: `all` and `no`. The audit classes are described in the [`audit\_class\(4\)`](#) man page. For the list of classes, read the `/etc/security/audit_class` file.

The mapping of audit events to classes is configurable. You can remove events from a class, add events to a class, and create a new class to contain selected events. For the procedure, see [“How to Change an Audit Event's Class Membership” on page 55](#). To view the events that are mapped to a class, use the `auditrecord -c class` command.

## Audit Records and Audit Tokens

Each *audit record* records the occurrence of a single audited event. The record includes information such as who did the action, which files were affected, what action was attempted, and where and when the action occurred. The following example shows a `login` audit record with three tokens, header, subject, and return:

```
header,69,2,login - local,,example_system,2010-10-10 10:10:10.020 -07:00
subject,jdoe,jdoe,staff,jdoe,staff,1210,4076076536,69 2 example_system
return,success,0
```

The type of information that is saved for each audit event is defined by a set of *audit tokens*. Each time an audit record is created for an event, the record contains some or all of the tokens that are defined for the event. The nature of the event determines which tokens are recorded. In



the preceding example, each line begins with the name of the audit token. The content of the audit token follows the token name. Together, the header, subject, and return audit tokens comprise the `login - local` audit record. To display the tokens that comprise an audit record, use the `auditrecord -e event` command.

---

**Note** - Files with the sensitive system attribute do not have their contents or content changes included in the audit record. The attribute ensures that no sensitive information in specific files, such as passwords, PINs, keys, and so on, is accessible to anyone. For more details, refer to the [pfedit\(1M\)](#) man page.

---

For a detailed description of the structure of each audit token with an example of `praudit` output, see [“Audit Token Formats” on page 121](#). For a description of the binary stream of audit tokens, see the [audit.log\(4\)](#) man page.

## Audit Plugin Modules

The audit plugin modules direct the audit records from the audit queue to a file or repository. At least one plugin must be active or ARS must be configured. By default, the `audit_binfile` plugin is active. You configure plugins with the `auditconfig -setplugin plugin-name` command.

The audit service provides the following plugins:

- `audit_binfile` plugin – Handles delivery of the audit queue to the binary audit files. For more information, see the [audit.log\(4\)](#) man page.
- `audit_remote` plugin – Handles secure delivery of binary audit records from the audit queue to a configured remote server. The `audit_remote` plugin uses the `libgss()` library to authenticate the server. The transmission is protected for privacy and integrity. For information about ARS, see [“Audit Remote Server” on page 117](#).
- `audit_syslog` plugin – Handles delivery of selected records from the audit queue to the `syslog` logs.

For information about how to configure a plugin, see the [auditconfig\(1M\)](#) man page. For examples of plugin configuration, see the tasks in [“Configuring Audit Logs” on page 73](#). For information about the plugins, see the [audit\\_binfile\(5\)](#), [audit\\_remote\(5\)](#), and [audit\\_syslog\(5\)](#) man pages.

## Audit Remote Server

The audit remote server (ARS) is the counterpart of the `audit_remote` plugin. Data sent by the plugin can be captured, processed, and stored by ARS according to the server configuration.

ARS is delivered as a disabled audit component in Oracle Solaris systems. You must configure ARS before it can be used to process a remote audit trail. To configure ARS:

- Configure the underlying security mechanisms used for secure audit data transport. See the `audit_remote(5)` man page.
- Configure the audit remote subsystem using the `auditconfig -setremote` command. The configuration includes both server configuration and connection group configuration. A connection group is the sets of systems sharing the same local storage parameters. For information and examples, see the `ars(5)` man page and the `auditconfig(1M)` man page.

See, also, [“Audit Remote Server” on page 117](#).

## Audit Logs

Audit records are collected in audit logs. The audit service provides three output modes for audit records.

- Logs that are called *audit files* store audit records in binary format. The set of audit files from a system or site provides a complete audit record. The complete audit record is called the *audit trail*. These logs are created by the `audit_binfile` plugin or the Audit Remote Server, and can be reviewed by the `praudit` and `auditreduce` post-selection commands.
- The `audit_remote` plugin streams audit records to a remote repository. The repository is responsible for maintaining an audit trail and supplying post-selection tools.
- The `syslog` utility collects and stores text summaries of the audit record. A `syslog` record is not complete. The following example shows a `syslog` entry for a login audit record:

```
Feb 5 11:54:57 example_system audit: [ID 702911 audit.notice] \  
login - login ok session 2870512630 by user as user:staff
```

---

**Note** - As an alternative to `syslog`, you can install the `rsyslog` package and use the `rsyslog` utility for remote `syslog` functionality. `Rsyslog` is a reliable and extended `syslog` implementation with a modular design that supports features such as filtering, TCP, encryption, high-precision time-stamps, and output control.

---

A site can configure auditing to collect audit records in all formats. You can configure the systems at your site to use binary mode locally, to send binary files to a remote repository, and to use `syslog` mode. The following table compares binary audit records with `syslog` audit records.

**TABLE 1** Comparison of Binary, Remote, and `syslog` Audit Records

Feature	Binary and Remote Records	<code>syslog</code> Records
Protocol	Binary – Writes to the file system	Uses UDP for remote logging

Feature	Binary and Remote Records	syslog Records
	Remote – Streams to a remote repository	rsyslog uses TCP
Data type	Binary	Text
Record length	No limit	Up to 1024 characters per audit record
Location	Binary – Stored in a zpool on the system	Stored in a location that is specified in the syslog.conf file
	Remote – Remote repository	
How to configure	Binary – Set the p_dir attribute on the audit_binfile plugin	Make the audit_syslog plugin active and configure the syslog.conf file
	Remote – Set the p_hosts attribute on the audit_remote plugin and make the plugin active	
How to read	Binary – Typically, in batch mode, browser output in XML	In real time or searched by scripts that you have created for syslog
	Remote – Repository dictates the procedure	Plain text output
Completeness	Guaranteed to be complete and to appear in the correct order	A summary that is not guaranteed to be complete
Time stamp	Coordinated Universal Time (UTC)	Time on the system that is being audited

For more information about plugins and audit logs, refer to the following:

- [audit\\_binfile\(5\)](#) man page
- [audit\\_syslog\(5\)](#) man page
- [audit.log\(4\)](#) man page
- [ars\(5\)](#)
- “How to Assign Audit Space for the Audit Trail” on page 77
- “How to Configure syslog Audit Logs” on page 86

## About Binary Records

Binary records provide the greatest security and coverage. Binary output meets the requirements of security certifications, such as the [Common Criteria](http://www.commoncriteriaportal.org/) (<http://www.commoncriteriaportal.org/>) audit requirements.

The `audit_binfile` plugin writes the records to a file system that you protect from snooping. On a single system, all binary records are collected and displayed in order. The UTC time stamp on binary logs enables accurate comparison when systems on one audit trail are distributed across time zones. The `praudit` command enables you to view the records in a browser in XML. You can also use scripts to parse the XML output.

The `audit_remote` plugin writes the records to a remote repository. The repository handles storage and post-selection.

The audit remote server also yields binary.

## About syslog Audit Records

In contrast, the syslog or rsyslog records might provide greater convenience and flexibility. For example, you can collect the syslog data from a variety of sources. Also, when you monitor audit.notice events in the syslog.conf file, the syslog utility logs an audit record summary with the current time stamp. You can use the same management and analysis tools that you have developed for syslog messages from a variety of sources, including workstations, network servers, firewalls, and routers. The records can be viewed in real time, and can be stored on a remote system.

By using syslog.conf to store audit records remotely, you protect log data from alteration or deletion by an attacker. However, consider the following drawbacks to the syslog mode.

- The records are susceptible to network attacks such as denial of service and spoofed source addresses.
- The UDP protocol can drop packets or can deliver packets out of order.
- The 1024 character limit for syslog entries can cause some audit records to be truncated in the log.
- On a single system, not all audit records are collected, and might not be displayed in order.
- Each audit record is stamped with the local system's date and time. Thus, you cannot rely on the time stamp to construct an audit trail for several systems.

## Storing and Managing the Audit Trail

When the audit\_binfile plugin is active, an *audit file system* holds audit files in binary format. A typical installation uses /var/audit and can use additional file systems. The contents of all audit file systems comprise the *audit trail*. Audit records are stored in these file systems in the following order:

- **Primary audit file system**— /var/audit functions as the default file system for audit files for a system

---

**Note** - /var/audit is actually a symbolic link to the /var/share file system, which enables shared access as part of a root pool. But, Oracle Solaris treats /var/audit like a file system.

---

- **Secondary audit file systems** – File systems where the audit files for a system are placed at administrator discretion

The file systems are specified as arguments to the p\_dir attribute of the audit\_binfile plugin. A file system is not used until a file system that is earlier in the list is full. For an

example with a list of file system entries, see [“How to Create ZFS File Systems for Audit Files” on page 74](#).

Placing the audit files in the default audit root directory assists the audit reviewer when reviewing the audit trail. The `auditreduce` command uses the audit root directory to find all files in the audit trail. The `/var/audit` functions as the default audit root directory.

You can use the following options with the `auditreduce` command:

- The `-M` option to the `auditreduce` command can be used to specify the audit files from a specific physical machine.
- The `-S` option can be used to specify a different audit file system.

For examples of the use of the `auditreduce` command, see [“How to Merge Audit Files From the Audit Trail” on page 97](#). For more information, see the `auditreduce(1M)` man page.

The audit service provides commands to combine and filter files from the audit trail. The `auditreduce` command can merge audit files from the audit trail. The command can also filter files to locate particular events. The `praudit` command reads the binary files. Options to the `praudit` command provide output that is suitable for scripting and for browser display.

## Ensuring Reliable Time Stamps

When you merge audit logs from several systems, the date and time on those systems must be accurate. Similarly, when you send audit logs to a remote system, the recording system and the repository system must have accurate clocks. The Network Time Protocol (NTP) keeps system clocks accurate and coordinated. For more information, see [Chapter 3, “Time-Related Services” in \*Introduction to Oracle Solaris 11.2 Network Services\*](#) and the `ntpd(1M)` man page.

## Managing a Remote Repository

After the `audit_remote` plugin is configured, a remote repository receives the audit records. The ARS provides a receiver for audit records. The audit records stream to the ARS over a protected connection and can be stored similarly to how they are stored locally. To configure an ARS, see [“How to Configure a Remote Repository for Audit Files” on page 82](#). For a description of the ARS, see [“Audit Remote Server” on page 117](#) and the `ars(5)` man page.

## How Is Auditing Related to Security?

Auditing helps to detect potential security breaches by revealing suspicious or abnormal patterns of system usage. Auditing also provides a means to trace suspect actions back to

a particular user, thus serving as a deterrent. Users who know that their activities are being audited are less likely to attempt malicious activities.

To protect a computer system, especially a system on a network, requires mechanisms that control activities before system processes or user processes begin. Security requires tools that monitor activities as the activities occur. Security also requires reports of activities after the activities have happened.

Set audit parameters before users log in or system processes begin, because most audit activity involves monitoring current events and reporting the events that meet the specified parameters. How the audit service monitors and reports these events is discussed in detail in [Chapter 2, “Planning for Auditing”](#) and [Chapter 3, “Managing the Audit Service”](#).

Auditing cannot prevent hackers from unauthorized entry. However, the audit service can report, for example, that a specific user performed specific actions at a specific time and date. The audit report can identify the user by entry path and user name. Such information can be reported immediately to your terminal and to a file for later analysis. Thus, the audit service provides data that helps you determine the following:

- How system security was compromised
- What loopholes need to be closed to ensure the desired level of security

## How Does Auditing Work?

Auditing generates audit records when specified events occur. Most commonly, events that generate audit records include the following:

- System startup and system shutdown
- Login and logout
- Process creation or process destruction, or thread creation or thread destruction
- Opening, closing, creating, destroying, or renaming of objects
- Use of rights
- Identification actions and authentication actions
- Permission changes by a process or user
- Administrative actions, such as installing a package
- Site-specific applications

Audit records are generated from three sources:

- By an application
- As a result of an [asynchronous audit event](#)
- As a result of a process system call

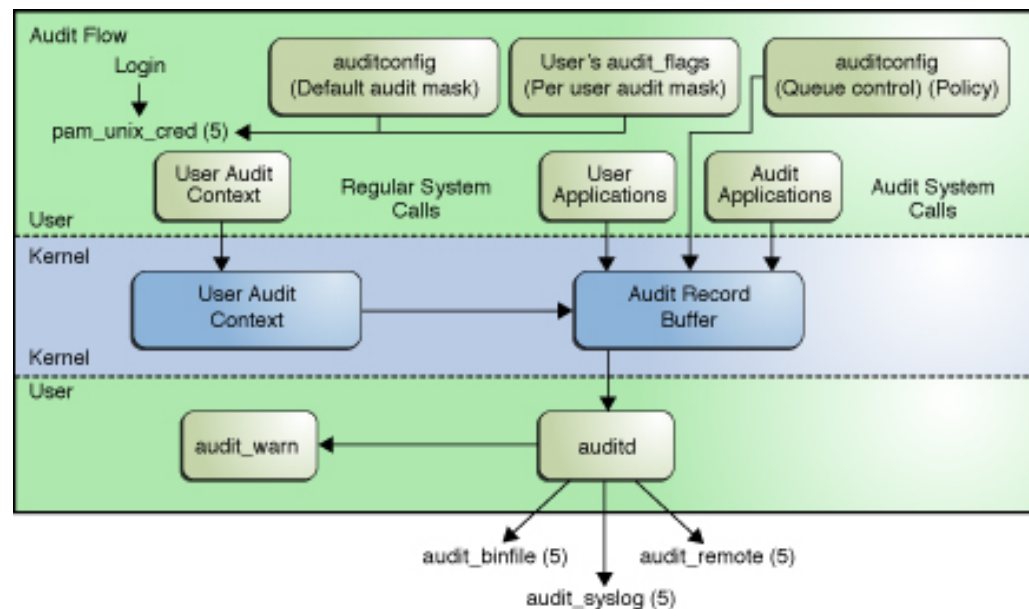
After the relevant event information has been captured, the information is formatted into an audit record. Contained in each audit record is information that identifies the event, what

caused the event, the time of the event, and other relevant information. This record is then placed in an audit queue and sent to the active *plugins* for storage. At least one plugin must be active or the remote audit server must be configured, although all plugins can be active. Plugins are described in [“How Is Auditing Configured?” on page 23](#) and [“Audit Plugin Modules” on page 17](#).

## How Is Auditing Configured?

During system configuration, you *preselect* which classes of audit records to monitor. You can also fine-tune the degree of auditing that is done for individual users. The following figure shows details of the flow of auditing in Oracle Solaris.

**FIGURE 1** Flow of Auditing



After audit data is collected in the kernel, plugins distribute the data to the appropriate locations.

- The `audit_binfile` plugin places binary audit records in `/var/audit`. By default, the `audit_binfile` plugin is active. Post-selection tools enable you to examine interesting parts of the audit trail.

Audit files can be stored in one or more ZFS pools. The collection of audit files that are linked together is considered an *audit trail*.

- The `audit_remote` plugin sends binary audit records across a protected link to an audit remote server.
- The `audit_syslog` plugin sends text summaries of audit records to the `syslog` utility.

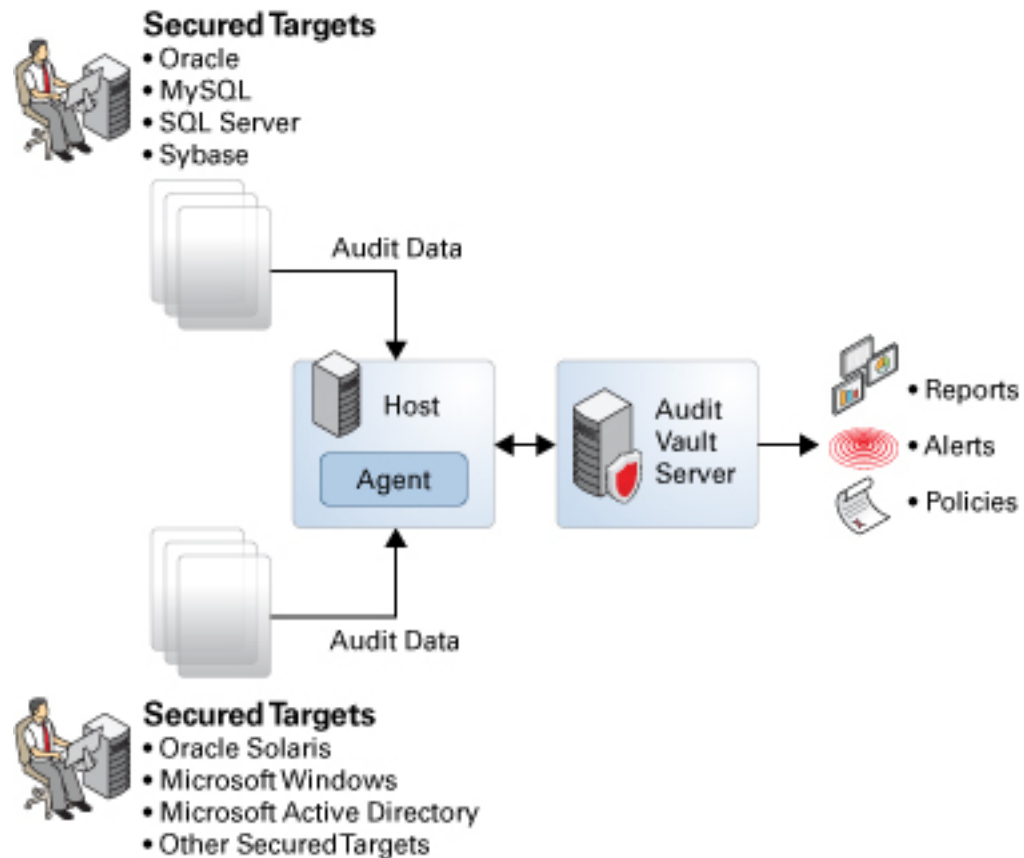
Systems that install non-global zones can audit all zones identically from the global zone. These systems can also be configured to collect different records in the non-global zones. For more information, see “[Auditing on a System With Oracle Solaris Zones](#)” on page 26.

## Using Oracle Audit Vault and Database Firewall for Storage and Analysis of Audit Records

Audit records from an Oracle Solaris system can plug in to Oracle Audit Vault and Database Firewall, release 12.1.0.0. Oracle Audit Vault and Database Firewall automates the consolidation and monitoring of audit data from Oracle and non-Oracle databases. Oracle Audit Vault and Database Firewall can then be used for analysis and reports of audited events on Oracle Solaris systems. For more information, see [Oracle Audit Vault and Database Firewall](http://www.oracle.com/technetwork/products/audit-vault/overview/index.html) (<http://www.oracle.com/technetwork/products/audit-vault/overview/index.html>).

The following figure shows how Oracle Audit Vault and Database Firewall collects Oracle Solaris audit records from designated secured targets. A secured target is any system that stores audit records or data.



**FIGURE 2** Oracle Solaris and Audit Vault

A host system is designated to run the AV agent that communicates with Oracle Audit Vault and Database Firewall. The agent enables Oracle Audit Vault and Database Firewall to receive and process audit data from secured targets. The agent reads the audit records from a designated audit trail on the secured target. These audit records are encoded in the native binary format. The agent converts the data to a format parseable by Oracle Audit Vault and Database Firewall. Oracle Audit Vault and Database Firewall receives the data and generates reports for administrators and security managers as required.

The agent can be installed on a secured target instead of on a separate system. Multiple systems with agents can also be configured to connect to the Audit Vault server. However, when registering secured targets, indicate a specific system with which the AV server communicates to obtain audit data.

To configure Oracle Audit Vault and Database Firewall to accept audit records from both Oracle Solaris secured targets and non-Oracle Solaris secured targets, ensure that the agent is installed and activated on the designated host system. For more information, see the [Oracle Audit Vault and Database Firewall documentation \(http://www.oracle.com/technetwork/database/database-technologies/audit-vault-and-database-firewall/documentation/index.html\)](http://www.oracle.com/technetwork/database/database-technologies/audit-vault-and-database-firewall/documentation/index.html).

## Auditing on a System With Oracle Solaris Zones

A zone is a virtualized operating system environment that is created within a single instance of the Oracle Solaris OS. The audit service audits the entire system, including activities in zones. A system that has installed non-global zones can run a single audit service in the global zones to audit all zones identically. Or, it can run one audit service per non-global zone, including an audit service for the global zone. These audit services would be administered separately.

Sites can run a single audit service in the global zone when:

- The site requires a single-image audit trail.
- The non-global zones are used as application containers. The zones are part of one administrative domain. That is, no non-global zone has customized naming service files.

If all the zones on a system are within one administrative domain, the `zonename` audit policy can be used to distinguish audit events that are configured in different zones.

- Administrators want low audit overhead. The global zone administrator audits all zones identically. Also, the global zone's audit daemon serves all zones on the system.

Sites can run one audit service per non-global zone when:

- The site does not require a single-image audit trail.
- The non-global zones have customized naming service files. These separate administrative domains typically function as network servers.
- Individual zone administrators want to control auditing in the zones that they administer. In per-zone auditing, zone administrators can decide to enable or to disable auditing for the zone that they administer.

The advantages of per-zone auditing are a customized audit trail for each zone, and the ability to disable auditing on a zone-by-zone basis. These advantages can be offset by the administrative overhead. Each zone administrator must administer auditing. Each zone runs its own audit daemon, and has its own audit queue and audit logs. These audit logs must be managed.

## ◆ ◆ ◆ CHAPTER 2

# Planning for Auditing

---

This chapter describes how to plan the customization of the audit service for your Oracle Solaris installation:

- [“Concepts in Planning Auditing” on page 27](#)
- [“Planning Auditing” on page 29](#)
- [“Understanding Audit Policy” on page 34](#)
- [“Controlling Auditing Costs” on page 36](#)
- [“Auditing Efficiently” on page 38](#)

For an overview of auditing, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For procedures to configure auditing at your site, see the following chapters:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 5, “Working With Audit Data”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

For reference information, see [Chapter 7, “Auditing Reference”](#).

## Concepts in Planning Auditing

You want to be selective about what kinds of activities are audited. At the same time, you want to collect useful audit information. You also need to carefully plan who to audit and what to audit. If you are using the default `audit_binfile` plugin, note that audit files can quickly grow to fill the available space.

## Planning a Single System Audit Trail

---

**Note** - Implementing a single system audit trail applies only to the `audit_binfile` plugin.

---

Systems within a single administrative domain can create a single-system image audit trail.

To create a single-system image audit trail for a site, follow these requirements:

- Use the same naming service for all systems.  
For correct interpretation of the audit records, the `passwd`, `group`, and `hosts` files must be consistent.
- Configure the audit service identically on all systems. For information about displaying and modifying the service settings, see the [auditconfig\(1M\)](#) man page.
- Use the same `audit_warn`, `audit_event`, and `audit_class` files for all systems.

Refer to “[How to Plan Who and What to Audit](#)” on page 30 for additional considerations for enabling auditing on the systems.

## Planning Auditing in Zones

If your system contains non-global zones, the zones can be audited as the global zone is audited, or the audit service for each non-global zone can be configured, enabled, and disabled separately. For example, you could audit only the non-global zones and not audit the global zone.

For a discussion of the trade-offs, see “[Auditing on a System With Oracle Solaris Zones](#)” on page 26.

The following options are available when implementing auditing in zones.

## Implementing One Audit Service for All Zones

Auditing all zones identically can create a single-image audit trail. A single-image audit trail occurs when you are using the `audit_binfile` or the `audit_remote` plugin, and all zones on a system are part of one administrative domain. The audit records can then be easily compared because the records in every zone are preselected with identical settings.

This configuration treats all zones as part of one system. The global zone runs the only audit service on a system and collects audit records for every zone. You customize the `audit_class` and `audit_event` files only in the global zone, then copy these files to every non-global zone.

Use the following guidelines when configuring a single audit service for all the zones.

- Use the same naming service for every zone.

---

**Note** - If naming service files are customized in non-global zones, and `perzone` policy is not set, then careful use of the audit tools is required to select usable records. A user ID in one zone can refer to a different user from the same ID in a different zone.

---

- Enable the audit records to include the name of the zone.

To put the zone name as part of the audit record, set the `zonename` policy in the global zone. The `auditreduce` command can then select audit events by zone from the audit trail. For an example, see the [auditreduce\(1M\)](#) man page.

To plan a single-image audit trail, refer to [“How to Plan Who and What to Audit” on page 30](#). Start with the first step. The global zone administrator must also set aside storage, as described in [“How to Plan Disk Space for Audit Records” on page 32](#).

## Implementing One Audit Service Per Zone

Choose to configure per-zone auditing if different zones use different naming service databases, or if zone administrators want to control auditing in their zones.

---

**Note** - To audit non-global zones, the `perzone` policy must be set but the audit service does not have to be enabled in the global zone. Non-global zone auditing is configured and its audit service is enabled and disabled separately from the global zone.

---

- When you configure per-zone auditing, you set the `perzone` audit policy in the global zone. If per-zone auditing is set before a non-global zone is first booted, auditing begins at the zone's first boot. To set audit policy, see [“How to Configure Per-Zone Auditing” on page 68](#).
- Each zone administrator configures auditing for the zone.  
A non-global zone administrator can set all policy options except `perzone` and `ahlt`.
- Each zone administrator can enable or disable auditing in the zone.
- To generate records that can be traced to their originating zone during review, set the `zonename` audit policy.

## Planning Auditing

The following task map points to the major tasks that are required for planning disk space and which events to record.

**TABLE 2** Planning Auditing Task Map

Task	For Instructions
Determine who and what to audit	<a href="#">“How to Plan Who and What to Audit” on page 30</a>
Plan storage space for the audit trail	<a href="#">“How to Plan Disk Space for Audit Records” on page 32</a>
Plan transmission of the audit trail to a remote server	<a href="#">“How to Prepare to Stream Audit Records to Remote Storage” on page 33</a>

## ▼ How to Plan Who and What to Audit

**Before You Begin** If you are implementing non-global zones, review [“Planning Auditing in Zones” on page 28](#) before using this procedure.

### 1. Determine the audit policy.

By default, only the `cnt` policy is enabled.

Use the `auditconfig -lspolicy` command to see a description of available policy options.

- For the effects of the policy options, see [“Understanding Audit Policy” on page 34](#).
- For the effect of the `cnt` policy, see [“Audit Policies for Asynchronous and Synchronous Events” on page 118](#).
- To set audit policy, see [“How to Change Audit Policy” on page 48](#).

### 2. Determine whether you want to add event-to-class mappings.

In almost all situations, the default mapping is sufficient. However, if you add new classes, you should add them to event-to-class mappings.

For an example, see [“How to Change an Audit Event's Class Membership” on page 55](#).

### 3. Determine which audit classes to preselect.

The best time to add audit classes or to change the default classes is before users log in to the system.

The audit classes that you preselect with the `-setflags` and `-setnaflags` options to the `auditconfig` command apply to all users and processes. You can preselect a class for success, for failure, or for both.

For the list of audit classes, read the `/etc/security/audit_class` file.

### 4. Determine user modifications to the system-wide preselections.

If you decide that some users should be audited differently from the system, you can modify the `audit_flags` security attribute for individual users or for a rights profile. The user preselection mask is modified for users whose audit flags are explicitly set or who are assigned a rights profile with explicit audit flags.

For the procedure, see [“How to Configure a User's Audit Characteristics” on page 45](#). For which audit flag values are in effect, see [“Order of Search for Assigned Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 5. Decide how to manage the `audit_warn` email alias.

The `audit_warn` script is run whenever the audit system detects a situation that requires administrative attention. By default, the `audit_warn` script sends email to an `audit_warn` alias and sends a message to the console.

To set up the alias, see [“How to Configure the audit\\_warn Email Alias” on page 53](#).

## 6. Decide in which format and where to collect audit records.

You have three choices.

- By default, store binary audit records locally. The default storage directory is `/var/audit`. To further configure the `audit_binfile` plugin, see [“How to Create ZFS File Systems for Audit Files” on page 74](#).
- Stream binary audit records to a remote protected repository by using the `audit_remote` plugin. You must have a receiver for the records. For the requirements, see [“Managing a Remote Repository” on page 21](#). For the procedure, see [“How to Send Audit Files to a Remote Repository” on page 80](#).
- Send audit record summaries to syslog by using the `audit_syslog` plugin. For the procedure, see [“How to Configure syslog Audit Logs” on page 86](#).

For a comparison of binary and syslog formats, see [“Audit Logs” on page 18](#).

## 7. Determine when to warn the administrator about shrinking disk space.

---

**Note** - This step applies only to the `audit_binfile` plugin.

---

When disk space on an audit file system drops below the minimum free space percentage, or soft limit, the audit service switches to the next available audit directory. The service then sends a warning that the soft limit has been exceeded.

To see how to set a minimum free space percentage, see [Example 25, “Setting a Soft Limit for Warnings,” on page 80](#).

## 8. Decide what action to take when all the audit directories are full.

---

**Note** - This step applies only to the `audit_binfile` plugin.

---

In the default configuration, the `audit_binfile` plugin is active and the `cnt` policy is set. In this configuration, when the kernel audit queue is full, the system continues to work. The system counts the audit records that are dropped but does not record the events. For greater security, you can disable the `cnt` policy and enable the `ahlt` policy. The `ahlt` policy stops the system when an asynchronous event cannot be placed in the audit queue.

However, if the `audit_binfile` queue is full, and the queue for another active plugin is not full, then the kernel queue will continue to send records to the plugin that is not full. When the `audit_binfile` queue can again accept records, the audit service will resume sending records to it.

For a discussion of the `cnt` and `ahlt` policy options, see [“Audit Policies for Asynchronous and Synchronous Events” on page 118](#). To see how to configure these policy options, see [Example 10, “Setting the ahlT Audit Policy Option,” on page 50](#).

---

**Note** - The `cnt` or `ahlt` policy is not triggered if the queue for at least one plugin is accepting audit records.

---

## Planning Disk Space for Audit Records

The `audit_binfile` plugin creates an audit trail. The audit trail requires dedicated file space. This space must be available and secure. The system uses the `/var/audit` file system for initial storage. You can configure additional audit file systems for audit files. The following procedure covers the issues that you must resolve when you plan for audit trail storage.

### ▼ How to Plan Disk Space for Audit Records

**Before You Begin** If you are implementing non-global zones, complete [“Planning Auditing in Zones” on page 28](#) before using this procedure.

This procedure assumes that you are using the `audit_binfile` plugin.

- 1. Determine how much auditing your site needs.**

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

For guidance on how to reduce space requirements while still maintaining site security, as well as how to design audit storage, see [“Controlling Auditing Costs” on page 36](#) and [“Auditing Efficiently” on page 38](#).

For practical steps, see [“Volume of Audit Records Is Large” on page 106](#), [“How to Compress Audit Files on a Dedicated File System” on page 63](#), and [Example 34, “Combining and Reducing Audit Files,” on page 92](#).

- 2. Determine which systems are to be audited and configure their audit file systems.**

Create a list of all the file systems that you plan to use. For configuration guidelines, see [“Storing and Managing the Audit Trail” on page 20](#) and the `auditreduce(1M)` man page. To specify the audit file systems, see [“How to Assign Audit Space for the Audit Trail” on page 77](#).

- 3. Synchronize the clocks on all systems.**

For more information, see [“Ensuring Reliable Time Stamps” on page 21](#).



## Preparing to Stream Audit Records to Remote Storage

The `audit_remote` plugin sends the binary audit trail to an ARS in the same format as the `audit_binfile` plugin writes to the local audit files. The `audit_remote` plugin uses the `libgss` library to authenticate the ARS, and a GSS-API mechanism to protect the transmission with privacy and integrity. For reference, see [“What Is the Kerberos Service?” in \*Managing Kerberos and Other Authentication Services in Oracle Solaris 11.3\*](#) and [“Kerberos Utilities” in \*Managing Kerberos and Other Authentication Services in Oracle Solaris 11.3\*](#).

The only currently supported GSS-API mechanism is `kerberosv5`. For more information, see the [`mech\(4\)`](#) man page.

### ▼ How to Prepare to Stream Audit Records to Remote Storage

---

**Note** - If you have a Kerberos realm configured with an identified Audit Remote Server (ARS) and all audited systems within the realm, you can skip this procedure. The steps to configure the ARS and the audited systems are covered in [“How to Configure a Remote Repository for Audit Files” on page 82](#) and [“How to Send Audit Files to a Remote Repository” on page 80](#).

To verify whether a Kerberos realm is configured, issue the following command. The sample output indicates that Kerberos is not installed on the system.

```
# pkg info system/security/kerberos-5
pkg: info: no packages matching these patterns are installed on the system.
```

---

**Before You Begin** This procedure assumes that you are using the `audit_remote` plugin.

**1. Install the master KDC (Key Distribution Center) package.**

You can use the system that will serve as the ARS, or you can use a nearby system. The ARS sends a significant amount of authentication traffic to the master KDC.

```
# pkg install pkg:/system/security/kerberos-5
```

On the master KDC, you use the Kerberos `kdcmgr` and `kadmin` commands to manage the realm. For more information, see the [`kdcmgr\(1M\)`](#) and [`kadmin\(1M\)`](#) man pages.

**2. On every audited system that will send audit records to the ARS, install the master KDC package.**

```
# pkg install pkg:/system/security/kerberos-5
```

This package includes the `kclient` command. On these systems, you run the `kclient` command to connect with the KDC. For more information, see the [`kclient\(1M\)`](#) man page.

### 3. Synchronize the clocks in the KDC realm.

If the clock skew is too big between the audited systems and the ARS, the attempt at connection will fail. After a connection is established, the local time on the ARS determines the names of the stored audit files, as described in [“Conventions for Binary Audit File Names” on page 120](#).

For more information about the clocks, see [“Ensuring Reliable Time Stamps” on page 21](#).

## Understanding Audit Policy

Audit policy determines the characteristics of the audit records for the local system. You use the `auditconfig` command to set these policies. For more information, see the [auditconfig\(1M\)](#) man page.

Most audit policy options are disabled by default to minimize storage requirements and system processing demands. These options are properties of the audit service and determine the policies that are in effect at system boot. For more information, see the [auditconfig\(1M\)](#) man page.

Use the following table to determine whether the needs of your site justify the additional overhead that results from enabling one or more audit policy options.

**TABLE 3** Effects of Audit Policy Options

Policy Name	Description	Policy Considerations
ahlt	<p>This policy applies to asynchronous events only. When disabled, this policy allows the event to complete without an audit record being generated.</p> <p>When enabled, this policy stops the system when the audit queue is full. Administrative intervention is required to clean up the audit queue, make space available for audit records, and reboot. This policy can be enabled only in the global zone. The policy affects all zones.</p>	<p>The disabled option is preferable when system availability is more important than security.</p> <p>The enabled option is preferable in an environment where security is paramount. For a fuller discussion, see <a href="#">“Audit Policies for Asynchronous and Synchronous Events” on page 118</a>.</p>
arge	<p>When disabled, this policy omits environment variables of an executed program from the <code>execve</code> audit record.</p> <p>When enabled, this policy adds the environment variables of an executed program to the <code>execve</code> audit record. The resulting audit records contain much more detail than when this policy is disabled.</p>	<p>The disabled option collects much less information than the enabled option. For a comparison, see <a href="#">“How to Audit All Commands by Users” on page 57</a>.</p> <p>The enabled option is preferable when you are auditing a few users. The option is also useful when you are unsure about the environment variables that are being used in programs in the <code>ex</code> audit class.</p>
argv	<p>When disabled, this policy omits the arguments of an executed program from the <code>execve</code> audit record.</p>	<p>The disabled option collects much less information than the enabled option. For a comparison, see <a href="#">“How to Audit All Commands by Users” on page 57</a>.</p>

Policy Name	Description	Policy Considerations
	When enabled, this policy adds the arguments of an executed program to the execve audit record. The resulting audit records contain much more detail than when this policy is disabled.	The enabled option is preferable when you are auditing a few users. The option is also useful when you have reason to believe that unusual programs in the ex audit class are being run.
cnt	<p>When disabled, this policy blocks a user or application from running. The blocking happens when audit records cannot be added to the audit trail because the audit queue is full.</p> <p>When enabled, this policy allows the event to complete without an audit record being generated. The policy maintains a count of audit records that are dropped.</p>	<p>The disabled option is preferable in an environment where security is paramount.</p> <p>The enabled option is preferable when system availability is more important than security. For a fuller discussion, see <a href="#">“Audit Policies for Asynchronous and Synchronous Events” on page 118</a>.</p>
group	<p>When disabled, this policy does not add a groups list to audit records.</p> <p>When enabled, this policy adds a groups list to every audit record as a special token.</p>	<p>The disabled option usually satisfies requirements for site security.</p> <p>The enabled option is preferable when you need to audit the supplemental groups to which the subject belongs to.</p>
path	<p>When disabled, this policy records in an audit record at most one path that is used during a system call.</p> <p>When enabled, this policy records every path that is used in conjunction with an audit event to every audit record.</p>	<p>The disabled option places at most one path in an audit record.</p> <p>The enabled option enters each file name or path that is used during a system call in the audit record as a path token.</p>
perzone	<p>When disabled, this policy maintains a single audit configuration for a system. One audit service runs in the global zone. Audit events in specific zones can be located in the audit record if the zonename audit token was preselected.</p> <p>When enabled, this policy maintains a separate audit configuration, audit queue, and audit logs for each zone. An audit service runs in each zone. This policy can be enabled in the global zone only.</p>	<p>The disabled option is useful when you have no special reason to maintain a separate audit log, queue, and daemon for each zone.</p> <p>The enabled option is useful when you cannot monitor your system effectively by simply examining audit records with the zonename audit token.</p>
public	<p>When disabled, this policy does not add read-only events of public objects to the audit trail when the reading of files is preselected. Audit classes that contain read-only events include fr, fa, and cl.</p> <p>When enabled, this policy records every read-only audit event of public objects if an appropriate audit class is preselected.</p>	<p>The disabled option usually satisfies requirements for site security.</p> <p>The enabled option is rarely useful.</p>
seq	<p>When disabled, this policy does not add a sequence number to every audit record.</p> <p>When enabled, this policy adds a sequence number to every audit record. The sequence token holds the sequence number.</p>	<p>The disabled option is sufficient when auditing is running smoothly.</p> <p>The enabled option is preferable when the cnt policy is enabled. The seq policy enables you to determine when data was discarded. Alternatively, you can use the auditstat command to view dropped records.</p>

Policy Name	Description	Policy Considerations
trail	When disabled, this policy does not add a trailer token to audit records.	The disabled option creates a smaller audit record.
	When enabled, this policy adds a trailer token to every audit record.	The enabled option clearly marks the end of each audit record with a trailer token. The trailer token is often used with the sequence token. The trailer token aids in the recovery of damaged audit trails.
zonename	When disabled, this policy does not include a zonename token in audit records.	The disabled option is useful when you do not need to track audit behavior per zone.
	When enabled, this policy includes a zonename token in every audit record.	The enabled option is useful when you want to isolate and compare audit behavior across zones by post-selecting records according to zone.

## Controlling Auditing 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 costs of auditing:

- Cost of increased processing time
- Cost of analysis of audit data

If you are using the default plugin, `audit_binfile`, you must also consider the storage cost of audit data.

### Cost of Increased Processing Time of Audit Data

The cost of increased processing time is the least significant of the costs of auditing. Auditing generally does not occur during computation-intensive tasks, such as image processing, complex calculations, and so forth. Also, if you are using the `audit_binfile` plugin, audit administrators can move the post-selection tasks from the audited system to systems that are dedicated to analyzing audit data. Finally, unless kernel events are preselected, the audit service has no measurable impact on system performance.

### Cost of Analysis of Audit Data

The cost of analysis is roughly proportional to the amount of audit data that is collected. The cost of analysis includes the time that is required to merge and review audit records.

For records that are collected by the `audit_binfile` plugin, cost also includes the time that is required to archive the records and their supporting name service databases, and to keep the records in a safe place. Supporting databases include `groups`, `hosts`, and `passwd`.

The fewer records that you generate, the less time that is required to analyze the audit trail. The sections [“Cost of Storage of Audit Data” on page 37](#) and [“Auditing Efficiently” on page 38](#) describe ways to audit efficiently. Efficient auditing reduces the amount of audit data while still providing enough coverage to achieve your site's security goals.

## Cost of Storage of Audit Data

If you are using the `audit_binfile` plugin, 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 traceability and accountability that is required

Because these factors vary from site to site, no formula can predetermine the amount of disk space to set aside for audit data storage. Use the following information as a guide:

- Understand the audit classes

Before you configure auditing, you should understand the types of events that the classes contain. You can change the audit event-class mappings to optimize audit record collection.
- Preselect audit classes judiciously to reduce the volume of records that are generated.

Full auditing, that is, with the `all` class, fills disk space quickly. Even a simple task such as compiling a program could generate a large audit file. A program of modest size could generate thousands of audit records in less than a minute.

For example, by omitting the `file_read` audit class, `fr`, you can significantly reduce audit volume. By choosing to audit for failed operations only, you can at times reduce audit volume. For example, by auditing for failed `file_read` operations, `-fr`, you can generate far fewer records than by auditing for all `file_read` events.
- If you are using the `audit_binfile` plugin, efficient audit file management is also important. For example, you can compress a ZFS file system that is dedicated to audit files.
- Develop a philosophy of auditing for your site.

Base your philosophy on measures such as the amount of traceability that your site requires, and the types of users that you administer.

## Auditing Efficiently

The following techniques can help you achieve your organization's security goals while auditing more efficiently.

- For as many audit classes as possible, preselect those classes only for users and roles, not system-wide.
- Randomly audit only a certain percentage of users at any one time.
- If the `audit_binfile` plugin is active, reduce the disk storage requirements for audit files by filtering, merging, and compressing the files. Develop procedures for archiving the files, for transferring the files to removable media, and for storing the files offline.
- Monitor the audit data in real time for unusual behaviors.
  - `audit_syslog` plugin – You can extend management and analysis tools that you have already developed to handle the audit records in `syslog` files.
  - `audit_binfile` plugin – You can set up procedures to monitor the audit trail for certain activities. You can write a script to trigger an automatic increase in the auditing of certain users or certain systems in response to detection of unusual events.

For example, you could write a script that does the following:

1. Monitors the creation of audit files on the audited systems.
2. Processes the audit files with the `tail` command.

The piping of the output from the `tail -0f` command through the `praudit` command can yield a stream of audit records as the records are generated. For more information, see the [tail\(1\)](#) man page.

3. Analyzes this stream for unusual message types or other indicators, and delivers the analysis to the auditor.

Alternatively, the script can be used to trigger automatic responses.

4. Constantly monitors the audit file systems for the appearance of new `not_terminated` audit files.
5. Terminates outstanding `tail` processes when their files are no longer being written to.

## Managing the Audit Service

---

This chapter provides procedures to help you configure and manage auditing on an Oracle Solaris system. The chapter covers the following tasks:

- [“Default Configuration of the Audit Service” on page 39](#)
- [“Configuring the Audit Service” on page 42](#)
- [“Customizing What Is Audited” on page 56](#)
- [“Configuring the Audit Service in Zones” on page 65](#)
- [“Example: Configuring Oracle Solaris Auditing” on page 69](#)

In addition, the following chapters describe other audit management tasks:

- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 5, “Working With Audit Data”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

For an overview of the audit service, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For reference information, see [Chapter 7, “Auditing Reference”](#).

### Default Configuration of the Audit Service

The audit service has a default configuration and is immediately operational on the global zone after you install Oracle Solaris. No additional action is required to enable or configure the service to become usable. With its default configuration, the audit service records the following operations:

- Login and logout operations
- Use of the su command
- Screen lock and screen unlock operations

Because the service's default configuration has no performance impact on the system, disabling the service on performance grounds is not required.

Provided that you have the appropriate audit-related rights, such as those in the Audit Review Rights profile, you can review the audit logs. The logs are stored in `/var/audit`. You view these files by using the `praudit` and `auditreduce` commands. For more information, see [“Displaying Audit Trail Data” on page 89](#).

The subsequent sections in this chapter provide instructions for customizing the audit service configuration, if the default configuration is insufficient for your needs.

## Displaying Audit Service Defaults

The audit service is regulated by the following parameters:

- Classes of attributable and non-attributable events
- Audit policy
- Audit plugins
- Queue controls

To display the audit service defaults, you typically use the `auditconfig -get*` subcommands. These subcommands display the current configuration of the parameter that is represented by the asterisk (\*), such as `-getflags`, `-getpolicy`, or `-getqctrl`. To display information about classes for non-attributable events, use the `auditconfig -getnaflags` subcommand.

For more information about the `auditconfig` command, see the [auditconfig\(1M\)](#) man page.

---

**Note** - To display the audit service configuration, you must become an administrator who is assigned the Audit Configuration or Audit Control rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

---

The following examples show the appropriate command syntax to use to display the default audit configuration settings.

### EXAMPLE 1     Displaying Default Class for Events

In this example, two subcommands are used to display the preselected classes for attributable and non-attributable events respectively. To see which events are assigned to a class, and therefore which events are being recorded, run the `auditrecord -c class` command.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
```

`lo` is the flag for the login/logout audit class. The format of the mask output is *(success,failure)*.



```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

**EXAMPLE 2**     Displaying the Default Audit Policy

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
```

The *active* policy is the current policy, but the policy value is not being stored by the audit service. The *configured* policy is stored by the audit service, so the policy is restored when you restart the audit service.

**EXAMPLE 3**     Displaying the Default Audit Plugins

```
$ auditconfig -getplugin
Plugin: audit_binfile
Attributes: p_age=0h;p_dir=/var/audit;p_fsize=4M;p_minfree=1;

Plugin: audit_syslog (inactive)
Attributes: p_flags=;

Plugin: audit_remote (inactive)
Attributes: p_hosts=;p_retries=3;p_timeout=5;
```

The `audit_binfile` plugin is active by default.

**EXAMPLE 4**     Displaying the Audit Queue Controls

```
$ auditconfig -getqctrl
no configured audit queue hiwater mark
no configured audit queue lowater mark
no configured audit queue buffer size
no configured audit queue delay
active audit queue hiwater mark (records) = 100
active audit queue lowater mark (records) = 10
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

The *active* queue control is the queue control that is currently used by the kernel. The string *no configured* indicates that the system is using the default values.

## Enabling and Disabling the Audit Service

The audit service is enabled by default. If the perzone audit policy is set, zone administrators must enable, refresh, or disable the audit service in each non-global zone as desired. If the perzone audit policy is not set, enabling, refreshing, or disabling the audit service from the global zone is effective for all non-global zones.

To disable or enable the audit service, you must become an administrator who is assigned the Audit Control rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

To disable the audit service, use the following command:

```
# audit -t
```

To enable the audit service, use the following command:

```
# audit -s
```

To verify that the audit service is running, use the following command:

```
# auditconfig -getcond  
audit condition = auditing
```

If the perzone audit policy is set, then you must perform this verification in the non-global zones where you enabled auditing.

For more information, see the [audit\(1M\)](#) and [auditd\(1M\)](#) man pages.

## Configuring the Audit Service

Before you enable auditing on your network, you can modify the defaults to satisfy your site auditing requirements. Best practice is to customize your audit configuration as much as possible before the first users log in.

If you have implemented zones, you can choose to audit all zones from the global zone or to audit non-global zones individually. For an overview, see [“Auditing and Oracle Solaris Zones” on page 114](#). For planning, see [“Planning Auditing in Zones” on page 28](#). For procedures, see [“Configuring the Audit Service in Zones” on page 65](#).

To configure the audit service, you typically use `auditconfig` subcommands. The configuration that is set with these subcommands applies to the whole system.

- `auditconfig -get*` displays the current configuration of the parameter that is represented by the asterisk (\*), as shown in the examples of [“Displaying Audit Service Defaults” on page 40](#).

- `auditconfig -set*` assigns a value to the parameter that is represented by the asterisk (\*), such as `-setflags`, `-setpolicy`, or `-setqctrl`. To configure classes for non-attributable events, you use the `auditconfig setnaflags` subcommand.
- `auditconfig -conf` configures kernel audit event to class mappings. Runtime class mappings are changed to match those in the audit event to class database file.

You can also customize auditing to apply to users or profiles, rather than to the entire system. Audit class preselections for each user are specified by the `audit_flags` security attribute. These user-specific values, plus the preselected classes for the system, determine the user's audit mask, as described in [“Process Audit Characteristics” on page 119](#).

By preselecting classes on a per user basis rather than on a per system basis, you can sometimes reduce the impact of auditing on system performance. Also, you might want to audit specific users slightly differently from the system.

To configure auditing that applies to users or profiles, you use the following commands:

- `usrattr` displays the `audit_flags` value that is set for users. By default, users are audited for the system-wide settings only.
- `usermod -K` sets flags that apply to users.
- `profile` sets flags that apply to profiles.

For a description of the `usrattr` command, see the [`usrattr\(1\)`](#) man page. For a description of the `audit_flags` keyword, see the [`user\_attr\(4\)`](#) man page and the [`audit\_flags\(5\)`](#).

The following task map points to the procedures for configuring auditing. All tasks are optional.

**TABLE 4** Configuring the Audit Service Task Map

Task	Description	For Instructions
Select which events are audited.	Preselects system-wide audit classes. If an event is attributable, then all users are audited for this event.	<a href="#">“How to Preselect Audit Classes” on page 44</a>
Select which events are audited for specific users.	Sets user-specific differences from the system-wide audit classes.	<a href="#">“How to Configure a User's Audit Characteristics” on page 45</a>
Specify audit policy.	Defines additional audit data that your site requires.	<a href="#">“How to Change Audit Policy” on page 48</a>
Specify queue controls.	Modifies the default buffer size, audit records in the queue, and interval between writing audit records to the buffer.	<a href="#">“How to Change Audit Queue Controls” on page 51</a>
Create the <code>audit_warn</code> email alias.	Defines who receives email warnings when the audit service needs attention.	<a href="#">“How to Configure the <code>audit_warn</code> Email Alias” on page 53</a>
Configure audit logs.	Configures the location of audit records for each plugin.	<a href="#">“Configuring Audit Logs” on page 73</a>
Add audit classes.	Reduces the number of audit records by creating a new audit class to hold critical events.	<a href="#">“How to Add an Audit Class” on page 54</a>

Task	Description	For Instructions
Change event-to-class mappings.	Reduces the number of audit records by changing the event-class mapping.	<a href="#">“How to Change an Audit Event’s Class Membership” on page 55</a>

## ▼ How to Preselect Audit Classes

Preselect audit classes that contain the events that you want to monitor. Events that are not in preselected classes are not recorded.

**Before You Begin** You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Determine the current preselected classes.

```
# auditconfig -getflags
...

# auditconfig -getnaflags
...
```

For an explanation of the output, see [“Displaying Audit Service Defaults” on page 40](#).

### 2. Preselect the attributable classes.

```
# auditconfig -setflags lo,ps,fw
user default audit flags = ps,lo,fw(0x101002,0x101002)
```

This command audits the events in the login/logout, process start/stop, and file write classes for success and for failure.

---

**Note** - The `auditconfig -setflags` command *replaces* the current preselection, so you must specify all classes that you want to preselect.

---

### 3. Preselect the non-attributable classes.

The na class contains PROM, boot, and non-attributable mounts, among other events.

```
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

lo and na are the only useful arguments to the `-setnaflags` option.

---

**Note** - The `auditconfig -setnaflags` command *replaces* the current preselection, so you must specify all classes that you want to preselect.

---

## ▼ How to Configure a User's Audit Characteristics

These user-specific audit characteristics that you set with this procedure are combined with the preselected classes for the system. Together they determine the user's audit mask, as described in [“Process Audit Characteristics” on page 119](#).

**Before You Begin** You must assume the root role. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. (Optional) Display the audit classes that are currently selected for existing users.

#### a. Display the list of users.

```
# who
adoc   pts/1      Oct 10 10:20 (:0.0)
adoc   pts/2      Oct 10 10:20 (:0.0)
jdoe   pts/5      Oct 12 12:20 (:0.0)
jdoe   pts/6      Oct 12 12:20 (:0.0)
...
```

#### b. Display each user's `audit_flags` attribute value.

```
# userattr audit_flags adoc
# userattr audit_flags jdoe
```

### 2. Set the audit flags in the `user_attr` or in the `prof_attr` database.

For example, you can create a rights profile that defines the rights of a subset of your users. Users who are assigned that rights profile are audited identically.

#### ■ To set audit flags for a user, use the `usermod` command.

```
# usermod -K audit_flags=fw:no jdoe
```

The format of the `audit_flags` keyword is *always-audit:never-audit*.

*always-audit* Lists the audit classes that are audited for this user.

*never-audit* Lists the audit classes that are never audited for the user, even if these audit events are audited system-wide.

To specify multiple audit classes, separate the classes with commas. For more information, see the [audit\\_flags\(5\)](#) man page.

#### ■ To set audit flags for a rights profile, use the `profiles` command.

```
# profiles -p "System Administrator"
```

```
profiles:System Administrator> set name="Audited System Administrator"
profiles:Audited System Administrator> set always_audit=fw,as
profiles:Audited System Administrator> end
profiles:Audited System Administrator> exit
```

When you assign the Audited System Administrator rights profile to a user or a role, that user or role is audited for those flags, subject to search order as described in [“Order of Search for Assigned Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

#### Example 5 Changing Which Events Are Audited for One User

In this example, the audit preselection mask for all users is the following:

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

No user except the administrator is logged in.

To lessen the impact of the AUE\_PFEXEC audit event on system resources, the administrator does not audit this event at the system level. Rather, the administrator preselects the pf class for a user, jdoe. The pf class is created in [Example 15, “Creating a New Audit Class,” on page 54](#).

```
# usermod -K audit_flags=pf:no jdoe
```

The userattr command shows the addition.

```
# userattr audit_flags jdoe
pf:no
```

When the user jdoe logs in, jdoe's audit preselection mask is a combination of the audit\_flags values with the system default values. 289 is the PID of jdoe's login shell.

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = ss,pf,lo(0x01000000008011000,0x01000000008011000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

#### Example 6 Modifying Audit Preselection Exception for One User

In this example, the audit preselection mask for all users is the following:

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

No users except the administrator are logged in.

The administrator decides not to collect failed `ss` events for the `jdoe` user.

```
# usermod -K audit_flags=~ss:~no jdoe
```

The `userattr` command shows the exception.

```
# userattr audit_flags jdoe
^~ss:~no
```

When the user `jdoe` logs in, `jdoe`'s audit preselection mask is a combination of the `audit_flags` values with the system default values. 289 is the PID of `jdoe`'s login shell.

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = +ss,lo(0x11000,0x1000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

#### Example 7 Auditing Selected Users, No System-Wide Auditing

In this example, the login and role activities of four selected users are audited on the system. No audit classes are preselected for the system.

First, the administrator removes all system-wide flags.

```
# auditconfig -setflags no
user default audit flags = no(0x0,0x0)
```

Then, the administrator preselects two audit classes for the four users. The `pf` class is created in [Example 15, “Creating a New Audit Class,” on page 54](#).

```
# usermod -K audit_flags=lo,pf:~no jdoe
# usermod -K audit_flags=lo,pf:~no kdoe
# usermod -K audit_flags=lo,pf:~no pdoe
# usermod -K audit_flags=lo,pf:~no zdoe
```

Then, the administrator preselects the `pf` class for the root role.

```
# userattr audit_flags root
# rolemod -K audit_flags=lo,pf:~no root
# userattr audit_flags root
lo,pf:~no
```

To continue to record unwarranted intrusion, the administrator does not change the auditing of non-attributable logins.

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

**Example 8** Removing a User's Audit Flags

In the following example, the administrator removes all user-specific audit flags. Existing processes of users who are currently logged in continue to be audited.

The administrator runs the `usermod` command with the `audit_flags` keyword set to no value.

```
# usermod -K audit_flags= jdoe
# usermod -K audit_flags= kdoe
# usermod -K audit_flags= ldoe
```

Then, the administrator verifies the removal.

```
# userattr audit_flags jdoe
# userattr audit_flags kdoe
# userattr audit_flags ldoe
```

**Example 9** Creating a Rights Profile for a Group of Users

The administrator wants all administrative rights profiles at the site to explicitly audit the `pf` class. For every rights profile that is going to be assigned, the administrator creates a site-specific version in LDAP that includes audit flags.

First, the administrator clones an existing rights profile, then changes the name and adds audit flags.

```
# profiles -p "Network Wifi Management" -S ldap
profiles: Network Wifi Management> set name="Wifi Management"
profiles: Wifi Management> set desc="Audited wifi management"
profiles: Wifi Management> set audit_always=pf
profiles: Wifi Management> exit
```

After repeating this procedure for every rights profile that is going to be used, the administrator lists the information in the Wifi Management profile.

```
# profiles -p "Wifi Management" -S ldap info
name=Wifi Management
desc=Audited wifi management
auths=solaris.network.wifi.config
help=RtNetWifiMngmnt.html
always_audit=pf
```

## ▼ How to Change Audit Policy

You might change default audit policy to record detailed information about audited commands, to add a zone name to every record, or to satisfy other site security requirements.



**Before You Begin** You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. View the current audit policy.

```
$ auditconfig -getpolicy
configured audit policies = argv,cnt
active audit policies = argv,cnt
```

For an explanation of the output, see [“Displaying Audit Service Defaults” on page 40](#).

### 2. View the available policy options.

```
$ auditconfig -lspolicy
policy string      description:
ahlt               halt machine if it can not record an async event
all               all policies for the zone
arge              include exec environment args in audit recs
argv              include exec command line args in audit recs
cnt               when no more space, drop recs and keep a cnt
group             include supplementary groups in audit recs
none              no policies
path              allow multiple paths per event
perzone           use a separate queue and auditd per zone
public            audit public files
seq               include a sequence number in audit recs
trail             include trailer token in audit recs
windata_down      include downgraded window information in audit recs
windata_up        include upgraded window information in audit recs
zonename          include zonename token in audit recs
```

---

**Note** - The perzone and ahlth policy options can be set only in the global zone. For the trade-offs to using a particular policy option, see [“Understanding Audit Policy” on page 34](#).

---

### 3. Enable or disable selected audit policy options.

```
# auditconfig [ -t ] -setpolicy [prefix]policy[,policy...]
```

**-t** Optional. Creates a temporary, or *active*, policy. You might set a temporary policy for debugging or testing purposes.

A temporary policy is in effect until the audit service is refreshed, or until the policy is modified by the `auditconfig -setpolicy` command, or the system is rebooted.

**prefix** A *prefix* value of + adds the list of policies to the current policy. A *prefix* value of - removes the list of policies from the current policy. Without a

prefix, the audit policy is reset. This option enables you to retain current audit policies.

*policy*                      Selects the policy to be enabled or to be disabled.

**Example 10**    Setting the `ahlt` Audit Policy Option

In this example, strict site security requires the `ahlt` policy.

```
# auditconfig -setpolicy -cnt
# auditconfig -setpolicy +ahlt
```

The plus sign (+) before the `ahlt` policy adds the policy to current policy settings. Without the plus sign, the `ahlt` policy replaces all current audit policies.

**Example 11**    Setting a Temporary Audit Policy

In this example, the `ahlt` audit policy is configured. For debugging, the administrator adds the `trail` audit policy to the active policy (+`trail`) temporarily (-`t`). The `trail` policy aids in the recovery of damaged audit trails.

```
$ auditconfig -setpolicy ahltn
$ auditconfig -getpolicy
configured audit policies = ahltn
active audit policies = ahltn
$ auditconfig -t -setpolicy +trail
$ auditconfig -getpolicy
configured audit policies = ahltn
active audit policies = ahltn, trail
```

The administrator disables the `trail` policy when the debugging is completed.

```
$ auditconfig -setpolicy -trail
$ auditconfig -getpolicy
configured audit policies = ahltn
active audit policies = ahltn
```

Refreshing the audit service by running the `audit -s` command also removes this temporary policy, plus any other temporary values in the audit service. For examples of other temporary values, see [“How to Change Audit Queue Controls” on page 51](#).

**Example 12**    Setting the `perzone` Audit Policy

In this example, the `perzone` audit policy is added to the existing policy in the global zone. The `perzone` policy setting is stored as a permanent property, so `perzone` policy is in effect during

the session and when the audit service is restarted. For the zones, the policy is available at the next zone boot, or when the administrator runs `zlogin` and then runs the `audit -s` command.

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
$ auditconfig -setpolicy +perzone
$ auditconfig -getpolicy
configured audit policies = perzone,cnt
active audit policies = perzone,cnt
```

### Example 13 Collecting Audit Records for External Auditors

In this example, the administrator is collecting audit records to satisfy external auditors' requirements. The administrator decides to use an Audit Remote Server (ARS) to collect information about administrative activities. The administrator also collects actions that cannot be attributed to a user, such as booting.

The administrator sets up ARS. In addition to auditing the `cusa` class, the administrator adds policies to the audit configuration.

```
# auditconfig -setflags cusa
user default audit flags = ex,xa,ua,as,ss,ap,lo,ft(0x80475080,0x80475080)
# auditconfig -setpolicy ahlt,argv,arge
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv
active audit policies = ahlt,arge,argv
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

When the administrator enables the `audit_remote` plugin and refreshes the audit service, the records are collected.

## ▼ How to Change Audit Queue Controls

The audit service provides default values for audit queue parameters. You can inspect, permanently change, and temporarily change these values with the `auditconfig` command.

**Before You Begin** You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. View the current values of the audit queue controls.

```
$ auditconfig -getqctrl
```

...

For an explanation of the output, see [“Displaying Audit Service Defaults” on page 40](#).

## 2. Modify selected audit queue controls.

For examples and a description of the audit queue controls, see the [auditconfig\(1M\)](#) man page.

- To modify some or all audit queue controls, use the `-setqctrl` option.

```
# auditconfig [ -t ] -setqctrl hiwater lowater bufisz interval
```

The high water (*hiwater*) and low water (*lowater*) values indicate the points at which processes are respectively suspended and resume. The points are measured in terms of the number of undelivered audit records. The buffer size (*bufisz*) refers to the buffer size of the queue. Interval indicates the delay, measured in number of ticks, between generation of audit output.

For example, set the *interval* value to 10 without setting the other controls.

```
# auditconfig -setqctrl 0 0 0 10
```

- To modify a specific audit queue control, specify its option. The `-setqdelay` option is the equivalent of `-setqctrl 0 0 0 interval`, as in `auditconfig -setqdelay 10`.

```
# auditconfig [ -t ] -setqhiwater value
# auditconfig [ -t ] -setqlowater value
# auditconfig [ -t ] -setqbufisz value
# auditconfig [ -t ] -setqdelay value
```

### Example 14 Resetting an Audit Queue Control to the Default

The administrator sets all audit queue controls, then changes the *lowater* value in the repository back to the default.

```
# auditconfig -setqctrl 200 5 10216 10
# auditconfig -setqctrl 200 0 10216 10
configured audit queue hiwater mark (records) = 200
no configured audit queue lowater mark
configured audit queue buffer size (bytes) = 10216
configured audit queue delay (ticks) = 10
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 5
active audit queue buffer size (bytes) = 10216
active audit queue delay (ticks) = 10
```

Later, the administrator sets the *lowater* value to the default for the current session.

```
# auditconfig -setqlowater 10
# auditconfig -getqlowater
```

```
configured audit queue lowater mark (records) = 10
active audit queue lowater mark (records) = 10
```

## ▼ How to Configure the audit\_warn Email Alias

The `/etc/security/audit_warn` script generates mail to notify the administrator of audit incidents that might need attention. You can customize the script and you can send the mail to an account other than `root`.

If the `perzone` policy is set, the non-global zone administrator must configure the `audit_warn` email alias in the non-global zone.

**Before You Begin** You must become an administrator who is assigned the `solaris.admin.edit/etc/security/audit_warn` authorization. By default, only the `root` role has this authorization. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### ● Configure the audit\_warn email alias.

Choose one of the following options:

- Replace the `audit_warn` email alias with another email account in the `audit_warn` script. Change the `audit_warn` email alias in the `ADDRESS` line of the script to another address:

```
#ADDRESS=audit_warn          # standard alias for audit alerts
ADDRESS=audadmin             # role alias for audit alerts
```

---

**Note** - For information about the effects of modifying an audit configuration file, see [“Audit Configuration Files and Packaging” on page 115](#).

---

- Redirect the `audit_warn` email to another mail account. Add the `audit_warn` email alias to the appropriate mail aliases file. You could add the alias to the local `/etc/mail/aliases` file or to the `mail_aliases` database in the name service. The `/etc/mail/aliases` entry would resemble the following example if the `root` and `audadmin` email accounts were added as members of the `audit_warn` email alias:

```
audit_warn: root,audadmin
```

Then, run the `newaliases` command to rebuild the random access database for the aliases file.

```
# newaliases
/etc/mail/aliases: 14 aliases, longest 10 bytes, 156 bytes total
```

## ▼ How to Add an Audit Class

When you create your own audit class, you can place into it just those audit events that you want to audit for your site. This strategy can reduce the number of records that are collected and reduce noise in your audit trail.

When you add the class on one system, copy the change to all systems that are being audited. Best practice is to create audit classes before the first users log in.

For information about the effects of modifying an audit configuration file, see [“Audit Configuration Files and Packaging” on page 115](#).

---

**Tip** - In Oracle Solaris you can create your own package that contains files and replace the Oracle Solaris packages with your site-customized files. When you set the `preserve` attribute to `true` in your package, the `pkg` subcommands, such as `verify`, `fix`, `revert`, and so on, will run relative to your packages. For more information, see the `pkg(1)` and `pkg(5)` man pages.

---

**Before You Begin** Choose free bits in the upper 8 bits for your unique entry. Verify which bits are available for customer use in the `/etc/security/audit_class` file.

You must become an administrator who is assigned the `solaris.admin.edit/etc/security/audit_class` authorization. By default, only the `root` role has this authorization. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

1. **(Optional) Save a backup copy of the `audit_class` file.**

```
# cp /etc/security/audit_class /etc/security/audit_class.orig
```

2. **Add new entries to the `audit_class` file.**

Each entry has the following format:

```
0x64bitnumber:flag:description
```

For a description of the fields, see the [`audit\_class\(4\)`](#) man page. For the list of existing classes, read the `/etc/security/audit_class` file.

**Example 15** Creating a New Audit Class

This example creates a class to hold administrative commands that are executed in a role. The added entry to the `audit_class` file is as follows:

```
0x0100000000000000:pf:profile command
```

The entry creates the new `pf` audit class. [Example 16, “Mapping Existing Audit Events to a New Class,” on page 56](#) shows how to populate the new audit class.

**Troubleshooting** If you have customized the `audit_class` file, make sure that any audit flags that are assigned directly to users or rights profiles are consistent with the new audit classes. Errors occur when an `audit_flags` value is not a subset of the `audit_class` file.

## ▼ How to Change an Audit Event's Class Membership

You might want to change an audit event's class membership to reduce the size of an existing audit class, or to place the event in a class of its own.



**Caution** - Never comment out events in the `audit_event` file. This file is used by the `praudit` command to read binary audit files. Archived audit files might contain events that are listed in the file. Also, never remove any existing audit classes.

When you reconfigure audit event-class mappings on one system, copy the change to all systems that are being audited. Best practice is to change event-class mappings before the first users log in.

**Note** - For information about the effects of modifying an audit configuration file, see [“Audit Configuration Files and Packaging” on page 115](#).

**Tip** - In Oracle Solaris you can create your own package that contains files and replace the Oracle Solaris packages with your site-customized files. When you set the `preserve` attribute to `true` in your package, the `pkg` subcommands, such as `verify`, `fix`, `revert`, and so on, will run relative to your packages. For more information, see the `pkg(1)` and `pkg(5)` man pages.

**Before You Begin** You must become an administrator who is assigned the `solaris.admin.edit/etc/security/audit_event` authorization. By default, only the root role has this authorization. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

- (Optional) Save a backup copy of the `audit_event` file.**

```
# cp /etc/security/audit_event /etc/security/audit_event.orig
```

- Change the class to which particular events belong by changing the `class-list` of the events.**

Each entry has the following format:

```
number:name:description:class-list
```

*number*                      The audit event ID.

<i>name</i>	The name of the audit event.
<i>description</i>	Typically, the system call or executable that triggers the creation of an audit record.
<i>class-list</i>	A comma-separated list of audit classes.

### 3. Refresh the kernel events.

```
# auditconfig -conf
Configured 283 kernel events.
```

#### Example 16 Mapping Existing Audit Events to a New Class

This example maps an existing audit event to the new class that was created in [Example 15, “Creating a New Audit Class,” on page 54](#). By default, the AUE\_PFEXEC audit event is mapped to several audit classes. By creating the new class, the administrator can audit AUE\_PFEXEC events without auditing the events in the other classes.

```
# grep pf /etc/security/audit_class
0x0100000000000000:pf:profile command
# grep AUE_PFEXEC /etc/security/audit_event
116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as
# pfedit /etc/security/audit_event
#116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as
116:AUE_PFEXEC:execve(2) with pfexec enabled:pf
# auditconfig -setflags lo,pf
user default audit flags = pf,lo(0x0100000000001000,0x010000000001000)
```

## Customizing What Is Audited

The following task map points to procedures to configure auditing that is specific to your needs.

**TABLE 5** Customizing Auditing Task Map

Task	Description	For Instructions
Audit everything that a user does on the system.	Audit one or more users for every command.	<a href="#">“How to Audit All Commands by Users” on page 57</a>
Change the audit events that are being recorded and have the change affect existing sessions.	Update a user's preselection mask.	<a href="#">“How to Update the Preselection Mask of Logged In Users” on page 61</a>
Locate modifications to particular files.	Audit file modifications, then use the <code>auditreduce</code> command to find particular files.	<a href="#">“How to Find Audit Records of Changes to Specific Files” on page 59</a>



Task	Description	For Instructions
Use less file system space for audit files.	Use ZFS quotas and compression.	<a href="#">“How to Compress Audit Files on a Dedicated File System” on page 63</a>
Remove audit events from the audit_event file.	Correctly update the audit_event file.	<a href="#">“How to Prevent the Auditing of Specific Events” on page 62</a>

## ▼ How to Audit All Commands by Users

As part of site security policy, some sites require audit records of all commands that are run by the root account and administrative roles. Some sites can require audit records of all commands by all users. Additionally, sites can require that the command arguments and environment be recorded.

**Before You Begin** To preselect audit classes and set audit policy, you must become an administrator who is assigned the Audit Configuration rights profile. To assign audit flags to users, roles, and rights profiles, you must have the `solaris.audit.assign` authorization which is included in the Rights Delegation profile. The role root has all authorizations.

### 1. Display user level event information for `lo` and `ex` classes.

The `ex` class audits all calls to the `exec()` and `execve()` functions.

The `lo` class audits logins, logouts, and screen locks. The following output lists all the events in the `ex` and `lo` classes.

```
% auditconfig -lsevent | egrep " lo |,lo|lo,"
AUE_login          6152 lo login - local
AUE_logout          6153 lo logout
AUE_telnet          6154 lo login - telnet
AUE_rlogin          6155 lo login - rlogin
AUE_rshd            6158 lo rsh access
AUE_su              6159 lo su
AUE_rexecd          6162 lo rexecd
AUE_passwd          6163 lo passwd
AUE_rexd            6164 lo rexd
AUE_ftpd            6165 lo ftp access
AUE_ftpd_logout     6171 lo ftp logout
AUE_ssh             6172 lo login - ssh
AUE_role_login      6173 lo role login
AUE_rad_login        6174 lo connect to RAD
AUE_newgrp_login     6212 lo newgrp login
AUE_admin_authenticate 6213 lo admin login
AUE_screenlock       6221 lo screenlock - lock
AUE_screenunlock    6222 lo screenlock - unlock
AUE_zlogin           6227 lo login - zlogin
AUE_su_logout        6228 lo su logout
AUE_role_logout      6229 lo role logout
```

```
AUE_smbd_session          6244 lo smbd(1m) session setup
AUE_smbd_logoff           6245 lo smbd(1m) session logoffAUE_sudo
        6650 lo,as,ua sudo(1M) execution
```

```
% auditconfig -lsevent | egrep " ex |,ex |ex,"
AUE_EXECVE                23 ex,ps execve(2)
AUE_PFEXEC                116, ex,ps,ua,as execve(2) with pfexec enabled
```

## 2. Audit the cusa class for administrators.

- **To audit these classes for administrative roles, modify the roles' security attributes.**

In the following example, root is a role. The site has created three roles, sysadm, auditadm, and netadm. All roles are audited for the success and failure of events in the cusa class.

```
# rolemod -K audit_flags=cusa:no root

# rolemod -K audit_flags=cusa:no sysadm

# rolemod -K audit_flags=cusa:no auditadm

# rolemod -K audit_flags=cusa:no netadm
```

- **To audit these classes for all users, set the system-wide flags.**

```
# auditconfig -setflags lo,ex
```

The output appears similar to the following:

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 12:17:12.616 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2486,50036632,82 0 mach1
return,success,0
```

## 3. Specify additional information to be recorded about command use.

- **To record the arguments to commands, add the argv policy.**

```
# auditconfig -setpolicy +argv
```

The exec\_args token records the command arguments. The following example wraps the lines for display purposes.

```
header,151,2,AUE_EXECVE,,mach1,2010-10-14 12:26:17.373 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args
```

```
,2,ls,/etc/security
subject,jdoe,root,root,root,root,2494,50036632,82 0 mach1
return,success,0
```

- **To record the environment in which the command is run, add the `exec_env` policy.**

```
# auditconfig -setpolicy +arge
```

The `exec_env` token records the command environment. The following example wraps lines for display purposes.

```
header,1460,2,AUE_EXECVE,,mach1,2010-10-14 12:29:39.679 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env
,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8,
PRINTER=example-dbl,...,_=/usr/bin/ls
subject,jdoe,root,root,root,root,2502,50036632,82 0 mach1
return,success,0
```

## ▼ How to Find Audit Records of Changes to Specific Files

If your goal is to log file writes against a limited number of files, such as `/etc/passwd` and the files in the `/etc/default` directory, you can use the `auditreduce` command to locate the files.

**Before You Begin** The root role can perform every task in this procedure.

If administrative rights are distributed in your organization, note the following:

- An administrator with the Audit Configuration rights profile can run the `auditconfig` command.
- An administrator with the Audit Review rights profile can run the `auditreduce` command.
- Only the root role can assign audit flags.

For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Perform one of the following steps to audit file changes.

- Audit the `fw` class.

Adding the `fw` class to the audit flags of a user or role generates fewer records than adding this class to the system-wide audit preselection mask. Perform one of the following steps:

- Add the fw class to specific roles.

```
# rolemod -K audit_flags=fw:no root
# rolemod -K audit_flags=fw:no sysadm
# rolemod -K audit_flags=fw:no auditadm
# rolemod -K audit_flags=fw:no netadm
```

- Add the fw class to the system-wide flags.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)

# auditconfig -setflags lo,fw
user default audit flags = lo,fw(0x1002,0x1002)
```

- Audit successful file-writes.

Auditing successes generates fewer records than auditing failures and successes. Perform one of the following steps:

- Add the +fw flag to specific roles.

```
# rolemod -K audit_flags=+fw:no root
# rolemod -K audit_flags=+fw:no sysadm
# rolemod -K audit_flags=+fw:no auditadm
# rolemod -K audit_flags=+fw:no netadm
```

- Add the +fw flag to the system-wide flags.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)

# auditconfig -setflags lo,+fw
user default audit flags = lo,+fw(0x1002,0x1000)
```

## 2. Obtain the audit records for specific files with the **auditreduce** command.

```
# auditreduce -o file=/etc/passwd,/etc/default -O filechg
```

The **auditreduce** command searches the audit trail for all occurrences of the **file** argument. The command creates a binary file with the suffix **filechg** which contains all records that include the path of the files of interest. See the [auditreduce\(1M\)](#) man page for the syntax of the **-o file=pathname** option.

## 3. Read the **filechg** file with the **praudit** command.

```
# praudit *filechg
```

## ▼ How to Update the Preselection Mask of Logged In Users

This procedure describes how to audit users who are already logged in for changes to the system-wide audit preselection mask. You can accomplish this task typically instructing the users to log out and to log back in. Alternatively, in a role that is assigned the Process Management rights profile, you can manually terminate active sessions with the `kill` command. The new sessions will inherit the new preselection mask.

However, terminating user sessions could be impractical. As an alternative, you can use the `auditconfig` command to dynamically change each logged-in user's preselection mask.

The following procedure assumes that you changed the system-wide audit preselection mask from `lo` to `lo,ex` by running the following command:

```
# auditconfig -setflags lo,ex
```

**Before You Begin** You must become an administrator who is assigned the Audit Configuration rights profile. To terminate user sessions, you must become an administrator who is assigned the Process Management rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. List the regular users who are logged in and their process IDs.

```
# who -a
jdoe - vt/2      Jan 25 07:56 4:10 1597 (:0)
jdoe + pts/1     Jan 25 10:10 .    1706 (:0.0)
...
jdoe + pts/2     Jan 25 11:36 3:41 1706 (:0.0)
```

### 2. For later comparison, display each user's preselection mask.

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = lo(0x1000,0x1000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

### 3. Modify the appropriate preselection mask by running one or more of the following commands:

```
# auditconfig -setpmask 1706 lo,ex          /* for this process */

# auditconfig -setumask jdoe lo,ex          /* for this user */

# auditconfig -setsmask 103203403 lo,ex     /* for this session */
```

### 4. Verify that the preselection mask for the user has changed.

For example, check a process that existed before you changed the mask.

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = ex,lo(0x40001000,0x40001000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

## ▼ How to Prevent the Auditing of Specific Events

For maintenance purposes, sometimes a site wants to prevent events from being audited.

---

**Note** - You cannot use this procedure on Solaris kernel events. However, you can use this procedure to allocate user level audit events with event numbers greater than 2047.

---

**Before You Begin** You must assume the root role. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Change the class of the event to the no class.

---

**Note** - For information about the effects of modifying an audit configuration file, see [“Audit Configuration Files and Packaging” on page 115](#).

---

For example, events 6214 and 6215 belong to the user administration, ua, class.

```
## audit_event file
...
6214:AUE_kadmind_auth:authenticated kadmind request:ua
6215:AUE_kadmind_unauth:unauthenticated kadmind req:ua
...
```

Change these events to the no class.

```
## audit_event file
...
6214:AUE_kadmind_auth:authenticated kadmind request:no
6215:AUE_kadmind_unauth:unauthenticated kadmind req:no
...
```

If the ua class is currently being audited, existing sessions will still audit events 6214 and 6215. To stop these events from being audited, you must update the users' preselection masks by following the instructions in [“How to Update the Preselection Mask of Logged In Users” on page 61](#).



**Caution** - Never comment out events in the `audit_event` file. This file is used by the `praudit` command to read binary audit files. Archived audit files might contain events that are listed in the file.

## 2. Refresh the kernel events.

```
# auditconfig -conf
Configured 283 kernel events.
```

# ▼ How to Compress Audit Files on a Dedicated File System

Audit files can grow large. You can set an upper limit to the size of a file, as shown in [Example 21, “Limiting File Size for the `audit\_binfile` Plugin,” on page 78](#). In this procedure, you use compression to reduce the size.

**Before You Begin** You must become an administrator who is assigned the ZFS File System Management and ZFS Storage Management rights profiles. The latter profile enables you to create storage pools. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

## 1. Dedicate a ZFS file system for audit files.

For the procedure, see [“How to Create ZFS File Systems for Audit Files” on page 74](#).

## 2. Compress the ZFS storage pool by using one of the following options.

With both options, the audit file system is compressed. After the audit service is refreshed, the compression ratio is displayed.

In the following examples, the ZFS pool `auditp/auditf` is the dataset.

### ■ Use the default compression algorithm.

```
# zfs set compression=on auditp/auditf
# audit -s
# zfs get compressratio auditp/auditf
NAME          PROPERTY      VALUE      SOURCE
auditp/auditf compressratio  4.54x      -
```

### ■ Use a higher compression algorithm.

```
# zfs set compression=gzip-9 auditp/auditf
# zfs get compression auditp/auditf
```

NAME	PROPERTY	VALUE	SOURCE
auditp/auditf	compression	gzip-9	local

The gzip-9 compression algorithm results in files that occupy one-third less space than the default compression algorithm, lzjb. For more information, see [Chapter 7, “Managing Oracle Solaris ZFS File Systems”](#) in *Managing ZFS File Systems in Oracle Solaris 11.3*.

**3. Refresh the audit service.**

```
# audit -s
```

**4. (Optional) Verify the new compression setting.**

For example, if you used the higher compression algorithm, the information would be similar to the following:

```
# zfs get compressratio auditp/auditf
NAME          PROPERTY      VALUE  SOURCE
auditp/auditf compressratio  16.89x  -
```

## ▼ How to Audit FTP and SFTP File Transfers

The FTP service creates logs of its file transfers. The SFTP service, which runs under the ssh protocol, can be audited by preselecting the ft audit class. Logins to both services can be audited.

---

**Note** - For information about how to log commands and file transfers of the FTP service, see the proftpd(8) man page.

For the available logging options, read [ProFTPD Logging \(http://www.proftpd.org/docs/howto/Logging.html\)](http://www.proftpd.org/docs/howto/Logging.html).

---

● **Perform one of the following depending on whether you want to audit SFTP or FTP.**

- To log sftp access and file transfers, edit the ft class.

The ft class includes the following SFTP transactions:

```
% auditrecord -c ft
file transfer: chmod ...
file transfer: chown ...
file transfer: get ...
file transfer: mkdir ...
file transfer: put ...
file transfer: remove ...
```



```
file transfer: rename ...
file transfer: rmdir ...
file transfer: session start ...
file transfer: session end ...
file transfer: symlink ...
file transfer: utimes
```

- To record access to the FTP server, audit the `lo` class.

As the following sample output indicates, logging in to and out of the `proftpd` daemon generates audit records.

```
% auditrecord -c lo | more
...
FTP server login
program      proftpd                See in.ftpd(1M)
event ID     6165                          AUE_ftpd
class        lo                    (0x0000000000001000)
header
subject
[text]                error message
return

FTP server logout
program      proftpd                See in.ftpd(1M)
event ID     6171                          AUE_ftpd_logout
class        lo                    (0x0000000000001000)
header
subject
return
...
```

## Configuring the Audit Service in Zones

The audit service audits the entire system, including audit events in zones. A system that has installed non-global zones can audit all zones identically, or can configure auditing per zone. For more information, see [“Planning Auditing in Zones” on page 28](#).

When you audit the non-global zones exactly as the global zone is audited, the non-global zone administrators might not have access to the audit records. Also, the global zone administrator can modify the audit preselection masks of users in non-global zones.

When you audit the non-global zones individually, the audit records are visible to the non-global zone and to the global zone from the non-global zone root.

## ▼ How to Configure All Zones Identically for Auditing

This procedure enables audits every zone identically. This method requires the least computer overhead and administrative resources.

**Before You Begin** You must assume the root role. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Configure the global zone for auditing.

Complete the tasks in [“Configuring the Audit Service” on page 42](#), with the following exceptions:

- Do not enable perzone audit policy.
- Set the zonename policy. This policy adds the name of the zone to every audit record.

```
# auditconfig -setpolicy +zonename
```

### 2. If you modified audit configuration files, copy them from the global zone to every non-global zone.

If you modified the `audit_class` or `audit_event` file, copy it in one of two ways:

- You can loopback mount the files.
- You can copy the files.

The non-global zone must be running.

- **Mount the changed `audit_class` and `audit_event` files as a loopback file system (`lofs`).**

#### a. From the global zone, halt the non-global zone.

```
# zoneadm -z non-global-zone halt
```

#### b. Create a read-only loopback mount for every audit configuration file that you modified in the global zone.

```
# zonecfg -z non-global-zone
zone: add fs
zone/fs: set special=/etc/security/audit-file
zone/fs: set dir=/etc/security/audit-file
zone/fs: set type=lofs
zone/fs: add options [ro,nodevices,nosetuid]
zone/fs: commit
zone/fs: end
zone: exit
```

```
#
```

**c. To make the changes effective, boot the non-global zone.**

```
# zoneadm -z non-global-zone boot
```

Later, if you modify an audit configuration file in the global zone, you reboot each zone to refresh the loopback-mounted files in the non-global zones.

■ **Copy the files.**

**a. From the global zone, list the /etc/security directory in each non-global zone.**

```
# ls /zone/zonename/root/etc/security/
```

**b. Copy the changed audit\_class and audit\_event files to each zone's /etc/security directory.**

```
# cp /etc/security/audit-file /zone/zonename/root/etc/security/audit-file
```

Later, if you change one of these files in the global zone, you must copy the changed file to the non-global zones.

**Example 17** Mounting Audit Configuration Files as Loopback Mounts in a Zone

In this example, the system administrator has modified the `audit_class`, `audit_event`, and `audit_warn` files.

The `audit_warn` file is read in the global zone only, so does not have to be mounted into the non-global zones.

On this system, `machine1`, the administrator has created two non-global zones, `machine1-webserver` and `machine1-appserver`. The administrator has finished modifying the audit configuration files. If the administrator later modifies the files, the zone must be rebooted to re-read the loopback mounts.

```
# zoneadm -z machine1-webserver halt
# zoneadm -z machine1-appserver halt
# zonecfg -z machine1-webserver
webserver: add fs
webserver/fs: set special=/etc/security/audit_class
webserver/fs: set dir=/etc/security/audit_class
webserver/fs: set type=lofs
webserver/fs: add options [ro,nodevices,nosetuid]
webserver/fs: commit
webserver/fs: end
webserver: add fs
```

```
webserver/fs: set special=/etc/security/audit_event
webserver/fs: set dir=/etc/security/audit_event
webserver/fs: set type=lofs
webserver/fs: add options [ro,nodevices,nosetuid]
webserver/fs: commit
webserver/fs: end
webserver: exit
#

# zonecfg -z machine1-appserver
appserver: add fs
appserver/fs: set special=/etc/security/audit_class
appserver/fs: set dir=/etc/security/audit_class
appserver/fs: set type=lofs
appserver/fs: add options [ro,nodevices,nosetuid]
appserver/fs: commit
appserver/fs: end
appserver: exit
```

When the non-global zones are rebooted, the `audit_class` and `audit_event` files are read-only in the zones.

## ▼ How to Configure Per-Zone Auditing

This procedure enables separate zone administrators to control the audit service in their zone. For the complete list of policy options, see the [auditconfig\(1M\)](#) man page.

**Before You Begin** To configure auditing, you must become an administrator who is assigned the Audit Configuration rights profile. To enable the audit service, you must become an administrator who is assigned the Audit Control rights profile. For more information, see “[Using Your Assigned Administrative Rights](#)” in *Securing Users and Processes in Oracle Solaris 11.3*.

1. In the global zone, configure auditing.
  - a. Complete the tasks in “[Configuring the Audit Service](#)” on page 42.
  - b. Add the `perzone` audit policy. For the command, see [Example 12, “Setting the perzone Audit Policy,”](#) on page 50.

---

**Note** - You are not required to enable the audit service in the global zone.

---

2. In each non-global zone that you plan to audit, configure the audit files.
  - a. Complete the tasks in “[Configuring the Audit Service](#)” on page 42.

**b. Do not configure system-wide audit settings.**

Specifically, do not add the `perzone` or `ahlt` policy to the non-global zone.

**3. Enable auditing in your zone.**

```
myzone# audit -s
```

**Example 18** Disabling Auditing in a Non-Global Zone

This example works if the `perzone` audit policy is set. The zone administrator of the `noaudit` zone disables auditing for that zone.

```
noauditzone # auditconfig -getcond
audit condition = auditing
noauditzone # audit -t
noauditzone # auditconfig -getcond
audit condition = noaudit
```

## Example: Configuring Oracle Solaris Auditing

This section provides an example of how you configure and implement Oracle Solaris auditing. It begins with the configuration of different attributes of the service according to specific needs and requirements. After configuration is completed, the audit service is started to effect the configuration settings. Each time that you need to revise an existing audit configuration to accommodate new requirements, follow the same sequence of actions in this example:

1. Configure the audit parameters.
  2. Refresh the audit service.
  3. Verify the new audit configuration.
- First, the administrator adds a temporary policy.

```
# auditconfig -t -setpolicy +zonename
# auditconfig -getpolicy
configured audit policies = ahl,t,ar,g,ar,gv,perzone
active audit policies = ahl,t,ar,g,ar,gv,perzone,zonename
```

- Then, the administrator specifies queue controls.

```
# auditconfig -setqctrl 200 20 0 0
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
```

```
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

- Then, the administrator specifies plugin attributes.
  - For the audit\_binfile plugin, the administrator removes the qsize value.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/audit/sys1.1,/var/audit;
p_minfree=2;p_fsize=4G;
Queue size: 200
# auditconfig -setplugin audit_binfile "" 0
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/audit/sys1.1,/var/audit
p_minfree=2;p_fsize=4G;
```

- For the audit\_syslog plugin, the administrator specifies that successful login and logout events and failed executables be sent to syslog. The qsize for this plugin is set to 150.

```
# auditconfig -setplugin audit_syslog active p_flags=+lo,-ex 150
# auditconfig -getplugin audit_syslog
auditconfig -getplugin audit_syslog
Plugin: audit_syslog
Attributes: p_flags=+lo,-ex;
Queue size: 150
```

- The administrator does not configure or use the audit\_remote plugin.
- Then, the administrator refreshes the audit service and verifies the configuration.
  - The temporary zonename policy is no longer set.

```
# audit -s
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv,perzone
active audit policies = ahlt,arge,argv,perzone
```

- The queue controls remain the same.

```
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
```

```
active audit queue buffer size (bytes) = 8192
```

```
active audit queue delay (ticks) = 20
```

- The `audit_binfile` plugin does not have a specified queue size. The `audit_syslog` plugin has a specified queue size.

```
# auditconfig -getplugin
```

```
Plugin: audit_binfile
```

```
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
```

```
Plugin: audit_syslog
```

```
Attributes: p_flags=+lo,-ex;
```

```
Queue size: 150
```

```
...
```





# ◆◆◆ CHAPTER 4

## Monitoring System Activities

---

This chapter provides procedures to help you configure audit logs that enable to you to monitor activities in the system. In addition, the following chapters describe other audit management tasks:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 5, “Working With Audit Data”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

For an overview of the audit service, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For reference information, see [Chapter 7, “Auditing Reference”](#).

## Configuring Audit Logs

Two audit plugins, `audit_binfile` and `audit_syslog`, can create local audit logs. The following tasks explain how to configure these logs.

## Configuring Audit Logs

The following task map points to the procedures for configuring audit logs for the various plugins. Configuring logs for the `audit_binfile` plugin is optional. The logs for other plugins must be configured by an administrator.

**TABLE 6**      Configuring Audit Logs Task Map

Task	Description	For Instructions
Add local storage for the <code>audit_binfile</code> plugin	Creates additional disk space for the audit files and protects them with file permissions	<a href="#">“How to Create ZFS File Systems for Audit Files” on page 74</a>
Assign storage for the <code>audit_binfile</code> plugin	Identifies directories for binary audit records	<a href="#">“How to Assign Audit Space for the Audit Trail” on page 77</a>

Task	Description	For Instructions
Configure streaming audit records to a remote system	Enables you to send audit records to a remote repository through a protected mechanism	<a href="#">“How to Send Audit Files to a Remote Repository” on page 80</a>
Configure remote storage for audit files	Enables you to receive audit records on a remote system	<a href="#">“How to Configure a Remote Repository for Audit Files” on page 82</a>
Configure storage for the audit_syslog plugin.	Enables you to stream audit events in text format to syslog.	<a href="#">“How to Configure syslog Audit Logs” on page 86</a>

## ▼ How to Create ZFS File Systems for Audit Files

The following procedure shows how to create a ZFS pool for audit files, as well as the corresponding file systems and mount point. By default, `/var/audit` holds audit files for the `audit_binfile` plugin.

**Before You Begin** You must become an administrator who is assigned the ZFS File System Management and ZFS Storage Management rights profiles. The latter profile enables you to create storage pools. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Determine the amount of disk space that is required.

How much auditing you require dictates the disk space requirements.

---

**Note** - The default class preselection creates files in `/var/audit` that grow by about 80 bytes for every recorded instance of an event in the `lo` class, such as a login, logout, or role assumption.

---

### 2. Create a mirrored ZFS storage pool.

The `zpool create` command creates a storage pool, that is, a container for the ZFS file systems. For more information, see [Chapter 1, “Introducing the Oracle Solaris ZFS File System” in \*Managing ZFS File Systems in Oracle Solaris 11.3\*](#).

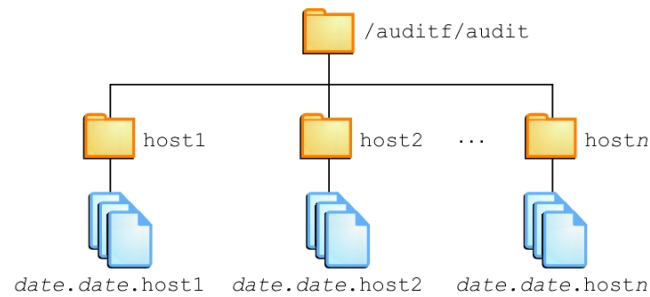
```
# zpool create audit-pool mirror disk1 disk2
```

For example, create the `auditp` pool from two disks, `c3t1d0` and `c3t2d0`, and mirror them.

```
# zpool create auditp mirror c3t1d0 c3t2d0
```

### 3. Create a ZFS file system and mount point for the audit files.

You create the file system and mount point with one command. At creation, the file system is mounted. For example, the following illustration shows audit trail storage that is stored by host name.



```
# zfs create -o mountpoint=/mountpoint audit-pool/mountpoint
```

For example, create the /audit mount point for the auditf file system.

```
# zfs create -o mountpoint=/audit auditp/auditf
```

#### 4. Create a ZFS file system for the audit files.

```
# zfs create -p auditp/auditf/system
```

For example, create an unencrypted ZFS file system for the sys1 system.

```
# zfs create -p auditp/auditf/sys1
```

#### 5. (Optional) Create additional file systems for audit files.

One reason to create additional file systems is to prevent audit overflow. You can set a ZFS quota per file system, as shown in [Step 7](#). The `audit_warn` email alias notifies you when each quota is reached. To free space, you can move the closed audit files to a remote server.

```
# zfs create -p auditp/auditf/sys1.1
```

```
# zfs create -p auditp/auditf/sys1.2
```

#### 6. Compress the audit files in the pool.

Typically, compression is set in ZFS at the file system level. However, because all the file systems in this pool contain audit files, compression is set at the top-level dataset for the pool.

```
# zfs set compression=on auditp
```

See also [“Interactions Between ZFS Compression, Deduplication, and Encryption Properties” in \*Managing ZFS File Systems in Oracle Solaris 11.3\*](#).

## 7. Set quotas.

You can set quotas at the parent file system, the descendant file systems, or both. If you set a quota on the parent audit file system, quotas on the descendant file systems impose an additional limit.

### a. Set a quota on the parent audit file system.

In the following example, when both disks in the `auditp` pool reach the quota, the `audit_warn` script notifies the audit administrator.

```
# zfs set quota=510G auditp/auditf
```

### b. Set a quota on the descendant audit file systems.

In the following example, when the quota for the `auditp/auditf/system` file system is reached, the `audit_warn` script notifies the audit administrator.

```
# zfs set quota=170G auditp/auditf/sys1
```

```
# zfs set quota=170G auditp/auditf/sys1.1
```

```
# zfs set quota=165G auditp/auditf/sys1.2
```

## 8. For a large pool, limit the size of the audit files.

By default, an audit file can grow to the size of the pool. For manageability, limit the size of the audit files. See [Example 21, “Limiting File Size for the audit\\_binfile Plugin,” on page 78](#).

### Example 19 Creating an Encrypted File System for Audit Archiving

To comply with site security requirements, the administrator performs the following steps:

1. Creates, if necessary, a new ZFS pool to store the encrypted audit logs.
2. Generates an encryption key.
3. Creates the audit file system with encryption turned on to store the audit logs, as well as sets the mount point.
4. Configures auditing to use the encrypted directory.
5. Refreshes the audit service to apply the new configuration settings.

```
# zpool create auditp mirror disk1 disk2
```

```
# pktool genkey keystore=file outkey=/filename keytype=aes keylen=256
```

```
# zfs create -o encryption=aes-256-ccm \  
-o keysource=raw,file:///filename \
```

```
-o compression=on -o mountpoint=/audit auditp/auditf

# auditconfig -setplugin audit_binfile p_dir=/audit/

# audit -s
```

You must back up and protect the file where the key is stored, such as *filename* in the example.

When the administrator creates additional file systems under the `auditf` file system, these descendant file systems are also encrypted.

**Example 20** Setting a Quota on `/var/audit`

In this example, the administrator sets a quota on the default audit file system. When this quota is reached, the `audit_warn` script warns the audit administrator.

```
# zfs set quota=252G rpool/var/audit
```

## ▼ How to Assign Audit Space for the Audit Trail

In this procedure, you use attributes to the `audit_binfile` plugin to assign additional disk space to the audit trail.

**Before You Begin** You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### 1. Determine the attributes to the `audit_binfile` plugin.

Read the OBJECT ATTRIBUTES section of the `audit_binfile(5)` man page.

```
# man audit_binfile
```

```
...
OBJECT ATTRIBUTES
The p_dir attribute specifies where the audit files will be created.
The directories are listed in the order in which they are to be used.
```

The `p_minfree` attribute defines the percentage of free space that the audit system requires before the audit daemon invokes the `audit_warn` script.

The `p_fsize` attribute defines the maximum size that an audit file can become before it is automatically closed and a new audit file is opened. ... The format of the `p_fsize` value can be specified as an exact value in bytes or in a human-readable form with a suffix of B, K, M, G, T, P, E, Z (for bytes, kilobytes, megabytes, gigabytes, terabytes, petabytes, exabytes,

or zettabytes, respectively). Suffixes of KB, MB, GB, TB, PB, EB, and ZB are also accepted.

**2. To add directories to the audit trail, specify the `p_dir` attribute.**

`/var/audit` functions as the default file system..

```
# auditconfig -setplugin audit_binfile p_dir=/audit/sys1.1,/var/audit
```

The preceding command sets the `/audit/sys1.1` file system as the primary directory for audit files and the default `/var/audit` as the secondary directory. In this scenario, `/var/audit` functions as the directory of last resort. For this configuration to succeed, the `/audit/sys1.1` file system must exist.

A similar file system is created in [“How to Create ZFS File Systems for Audit Files” on page 74](#).

**3. Refresh the audit service.**

The `auditconfig -setplugin` command sets the *configured* value. This value is a property of the audit service, so it is restored when the service is refreshed or restarted. The configured value becomes *active* when the audit service is refreshed or restarted. For information about configured and active values, see the [auditconfig\(1M\)](#) man page.

```
# audit -s
```

**Example 21** Limiting File Size for the `audit_binfile` Plugin

In the following example, the size of a binary audit file is set to a specific size. The size is specified in megabytes.

```
# auditconfig -setplugin audit_binfile p_fsize=4M
```

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_age=0h;p_dir=/var/audit;p_minfree=1;p_fsize=4M;
```

By default, an audit file can grow without limit. To create smaller audit files, the administrator specifies a file size limit of 4 MB. The audit service creates a new file when the size limit is reached. The file size limit goes into effect after the administrator refreshes the audit service.

```
# audit -s
```

**Example 22** Specifying Time for Log Rotation

In the following example, a time limit is set for an audit file. The time limit is specified in terms of hours, days, weeks, months, or years.

```
# auditconfig -setplugin audit_binfile p_age=1w
```

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_minfree=1;p_fsize=4M;p_age=1w;
Queue size: 200
```

By default, an audit file has no time limit. The file remains open indefinitely until an external operation causes a file rotation. The administrator sets the file's time limit to one week, beyond which a new audit file is opened. To implement the new time limit, the administrator refreshes the audit service.

```
# audit -s
```

#### Example 23 Specifying Several Changes to an Audit Plugin

In the following example, the administrator on a system with high throughput and a large ZFS pool changes the queue size, the binary file size, and the soft limit warning for the `audit_binfile` plugin. The administrator allows audit files to grow to 4 GB, is warned when 2 percent of the ZFS pool remains, and doubles the allowed queue size. The default queue size is the high water mark for the kernel audit queue, 100, as in `active audit queue hiwater mark (records) = 100`. The audit file is also set to have a time limit of 2 weeks.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=2G;p_minfree=1;

# auditconfig -setplugin audit_binfile \
    "p_minfree=2;p_fsize=4G;p_age=2w" 200

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;p_age=2w;
Queue size: 200
```

The changed specifications go into effect after the administrator refreshes the audit service.

```
# audit -s
```

#### Example 24 Removing Queue Size for an Audit Plugin

In the following example, the queue size for the `audit_binfile` plugin is removed.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
Queue size: 200

# auditconfig -setplugin audit_binfile "" 0

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
```

```
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
```

The empty quotation marks ("" ) retain the current attribute values. The final 0 sets the queue size for the plugin to the default.

The change in qsize specification for the plugin goes into effect after the administrator refreshes the audit service.

```
# audit -s
```

#### **Example 25** Setting a Soft Limit for Warnings

In this example, the minimum free-space level for all audit file systems is set so that a warning is issued when two percent of the file system is still available.

```
# auditconfig -setplugin audit_binfile p_minfree=2
```

The default percentage is one (1). For a large ZFS pool, choose a reasonably low percentage. For example, 10 percent of a 16 TB pool is around 16 GB, which would warn the audit administrator when plenty of disk space remains. A value of 2 sends the audit\_warn message when about two GB of disk space remains.

The audit\_warn email alias receives the warning. To set up the alias, see [“How to Configure the audit\\_warn Email Alias” on page 53](#).

For a large pool, the administrator also limits the file size to 3 GB.

```
# auditconfig -setplugin audit_binfile p_fsize=3G
```

The p\_minfree and p\_fsize specifications for the plugin go into effect after the administrator refreshes the audit service.

```
# audit -s
```

## ▼ How to Send Audit Files to a Remote Repository

In this procedure, you use attributes of the audit\_remote plugin to send the audit trail to a remote audit repository. To configure a remote repository on an Oracle Solaris system, see [“How to Configure a Remote Repository for Audit Files” on page 82](#).

**Before You Begin** You must have a receiver of audit files at your remote repository. You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

### **1. Determine the attributes of the audit\_remote plugin.**

Read the OBJECT ATTRIBUTES section of the [audit\\_remote\(5\)](#) man page.



```
# man audit_remote
```

```
...
```

#### OBJECT ATTRIBUTES

The `p_hosts` attribute specifies the remote servers. You can also specify the port number and the GSS-API mechanism.

The `p_retries` attribute specifies the number of retries for connecting and sending data. The default is 3.

The `p_timeout` attribute specifies the number of seconds in which a connection times out.

The default port is the `solaris_audit` IANA-assigned port, 16162/tcp. The default mechanism is `kerberos_v5`. The timeout default is 5 seconds. You can also specify a queue size for the plugin.

### 2. To specify the remote receiving system, use the `p_hosts` attribute.

In this example, the receiving system uses a different port.

```
# auditconfig -setplugin audit_remote \
    p_hosts=ars.example.com:16088:kerberos_v5
```

### 3. Specify other attributes of the plugin that you want to change.

For example, the following command specifies values for all optional attributes:

```
# auditconfig -setplugin audit_remote "p_retries=;p_timeout=3" 300
```

### 4. Verify the values, then activate the plugin.

For example, the following commands specify and verify the values of the plugin:

```
# auditconfig -getplugin audit_remote
Plugin: audit_remote (inactive)
Attributes: p_hosts=ars.example.com:16088:kerberos_v5;p_retries=5;p_timeout=3;
Queue size: 300
```

```
# auditconfig -setplugin audit_remote active
```

### 5. Refresh the audit service.

The audit service reads the audit plugin change upon refresh.

```
# audit -s
```

## Example 26 Tuning the Audit Queue Buffer Size

In this example, the audit queue is full behind the `audit_remote` plugin. This audited system is configured to audit many classes and is transmitting across a high-traffic, slow network. The

administrator enlarges the plugin's buffer size to enable the audit queue to grow and not exceed the buffer's limit before records are removed from the queue.

```
audsys1 # auditconfig -setplugin audit_remote "" 1000
```

```
audsys1 # audit -s
```

## ▼ How to Configure a Remote Repository for Audit Files

In this procedure, you configure a remote system, the Audit Remote Server (ARS), to receive and store audit records from one or more audited systems. Then, you activate the audit daemon on the remote server.

The configuration is twofold. First, you configure the underlying security mechanisms to securely transport the audit data, that is, you configure the KDC. Second, you configure the audit service on both the audited system and the ARS. This procedure illustrates a scenario with one audited client and one ARS, where the ARS and the KDC are on the same server. More complex scenarios can be configured similarly. The first four steps describe the configuration of the KDC, while the final step describes the configuration of the audit service.

**Before You Begin** Ensure that you have completed the following:

- You have assumed the root role.
- You have installed the Kerberos packages, as described in [“How to Prepare to Stream Audit Records to Remote Storage” on page 33](#).
- You are working with an administrator who has configured the audited system, as described in [“How to Send Audit Files to a Remote Repository” on page 80](#).

### 1. If your site has not yet configured a KDC, configure one.

You need a KDC on a system that both the audited system and the ARS can use, a host principal for each system, and an audit service principal. The following example illustrates a KDC configuration strategy:

```
arstore # kdcmgr -a audr/admin -r EXAMPLE.COM create master
```

This command uses the administrative principal `audr/admin` to create a master KDC in the `EXAMPLE.COM` realm, enables the master KDC, and starts the Kerberos service.

### 2. Verify that the KDC is available.

For more information, see the [kdcmgr\(1M\)](#) man page.

```
# kdcmgr status
```

## KDC Status Information

```
-----
svc:/network/security/krb5kdc:default (Kerberos key distribution center)
State: online since Wed Feb 29 01:59:27 2012
See: man -M /usr/share/man -s 1M krb5kdc
See: /var/svc/log/network-security-krb5kdc:default.log
Impact: None.
```

## KDC Master Status Information

```
-----
svc:/network/security/kadmin:default (Kerberos administration daemon)
State: online since Wed Feb 29 01:59:28 2012
See: man -M /usr/share/man -s 1M kadmind
See: /var/svc/log/network-security-kadmin:default.log
Impact: None.
```

## Transaction Log Information

```
-----
Kerberos update log (/var/krb5/principal.ulong)
Update log dump :
Log version # : 1
Log state : Stable
Entry block size : 2048
Number of entries : 13
First serial # : 1
Last serial # : 13
First time stamp : Wed Feb 29 01:59:27 2012
Last time stamp : Mon Mar 5 19:29:28 2012
```

## Kerberos Related File Information

```
-----
(Displays any missing files)
```

### 3. Add the audit service principal to the KDC keytab file.

You can add the principal by typing the `kadmin.local` command on the KDC system. Or, you can remotely add the principal by using the `kadmin` command and providing a password. In this example, the `arstore` system is running the KDC.

```
# kadmin -p audr/admin
```

```
kadmin: addprinc -randkey audit/arstore.example.com@EXAMPLE.COM
```

```
kadmin: ktadd audit/arstore.example.com@EXAMPLE.COM
```

### 4. On each audited system, add keys.

The receiver and the sender must have keys.

```
enigma # kclient
```

```
.. Enter the Kerberos realm:
EXAMPLE.COM

.. KDC hostname for the above realm:
arstore.example.com

.. Will this client need service keys ? [y/n]:
y
```

## 5. Configure the audit service on the ARS.

- **Create and name a connection group that accepts audit records from any audited system in the Kerberos realm.**

```
# auditconfig -setremote group create Bank_A
```

Bank\_A is a connection group. Because the hosts attribute is not defined, this group accepts all connections, which means that it is a *wildcard* group. Any audited system in this Kerberos realm whose audit\_remote plugin is correctly configured can reach this ARS.

- **To limit connections to this group, specify the audited systems that can use this repository.**

```
# auditconfig -setremote group Bank_A "hosts=enigma.example.com"
```

Connection group Bank\_A now accepts only connections from the enigma system. A connection from any other host is refused.

- **To prevent an audit file in this group from growing too large, set a maximum size.**

```
# auditconfig -setremote group Bank_A "binfile_fsize=4GB"
```

```
# auditconfig -getremote
Audit Remote Server
Attributes: listen_address=;login_grace_time=30;max_startups=10;listen_port=0;
Connection group: Bank_A (inactive)
Attributes: binfile_dir=/var/audit;binfile_fsize=4GB;binfile_minfree=1;
hosts=enigma.example.com;
```

## 6. Configure the audit service on the audited system.

To specify the ARS, use the p\_hosts attribute.

```
enigma # auditconfig -setplugin audit_remote \
        active p_hosts=arstore.example.com

enigma # auditconfig -getplugin audit_remote
```

```
Plugin: audit_remote
Attributes: p_retries=3;p_timeout=5;p_hosts=arstore.example.com;
```

## 7. Refresh the audit service.

The audit service reads the audit plugin change upon refresh.

```
# audit -s
```

The KDC now manages the connection between the audited system *enigma* and the ARS.

### Example 27 Streaming Audit Records to Different File Locations on the Same ARS

This example extends the example in the procedure. The administrator separates audit records by host on the ARS by creating two connection groups.

Audit files from *audsys1* stream to the *Bank\_A* connection group on this ARS.

```
arstore # auditconfig -setremote group create Bank_A

arstore # auditconfig -setremote group active Bank_A "hosts=audsys1" \
"hosts=audsys1;binfile_dir=/var/audit/audsys1;binfile_fsize=4M;"
```

Audit files from *audsys2* stream to the *Bank\_B* connection group.

```
arstore # auditconfig -setremote group create Bank_B

arstore # auditconfig -setremote group active Bank_B \
"hosts=audsys2;binfile_dir=/var/audit/audsys2;binfile_fsize=4M;"
```

For easier maintenance, the administrator sets other attribute values identically.

```
arstore # auditconfig -getremote
Audit Remote Server
Attributes: listen_address=;login_grace_time=30;max_startups=10;listen_port=0;

Connection group: Bank_A
Attributes: binfile_dir=/var/audit/audsys1;binfile_fsize=4M;binfile_minfree=1;
hosts=audsys1

Connection group: Bank_B
Attributes: binfile_dir=/var/audit/audsys2;binfile_fsize=4M;binfile_minfree=1;
hosts=audsys2
```

### Example 28 Placing the ARS on a Different System From the KDC

In this example, the administrator places the ARS on a different system from the KDC. First, the administrator creates and configures the master KDC.

```
kserv # kdcmgr -a audr/admin -r EXAMPLE.COM create master
```

```
kserve # kadmin.local -p audr/admin

kadmin: addprinc -randkey \
audit/arstore.example.com@EXAMPLE.COM

kadmin: ktadd -k /var/user/root/krb5.keytab.audit \
audit/arstore.example.com@EXAMPLE.COM
```

After securely transmitting the /tmp/krb5.keytab.audit file to the ARS, arstore, the administrator moves the file to the correct location.

```
arstore # chown root:root krb5.keytab.audit

arstore # chmod 600 krb5.keytab.audit

arstore # mv krb5.keytab.audit /etc/krb5/krb5.keytab
```

Rather than rewrite the file, the administrator also has the option to use the ktutil command on the ARS to merge the KDC krb5.keytab.audit file with existing keys in the arstore's /etc/krb5/krb5.keytab file.

Finally, the administrator generates keys on the audited system.

```
enigma # kclient

.. Enter the Kerberos realm: EXAMPLE.COM

.. KDC hostname for the above realm: kserve.example.com

.. Will this client need service keys ? [y/n]: y
```

## ▼ How to Configure syslog Audit Logs

You can instruct the audit service to copy some or all of the audit records in the audit queue to the syslog utility. If you record both binary audit data and text summaries, the binary data provide a complete audit record, while the summaries filter the data for real-time review.

**Before You Begin** To configure the audit\_syslog plugin, you must become an administrator who is assigned the Audit Configuration rights profile. To configure the syslog utility and create the auditlog file, you must assume the root role.

1. **Select audit classes to be sent to the audit\_syslog plugin, and make the plugin active.**

---

**Note** - p\_flags audit classes must be preselected as either system defaults or in the audit flags of a user or a rights profile. Records are not collected for a class that is not preselected.

---

```
# auditconfig -setplugin audit_syslog \
    active p_flags=lo,+as,-ss
```

## 2. Configure the syslog utility.

### a. Add an audit.notice entry to the syslog.conf file.

The entry includes the location of the log file.

```
# cat /etc/syslog.conf
```

```
...
audit.notice      /var/adm/auditlog
```

### b. Create the log file.

```
# touch /var/adm/auditlog
```

### c. Set the log file's permissions to 640.

```
# chmod 640 /var/adm/auditlog
```

### d. Check which system-log service instance is running on the system.

```
# svcs system-log
```

STATE	STIME	FMRI
online	Nov_27	svc:/system/system-log:default
disabled	Nov_27	svc:/system/system-log:rsyslog

### e. Refresh the configuration information for the active syslog service instance.

```
# svcadm refresh system/system-log:default
```

## 3. Refresh the audit service.

The audit service reads the changes to the audit plugin upon refresh.

```
# audit -s
```

## 4. Regularly archive the syslog log files.

The audit service can generate extensive output. To manage the logs, see the [logadm\(1M\)](#) man page.

### Example 29 Specifying Audit Classes for syslog Output

In the following example, the syslog utility collects a subset of the preselected audit classes. The pf class is created in [Example 15, “Creating a New Audit Class,”](#) on page 54.

```
# auditconfig -setnaflags lo,na
# auditconfig -setflags lo,ss
# usermod -K audit_flags=pf:no jdoe
# auditconfig -setplugin audit_syslog \
    active p_flags=lo,+na,-ss,+pf
```

The arguments to the `auditconfig` command instruct the system to collect all login/logout, non-attributable, and change of system state audit records. The `audit_syslog` plugin entry instructs the `syslog` utility to collect all logins, successful non-attributable events, and failed changes of system state.

For the `jdoe` user, the binary utility collects successful and failed calls to the `pfexec` command. The `syslog` utility collects successful calls to the `pfexec` command.

**Example 30** Putting `syslog` Audit Records on a Remote System

You can change the `audit.notice` entry in the `syslog.conf` file to point to a remote system. In this example, the name of the local system is `sys1.1`. The remote system is `remote1`.

```
sys1.1 # cat /etc/syslog.conf
```

```
...
audit.notice      @remote1
```

The `audit.notice` entry in the `syslog.conf` file on the `remote1` system points to the log file.

```
remote1 # cat /etc/syslog.conf
```

```
...
audit.notice      /var/adm/auditlog
```



## Working With Audit Data

---

This chapter provides procedures to help you with audit data that are generated from different local systems. This chapter covers the following topics:

- [“Displaying Audit Trail Data” on page 89](#)
- [“Managing Audit Records on Local Systems” on page 97](#)

In addition, the following chapters describe other audit management tasks:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

For an overview of the audit service, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For reference information, see [Chapter 7, “Auditing Reference”](#).

### Displaying Audit Trail Data

The default plugin, `audit_binfile`, creates an audit trail. The trail can contain large amounts of data. The following sections describe how to work with this data.

### Displaying Audit Record Definitions

To display audit record definitions, use the `auditrecord` command. The definitions provide the audit event number, audit class, selection mask, and record format of an audit event.

% **auditrecord** *-options*

The screen output generated by the command depends on the option that you use, as shown in the following partial list.

- The `-p` option displays the audit record definitions of a program.

- The -c option displays the audit record definitions of an audit class.
- The -a option lists all audit event definitions.

You can also the print displayed output to a file.

For more information, see the [auditrecord\(1M\)](#) man page.

**EXAMPLE 31** Displaying the Audit Record Definitions of a Program

In this example, the definition of all audit records that are generated by the login program are displayed. Login programs include rlogin, telnet, newgrp, and the Secure Shell feature of Oracle Solaris.

```
% auditrecord -p login
...
login: logout
program      various          See login(1)
event ID     6153              AUE_logout
class        lo               (0x0000000000001000)
...
newgrp
program      newgrp           See newgrp login
event ID     6212              AUE_newgrp_login
class        lo               (0x0000000000001000)
...
rlogin
program      /usr/sbin/login     See login(1) - rlogin
event ID     6155              AUE_rlogin
class        lo               (0x0000000000001000)
...
/usr/lib/ssh/sshd
program      /usr/lib/ssh/sshd  See login - ssh
event ID     6172              AUE_ssh
class        lo               (0x0000000000001000)
...
telnet login
program      /usr/sbin/login     See login(1) - telnet
event ID     6154              AUE_telnet
class        lo               (0x0000000000001000)
...
```

**EXAMPLE 32** Displaying the Audit Record Definitions of an Audit Class

In this example, the definitions of all audit records in the pf class that was created in [Example 15, “Creating a New Audit Class,”](#) on page 54 is displayed.

```
% auditrecord -c pf
```

```

pfexec
system call pfexec          See execve(2) with pfexec enabled
event ID    116             AUE_PFEXEC
class       pf              (0x0100000000000000)
header
path         pathname of the executable
path         pathname of working directory
[privileges] privileges if the limit or inheritable set are changed
[privileges] privileges if the limit or inheritable set are changed
[process]    process if ruid, euid, rgid or egid is changed
exec_arguments
[exec_environment] output if arge policy is set
subject
[use_of_privilege]
return

```

The `use_of_privilege` token is recorded whenever privilege is used. The `privileges` tokens are recorded if the limit or inheritable set is changed. The process token is recorded if an ID is changed. No policy option is required for these tokens to be included in the record.

#### EXAMPLE 33 Printing Audit Record Definitions to a File

In this example, the `-h` option is added to put all the audit record definitions to a file in HTML format. When you display the HTML file in a browser, use the browser's Find tool to find specific audit record definitions.

```
% auditrecord -ah > audit.events.html
```

## Selecting Audit Events to Be Displayed

As an administrator who is assigned the Audit Review rights profile, you can filter audit records for examination by using the `auditreduce` command. This command can eliminate the less interesting records as it combines the input files.

```
auditreduce -option argument [optional-file]
```

where *argument* is the specific argument that an option requires.

The following is a partial list of *record selection* options and their corresponding arguments:

- c                      Selects an audit class where *argument* is an audit class, such as `ua`.
- d                      Selects all of the events on a particular date. The date format for *argument* is `yyymmdd`. Other date options such as `-b` and `-a` select events before and after a particular date respectively.

<code>-u</code>	Selects all of the events attributable to a particular user. For this option, you specify a user name. Another user option, <code>-e</code> , selects all of the events attributable to an effective user ID.
<code>-g</code>	Selects all of the events attributable to a particular group. For this option, specify a group name.
<code>-m</code>	Selects all of the instances of a particular audit event.
<code>-o</code>	Selects by object type. Use this option to select by file, group, file owner, FMRI, PID, and other object types.
<i>optional-file</i>	The name of an audit file.

The command also uses *file selection* options which are all in upper case as shown in the following examples. For the full list of options, see the [auditreduce\(1M\)](#) man page.

**EXAMPLE 34** Combining and Reducing Audit Files

In this example, only the login and logout records in audit files that are over a month old are retained. The example assumes that the current date is Sept 27. If you need to retrieve the complete audit trail, you could recover the trail from backup media. The `-O` option directs the command's output to a file named `lo.summary`.

```
# cd /var/audit/audit_summary
# auditreduce -O lo.summary -b 20100827 -c lo; compress *lo.summary
```

**EXAMPLE 35** Copying One User's Audit Records to a Summary File

In this example, the records in the audit trail that contain the name of a particular user are merged. The `-e` option finds the effective user. The `-u` option finds the login user. The `-O` option directs the output to the file `tamiko`.

```
# cd /var/audit/audit_summary
# auditreduce -e tamiko -O tamiko
```

You can further narrow the displayed information. In this next example, the following are filtered and printed to a file called `tamiko.lo`.

- Time of user login and logout, specified by the `-c` option.
- Date of Sept 7, 2013, specified by the `-d` option. The short form of the date is `yyymmdd`.
- User name of `tamiko`, specified by the `-u` option.
- Name of physical machine, specified by the `-M` option.

```
# auditreduce -M tamiko -O tamikolo -d 20130907 -u tamiko -c lo
```

#### EXAMPLE 36 Merging Selected Records to a Single File

In this example, login and logout records for a particular day are selected from the audit trail. The records are merged into a target file. The target file is written in a file system other than the file system that contains the audit root directory.

```
# auditreduce -c lo -d 20130827 -O /var/audit/audit_summary/logins
# ls /var/audit/audit_summary/*logins
/var/audit/audit_summary/20130827183936.20130827232326.logins
```

## Viewing the Contents of Binary Audit Files

As an administrator who is assigned the Audit Review rights profile, you can view the contents of binary audit files by using the `praudit` command.

```
# praudit options
```

The following is a partial list of options. You can combine any one of these options with the `-l` option to display each record on one line.

- `-s` Displays audit records in a short format, one token per line.
- `-r` Displays audit records in their raw format, one token per line.
- `-x` Displays audit records in XML format, one token per line. This option is useful for further processing.

You can also use the `auditreduce` and `praudit` commands together by piping the `praudit` output from the `auditreduce` command.

#### EXAMPLE 37 Displaying Audit Records in a Short Format

In this example, log in and log out events that are extracted by the `auditreduce` command are displayed in short format.

```
# auditreduce -c lo | praudit -s
header,69,2,AUE_screenlock,,mach1,2010-10-14 08:02:56.348 -07:00
subject,jdoe,root,staff,jdoe,staff,856,50036632,82 0 mach1
return,success,0
sequence,1298
```

**EXAMPLE 38** Displaying Audit Records in Raw Format

In this example, log in and log out events that are extracted by the `auditreduce` command are displayed in raw format.

```
# auditreduce -c lo | praudit -r

21,69,2,6222,0x0000,10.132.136.45,1287070091,698391050
36,26700,0,10,26700,10,856,50036632,82 0 10.132.136.45
39,0,0
47,1298
```

**EXAMPLE 39** Putting Audit Records in XML Format

In this example, the audit records are converted to XML format.

```
# praudit -x 20100827183214.20100827215318.logins > 20100827.logins.xml
```

Similarly, you can display audit records filtered by the `auditreduce` command in XML format.

```
# auditreduce -c lo | praudit -x
<record version="2" event="screenlock - unlock" host="mach1"
iso8601="2010-10-14 08:28:11.698 -07:00">
<subject audit-uid="jdoe" uid="root" gid="staff" ruid="jdoe
rgid="staff" pid="856" sid="50036632" tid="82 0 mach1"/>
<return errval="success" retval="0"/>
<sequence seq-num="1298"/>
</record>
```

The contents of the file can be operated on by a script to extract the relevant information.

**EXAMPLE 40** Making Audit Records in XML Format Readable in a Browser

You can reformat records in the XML file to become readable in any browser by using the `xsltproc` tool. This tool applies stylesheet definitions to the file contents. To put the reformatted contents in a separate file, you would type the following:

```
# auditreduce -c lo | praudit -x | xsltproc - > logins.html
```

In a browser, the contents of `logins.html` would be displayed in a format similar to the following:

```
Audit Trail Data

File: time: 2013-11-04 12:54:28.000 -08:00

Event: login - local
time: 2013-11-04 12:54:28.418 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jdoe uid: jdoe gid: staff ruid: jdoe rgid: staff
      pid: 1534 sid: 3583012893 tid: 0 0 host
```

RETURN errval: success retval: 0

Event: connect to RAD

time: 2013-11-04 12:54:52.029 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jdoe uid: jdoe gid: staff ruid: jdoe rgid: staff  
pid: 1835 sid: 3583012893 tid: 0 0 host

RETURN errval: success retval: 0

Event: role login

time: 2013-11-08 08:42:52.286 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jdoe uid: root gid: root ruid: root rgid: root  
pid: 4265 sid: 3583012893 tid: 0 0 host

RETURN errval: success retval: 0

Event: role logout

time: 2013-11-08 08:43:37.125 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jdoe uid: root gid: root ruid: root rgid: root  
pid: 4265 sid: 3583012893 tid: 0 0 host

RETURN errval: success retval: 0

Event: login - ssh

time: 2013-12-23 12:24:37.292 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jsmith uid: jsmith gid: staff ruid: jsmith rgid: staff  
pid: 2002 sid: 39351741 tid: 14632 202240 host.example.com

RETURN errval: success retval: 0

Event: role login

time: 2013-12-23 12:25:07.345 -08:00 vers: 2 mod: fe host: host

SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root  
pid: 2023 sid: 39351741 tid: 14632 202240 host.example.com

RETURN errval: failure retval: Permission denied

Event: su

time: 2013-12-23 17:19:24.031 -08:00 vers: 2 mod: na host: host

RETURN errval: success retval: 0

Event: su logout

time: 2013-12-23 17:19:24.362 -08:00 vers: 2 mod: na host: host

RETURN errval: success retval: 0

Event: login - ssh

time: 2013-12-23 17:27:21.306 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jsmith uid: jsmith gid: staff ruid: jsmith rgid: staff  
pid: 2583 sid: 3401970889 tid: 13861 5632 host.example.com

RETURN errval: success retval: 0

Event: role login

time: 2013-12-23 17:27:28.361 -08:00 vers: 2 mod: host: host

SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root  
pid: 2593 sid: 3401970889 tid: 13861 5632 host.example.com

RETURN errval: success retval: 0

```
Event: role logout
time: 2013-12-23 17:30:39.029 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root
      pid: 2593 sid: 3401970889 tid: 13861 5632 host.example.com
RETURN errval: success retval: 0
```

#### *Other events*

#### **EXAMPLE 41** Displaying pfedit Records Only

You can use filters to extract and view only specific records from the audit trail. In this example, records that capture the use of the `pfedit` command are filtered. Suppose that the summary file is `20130827183936.20130827232326.logins`. Use of the `pfedit` command generates the `AUE_admin_edit` event. Therefore, to extract `pfedit` records, run the following command:

```
auditreduce -m AUE_admin_edit 20130827183936.20130827232326.logins | praudit
```

#### **EXAMPLE 42** Printing the Entire Audit Trail

With a pipe to the `print` command, the output for the entire audit trail goes to the printer. For security reasons, the printer has limited access.

```
# auditreduce | praudit | lp -d example.protected.printer
```

#### **EXAMPLE 43** Viewing a Specific Audit File

In this example, a summary login file is examined in a terminal window.

```
# cd /var/audit/audit_summary/logins

# praudit 20100827183936.20100827232326.logins | more
```

#### **EXAMPLE 44** Processing praudit Output With a Script

You might want to process output from the `praudit` command as lines of text. For example, you might want to select records that the `auditreduce` command cannot select. You can use a simple shell script to process the output of the `praudit` command. The following sample script puts one audit record on one line, searches for a user-specified string, then returns the audit file to its original form.

```
#!/bin/sh
#
## This script takes an argument of a user-specified string.
```



```
# The sed command prefixes the header tokens with Control-A
# The first tr command puts the audit tokens for one record
# onto one line while preserving the line breaks as Control-A
#
praudit | sed -e '1,2d' -e '$s/^file.*$//' -e 's/^header/^aheader/' \
| tr '\012\001' '\002\012' \
| grep "$1" \
Finds the user-specified string

| tr '\002' '\012'
Restores the original newline breaks
```

Note that the `^a` in the script is Control-A, not the two characters `^` and `a`. The prefix distinguishes the header token from the string header that might appear as text.

A message similar to the following indicates that you do not have enough privilege to use the `praudit` command:

```
praudit: Can't assign 20090408164827.20090408171614.sys1.1 to stdin.
```

Run the `praudit` command in a profile shell. You must become an administrator who is assigned the Audit Review rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

## Managing Audit Records on Local Systems

The following task map points to procedures for selecting, analyzing, and managing audit records.

**TABLE 7** Managing Audit Records on Local Systems Task Map

Task	Description	For Instructions
Merge audit records.	Combines audit files from several audit directories into one audit trail.	<a href="#">“How to Merge Audit Files From the Audit Trail” on page 97</a>
Clean up incorrectly named audit files.	Provides an end time stamp to audit files that were inadvertently left open by the audit service.	<a href="#">“How to Clean Up a not_terminated Audit File” on page 99</a>
Prevent audit trail overflow.	Prevents the audit file systems from becoming full.	<a href="#">“Preventing Audit Trail Overflow” on page 100</a>

### ▼ How to Merge Audit Files From the Audit Trail

By merging the audit files from all the audit directories, you can analyze the contents of the entire audit trail.

---

**Note** - Because the time stamps in the audit trail are in Coordinated Universal Time (UTC), the date and hour must be translated to the current time zone to be meaningful. Be aware of this point whenever you manipulate these files with standard file commands rather than with the `auditreduce` command.

---

**Before You Begin** You must become an administrator who is assigned the Audit Review rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

**1. Create a file system for storing merged audit files.**

To lessen the chance of reaching the limit of disk space, this file system should be in a *different zpool* from the file systems that you created in [“How to Create ZFS File Systems for Audit Files” on page 74](#) to store the original files.

**2. Merge the audit records in the audit trail.**

Go to the directory for storing merged audit files. From this directory, merge the audit records into a file with a named suffix. All directories in the audit trail on the local system are merged and placed in this directory.

```
# cd audit-storage-directory
# auditreduce -Uppercase-option -O suffix
```

The uppercase options of the `auditreduce` command manipulate files in the audit trail. The uppercase options include the following:

-A	Selects all of the files in the audit trail.
-C	Selects complete files only.
-M	Selects files with a particular suffix. The suffix can be a physical machine name or a suffix that you have specified for a summary file.
-O	Creates an audit file with 14-character time stamps for both the start time and the end time, with the suffix <i>suffix</i> in the current directory.
-R <i>pathname</i>	Specifies to read audit files in <i>pathname</i> , an alternate audit root directory.
-S <i>server</i>	Specifies to read audit files from the specified server.

For the full list of options, see the [auditreduce\(1M\)](#) man page.

**Example 45** Copying Audit Files to a Summary File

In the following example, an administrator who is assigned the System Administrator rights profile copies all files from the audit trail into a merged file on a different file system. The `/var/`

audit/storage file system is on a separate disk from /var/audit which functions as the audit root file system.

```
$ cd /var/audit/storage
$ auditreduce -A -O All
$ ls /var/audit/storage/*All
20100827183214.20100827215318.All
```

In the following example, only complete files are copied from the audit trail into a merged file. The complete path is specified as the value of the `-O` option. The last component of the path, Complete, is used as the suffix.

```
$ auditreduce -C -O /var/audit/storage/Complete

$ ls /var/audit/storage/*Complete
20100827183214.20100827214217.Complete
```

In the following example, by adding the `-D` option, the original audit files are deleted.

```
$ auditreduce -C -O daily_sys1.1 -D sys1.1

$ ls *sys1.1
20100827183214.20100827214217.daily_sys1.1
```

## ▼ How to Clean Up a not\_terminated Audit File

When anomalous system interruptions occur, the audit service exits while its audit file is still open. Or, a file system becomes inaccessible and forces the system to switch to a new file system. In this situation, an audit file remains with the string `not_terminated` as the end time stamp, even though the file is no longer used for audit records. Use the `auditreduce -O` command to give the file the correct time stamp.

**Before You Begin** You must become an administrator who is assigned the Audit Review rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

1. **List the files with the `not_terminated` string on your audit file system in order of creation.**

```
# ls -Rlt audit-directory */* | grep not_terminated
```

-R	Lists files in subdirectories.
-t	Lists files from most recent to oldest.
-l	Lists the files in one column.

**2. Clean up the old not\_terminated file.**

Specify the name of the old file to the auditreduce -O command.

```
# auditreduce -O system-name old-not-terminated-file
```

**3. Remove the old not\_terminated file.**

```
# rm system-name old-not-terminated-file
```

**Example 46** Cleaning Up Closed not\_terminated Audit Files

In the following example, not\_terminated files are found, renamed, then the originals are removed.

```
ls -Rlt */* | grep not_terminated
.../egret.1/20100908162220.not_terminated.egret
.../egret.1/20100827215359.not_terminated.egret

# cd */egret.1
# auditreduce -O egret 20100908162220.not_terminated.egret
# ls -lt
20100908162220.not_terminated.egret      Current audit file

20100827230920.20100830000909.egret      Cleaned-up audit file

20100827215359.not_terminated.egret      Input (old) audit file

# rm 20100827215359.not_terminated.egret
# ls -lt
20100908162220.not_terminated.egret      Current audit file

20100827230920.20100830000909.egret      Cleaned-up audit file
```

The start time stamp on the new file reflects the time of the first audit event in the not\_terminated file. The end time stamp reflects the time of the last audit event in the file.

## Preventing Audit Trail Overflow

If your security policy requires that all audit data be saved, prevent audit record loss by observing the following practices.

---

**Note** - You must assume the root role. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

---

- Set a minimum free size on the audit\_binfile plugin.

Use the `p_minfree` attribute.

The `audit_warn` email alias sends a warning when the disk space fills to the minimum free size. See [Example 25, “Setting a Soft Limit for Warnings,” on page 80](#).

- Set up a schedule to regularly archive audit files.

Archive audit files by backing up the files to offline media. You can also move the files to an archive file system.

If you are collecting text audit logs with the `syslog` utility, archive the text logs. For more information, see the `logadm(1M)` man page.

- Set up a schedule to delete the archived audit files from the audit file system.
- Save and store auxiliary information.

Archive information that is necessary to interpret audit records along with the audit trail. Minimally, you save `passwd`, `group`, and `hosts`. You also might archive `audit_event` and `audit_class`.

- Keep records of which audit files have been archived.
- Store the archived media appropriately.
- Reduce the amount of file system capacity that is required by enabling ZFS compression.

On a ZFS file system that is dedicated to audit files, compression shrinks the files considerably. For an example, see [“How to Compress Audit Files on a Dedicated File System” on page 63](#).

See also [“Interactions Between ZFS Compression, Deduplication, and Encryption Properties” in \*Managing ZFS File Systems in Oracle Solaris 11.3\*](#).

- Reduce the volume of audit data that you store by creating summary files.

You can extract summary files from the audit trail by using options to the `auditreduce` command. The summary files contain only records for specified types of audit events. To extract summary files, see [Example 34, “Combining and Reducing Audit Files,” on page 92](#) and [Example 36, “Merging Selected Records to a Single File,” on page 93](#).



## Analyzing and Resolving Audit Service Issues

---

This chapter provides procedures to help you troubleshoot audit-related issues. In addition, the following chapters describe other audit management tasks:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 5, “Working With Audit Data”](#)

For an overview of the audit service, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For reference information, see [Chapter 7, “Auditing Reference”](#).

### Troubleshooting the Audit Service

This section covers various auditing error messages, preferences, and the auditing that is provided by other tools to help you debug audit problems.

Typically, different notices are sent to alert you of errors in the audit service. Review your email and the log files if you think that problems exist with the audit service.

- Read the email sent to the `audit_warn` alias.  
The `audit_warn` script sends alert messages to the `audit_warn` email alias. In the absence of a correctly configured alias, the messages are sent to the `root` account.
- Review the log files for the audit service.  
The output from the `svcs -s auditd` command lists the full path to the audit logs that the audit service produces.
- Review the system log files.  
The `audit_warn` script writes `daemon.alert` messages to the `/var/log/syslog` file.  
The `/var/adm/messages` file might contain information.

After you locate and fix the problems, enable or restart the audit service.

```
# audit -s
```

The following sections describe possible problem cases and the steps to resolve them.

---

**Note** - Before you perform any troubleshooting tasks, ensure that you have the proper authorization. For example, to configure auditing, you must become an administrator who is assigned the Audit Configuration rights profile. For more information, see [“Using Your Assigned Administrative Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

---

## Audit Records Are Not Being Logged

Auditing is enabled by default. If you believe that auditing has not been disabled, but no audit records are being sent to the active plugin, the causes might be one or a combination of the following factors discussed in this section. Note that to modify a system file, you must be assigned the `solaris.admin.edit/path-to-system-file` authorization. By default, the root role has this authorization.

### Audit Service Not Running

To check whether auditing is running, use any of the following methods:

- Verify the current audit condition.

The following output indicates that auditing is not running:

```
# auditconfig -getcond
audit condition = noaudit
```

The following output indicates that auditing is running:

```
# auditconfig -getcond
audit condition = auditing
```

- Verify that the audit service is running.

The following output indicates that auditing is not running:

```
# svcs -x auditd

svc:/system/auditd:default (Solaris audit daemon)
State: disabled since Sun Oct 10 10:10:10 2010
Reason: Disabled by an administrator.
See: http://support.oracle.com/msg/SMF-8000-05
See: auditd(1M)
See: audit(1M)
See: auditconfig(1M)
See: audit_flags(5)
See: audit_binfile(5)
```



```
See: audit_syslog(5)
See: audit_remote(5)
See: /var/svc/log/system-auditd:default.log
Impact: This service is not running.
```

The following output indicates that the audit service is running:

```
# svcs auditd
STATE      STIME      FMRI
online     10:10:10   svc:/system/auditd:default
```

If the audit service is not running, enable it. For the procedure, see [“Enabling and Disabling the Audit Service” on page 42](#).

## No Audit Plugin Active

Use the following command to check if any plugins are active. At least one plugin must be active for the audit service to work.

```
# audit -v
audit: no active plugin found
```

If no plugin is active, make one active.

```
# auditconfig -setplugin audit_binfile active
# audit -v
configuration ok
```

## Audit Class Undefined

You might be attempting to use an audit class that has not been defined. For a description of creating the pf class, see [“How to Add an Audit Class” on page 54](#).

For example, the following list of flags contains the pf class, which Oracle Solaris software did not deliver:

```
# auditconfig -getflags
active user default audit flags = pf,lo(0x0100000000000000,00x0100000000001000)
configured user default audit flags = pf,lo(0x0100000000000000,00x0100000000001000)
```

If you do not want to define the class, run the `auditconfig -setflags` command with valid values to reset the current flags. Otherwise, ensure the following when defining a class:

- The audit class is defined in the `audit_class` file.

```
# grep pf /etc/security/audit_class
Verify class exists
```

```
0x0100000000000000:pf:profile
```

- The mask is unique. If it is not unique, replace the mask.

```
# grep 0x0100000000000000 /etc/security/audit_class
    Ensure mask is unique
```

```
0x0100000000000000:pf:profile
```

## No Assigned Events to Audit Class

The customized class that you are using, although defined, might not have any events assigned to the class.

To verify whether events are assigned to the customized class, use one of the following methods:

```
# auditconfig -lsevent | egrep " pf|,pf|pf,"
AUE_PFEEXEC      116 pf execve(2) with pfexec enabled

# auditrecord -c pf
    List of audit events assigned to pf class
```

If events are not assigned to the class, assign the appropriate events to this class.

## Volume of Audit Records Is Large

After you have determined which events must be audited at your site, use the following suggestions to create audit files with just the information that you require. Note that to assign flags to users, roles, and rights profiles, you must assume the `root` role.

- Specifically, avoid adding events and audit tokens to the audit trail. The following policies increase the size of the audit trail.

arge	Adds environment variables to <code>execv</code> audit events. Although auditing <code>execv</code> events can be costly, adding variables to the audit record is not.
argv	Adds command parameters to <code>execv</code> audit events. Adding command parameters to the audit record is not costly.
group	Adds a group token to audit events that include an optional <code>newgroups</code> token.
path	Adds a path token to audit events that include an optional path token.

public	If file events are being audited, adds an event to the audit trail every time an auditable event happens to a <a href="#">public object</a> . File classes include fa, fc, fd, fm, fr, fw, and cl. For the definition of a public file, see “ <a href="#">Audit Terminology and Concepts</a> ” on page 12.
seq	Adds a sequence token to every audit event.
trail	Adds a trailer token to every audit event.
windata_down	On a system that is configured with Trusted Extensions, adds events when information in a labeled window is downgraded.
windata_up	On a system that is configured with Trusted Extensions, adds events when information in a labeled window is upgraded.
zonename	Adds the zone name to every audit event. If the global zone is the only configured zone, adds the string zone, global to every audit event.

The following audit record shows the use of the `ls` command. The `ex` class is being audited and the default policy is in use:

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 11:39:22.480 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2404,50036632,82 0 mach1
return,success,0
```

The following is the same record when all policies are turned on:

```
header,1578,2,AUE_EXECVE,,mach1,2010-10-14 11:45:46.658 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8, PRINTER=example-dbl,
...
path,/lib/ld.so.1
attribute,100755,root,bin,21,393073,18446744073709551615
subject,jdoe,root,root,root,root,2424,50036632,82 0 mach1
group,root,other,bin,sys,adm,uucp,mail,tty,lp,nuucp,daemon
return,success,0
zone,global
sequence,197
trailer,1578
```

- Use the `audit_syslog` plugin to send some audit events to `syslog`.  
Do not send those audit events to the `audit_binfile` or `audit_remote` plugin. This strategy works only if you are not required to keep binary records of the audit events that you send to the `syslog` logs.
- Set fewer system-wide audit flags and audit individual users.  
Reduce the amount of auditing for all users by reducing the number of audit classes that are audited system-wide.  
Use the `audit_flags` keyword to the `roleadd`, `rolemode`, `useradd`, and `usermod` commands to audit events for specific users and roles. For examples, see [Example 29, “Specifying Audit Classes for `syslog` Output,” on page 87](#) and the `usermod(1M)` man page.  
Use the `always_audit` and `never_audit` properties of the `profiles` command to audit events for specific rights profiles. For information, see the `profiles(1)` man page.

---

**Note** - Like other security attributes, audit flags are affected by search order. For more information, see [“Order of Search for Assigned Rights” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

---

- Create your own customized audit class.  
You can create audit classes at your site. Into these classes, put only those audit events that you need to monitor. For the procedure, see [“How to Add an Audit Class” on page 54](#).

---

**Note** - For information about the effects of modifying an audit configuration file, see [“Audit Configuration Files and Packaging” on page 115](#).

---

## Binary Audit File Sizes Grow Without Limit

As an administrator who is assigned the Audit Review rights profile, you can limit the size of binary files to facilitate archiving and searching. You can also create smaller binary files from the original file by using one of the options described in this section.

- Use the `p_fsize` attribute to limit the size of individual binary audit files.  
For a description of the `p_fsize` attribute, see the OBJECT ATTRIBUTES section of the `audit_binfile(5)` man page.  
For an example, see [Example 21, “Limiting File Size for the `audit\_binfile` Plugin,” on page 78](#).
- Use the `auditreduce` command to select records and write those records to a smaller file for further analysis.  
The `auditreduce -lowercase` options find specific records.

The `auditreduce -Uppercase` options write your selections to a file. For more information, see the [auditreduce\(1M\)](#) man page. See also “[Displaying Audit Trail Data](#)” on page 89.

## Logins From Other Operating Systems Not Being Audited

The Oracle Solaris OS can audit all logins independent of source. If logins are not being audited, then the `lo` class for both attributable and non-attributable events is probably not set. This class audits logins, logouts, and screen locks. These classes are audited by default.

---

**Note** - To audit `ssh` logins, your system must be running the `ssh` daemon from Oracle Solaris. This daemon is modified for the audit service on an Oracle Solaris system. For more information, see “[sunssh Implementation of Secure Shell](#)” in *Managing Secure Shell Access in Oracle Solaris 11.3*.

---

### EXAMPLE 47 Ensuring That Logins Are Audited

In this example, the output of the first two commands shows that the `lo` class for attributable and non-attributable events is not set. Then, the last two commands set the `lo` class to enable auditing of login events.

```
# auditconfig -getflags
active user default audit flags = as,st(0x20800,0x20800)
configured user default audit flags = as,st(0x20800,0x20800)

# auditconfig -getnaflags
active non-attributable audit flags = na(0x400,0x400)
configured non-attributable audit flags = na(0x400,0x400)

# auditconfig -setflags lo,as,st
user default audit flags = as,lo,st(0x21800,0x21800)

# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```



## Auditing Reference

---

This chapter describes the important components of auditing, and covers the following topics:

- “Audit Service” on page 111
- “Audit Service Man Pages” on page 112
- “Rights Profiles for Administering Auditing” on page 114
- “Auditing and Oracle Solaris Zones” on page 114
- “Audit Configuration Files and Packaging” on page 115
- “Audit Classes” on page 115
- “Audit Plugins” on page 116
- “Audit Remote Server” on page 117
- “Audit Policy” on page 117
- “Process Audit Characteristics” on page 119
- “Audit Trail” on page 120
- “Conventions for Binary Audit File Names” on page 120
- “Audit Record Structure” on page 120
- “Audit Token Formats” on page 121

For an overview of auditing, see [Chapter 1, “About Auditing in Oracle Solaris”](#). For planning suggestions, see [Chapter 2, “Planning for Auditing”](#). For procedures to configure auditing at your site, see the following chapters:

- [Chapter 3, “Managing the Audit Service”](#)
- [Chapter 4, “Monitoring System Activities”](#)
- [Chapter 5, “Working With Audit Data”](#)
- [Chapter 6, “Analyzing and Resolving Audit Service Issues”](#)

## Audit Service

The audit service, `auditd`, is enabled by default. To find out how to enable, refresh, or disable the service, see [“Enabling and Disabling the Audit Service” on page 42](#).

Without customer configuration, the following defaults are in place:

- All login events are audited.  
Both successful and unsuccessful login attempts are audited.
- All users are audited for login and logout events, including role assumption and screen lock.
- The `audit_binfile` plugin is active. `/var/audit` stores audit records, the size of an audit file is not limited, and the queue size is 100 records.
- The `cnt` policy is set.  
When audit records fill the available disk space, the system tracks the number of dropped audit records. A warning is issued when one percent of available disk space remains.
- The following audit queue controls are set:
  - Maximum number of records in the audit queue before generating the records locks is 100
  - Minimum number of records in the audit queue before blocked auditing processes unblock is 10
  - Buffer size for the audit queue is 8192 bytes
  - Interval between writing audit records to the audit trail is 20 seconds

To display the defaults, see [“Displaying Audit Service Defaults” on page 40](#).

The audit service enables you to set temporary, or active, values. These values can differ from configured, or property, values.

- Temporary values are not restored when you refresh or restart the audit service.  
Audit policy and audit queue controls accept temporary values. Audit flags do not have a temporary value.
- Configured values are stored as property values of the service, so they are restored when you refresh or restart the audit service.

Rights profiles control who can administer the audit service. For more information, see [“Rights Profiles for Administering Auditing” on page 114](#).

By default, all zones are audited identically. See [“Auditing and Oracle Solaris Zones” on page 114](#).

## Audit Service Man Pages

The following table summarizes the major administrative man pages for the audit service.

Man Page	Summary
<a href="#">audit(1M)</a>	Command that controls the actions of the audit service
	<code>audit -n</code> starts a new audit file for the <code>audit_binfile</code> plugin.



Man Page	Summary
	<code>audit -s</code> enables and refreshes auditing.
	<code>audit -t</code> disables auditing.
	<code>audit -v</code> verifies that at least one plugin is active.
<a href="#">audit_binfile(5)</a>	Default audit plugin, which sends audit records to a binary file. See also “ <a href="#">Audit Plugins</a> ” on page 116.
<a href="#">audit_remote(5)</a>	Audit plugin that sends audit records to a remote receiver.
<a href="#">audit_syslog(5)</a>	Audit plugin that sends text summaries of audit records to the <code>syslog</code> utility.
<a href="#">audit_class(4)</a>	File that contains the definitions of audit classes. The eight high-order bits are available for customers to create new audit classes. For more information about the effect of modifying this file on system upgrade, see “ <a href="#">How to Add an Audit Class</a> ” on page 54.
<a href="#">audit_event(4)</a>	File that contains the definitions of audit events and maps the events to audit classes. The mapping can be modified. For more information about the effect of modifying this file on system upgrade, see “ <a href="#">How to Change an Audit Event's Class Membership</a> ” on page 55.
<a href="#">audit_flags(5)</a>	Describes the syntax of audit class preselection, the prefixes for selecting only failed events or only successful events, and the prefixes that modify an existing preselection.
<a href="#">audit.log(4)</a>	Describes the naming of binary audit files, the internal structure of a file, and the structure of every audit token.
<a href="#">audit_warn(1M)</a>	Script that notifies an email alias when the audit service encounters an unusual condition while writing audit records. You can customize this script for your site to warn of conditions that might require manual intervention or can specify how to handle those conditions automatically.
<a href="#">auditconfig(1M)</a>	Command that retrieves and sets audit configuration parameters.
	Issue this <code>auditconfig</code> with no options to display a list of parameters that can be retrieved and set.
<a href="#">auditrecord(1M)</a>	Command that displays the definition of audit events in the <code>/etc/security/audit_event</code> file. For sample output, see “ <a href="#">Displaying Audit Record Definitions</a> ” on page 89.
<a href="#">auditreduce(1M)</a>	Command that post-selects and merges audit records that are stored in binary format. The command can merge audit records from one or more input audit files. The records remain in binary format.
	Uppercase options affect file selection. Lowercase options affect record selection.
<a href="#">auditstat(1M)</a>	Command that displays kernel audit statistics. For example, the command can display the number of records in the kernel audit queue, the number of dropped records, and the number of audit records that user processes produced in the kernel as a result of system calls.
<a href="#">praudit(1M)</a>	Command that reads audit records in binary format from standard input and displays the records in a presentable format. The input can be piped from the <code>auditreduce</code> command or from a single audit file or a list of audit files. Input can also be produced with the <code>tail -0f</code> command for a current audit file.
	For sample output, see “ <a href="#">Viewing the Contents of Binary Audit Files</a> ” on page 93.
<a href="#">syslog.conf(4)</a>	File that is configured to send text summaries of audit records to the <code>syslog</code> utility for the <code>audit_syslog</code> plugin.

## Rights Profiles for Administering Auditing

Oracle Solaris provides rights profiles for configuring the audit service, for enabling and disabling the service, and for analyzing the audit trail. You must have the privileges of root to edit an audit configuration file.

- **Audit Configuration** – Enables an administrator to configure the parameters of the audit service and to run the `auditconfig` command.
- **Audit Control** – Enables an administrator to start, refresh, and disable the audit service and to run the `audit` command to start, refresh, or stop the service.
- **Audit Review** – Enables an administrator to analyze audit records. This rights profile grants authorization to read audit records with the `praudit` and `auditreduce` commands. This administrator can also run the `auditstat` command.
- **System Administrator** – Includes the Audit Review rights profile. An administrator with the System Administrator rights profile can analyze audit records.

To configure roles to handle the audit service, see [“Creating a Role” in \*Securing Users and Processes in Oracle Solaris 11.3\*](#).

## Auditing and Oracle Solaris Zones

Non-global zones can be audited exactly as the global zone is audited, or non-global zones can set their own flags, storage, and audit policy.

When all zones are being audited identically through the global zone, the `audit_class` and `audit_event` files provide the class-event mappings for auditing in the global zone and in every non-global zone. The `+zonename` policy option is useful for post-selecting records by zone name.

Zones can also be audited individually. When the policy option, `perzone`, is set in the global zone, each non-global zone runs its own audit service, handles its own audit queue, and specifies the content and location of its audit records. A non-global zone can also set most audit policy options. It cannot set policy that affects the entire system, so a non-global zone cannot set the `ahlt` or `perzone` policy. For further discussion, see [“Auditing on a System With Oracle Solaris Zones” on page 26](#) and [“Planning Auditing in Zones” on page 28](#).

To learn about zones, see [Introduction to Oracle Solaris Zones](#).

## Audit Configuration Files and Packaging

The audit configuration files in Oracle Solaris are marked in the package with the `preserve=renamenew` package attribute. This attribute enables the files to be modified, and the modifications are retained across package updates and package fixes. For information about the effects of the preserve values, see the `pkg(5)` man page.

These configuration files are also marked with the `overlay=allow` package attribute. This attribute enables you to create your own package that contains these files and replace the Oracle Solaris files with files from your package. When you set the `overlay` attribute to `true` in your package, the `pkg` subcommands, such as `verify`, `fix`, `revert`, and so on, will return results on your packages. For more information, see the `pkg(1)` and `pkg(5)` man pages.

## Audit Classes

Oracle Solaris defines audit classes as convenient containers for large numbers of audit events.

You can reconfigure audit classes and make new audit classes. Audit class names can be up to 8 characters in length. The class description is limited to 72 characters. Numeric and non-alphanumeric characters are allowed. For more information, see the [audit\\_class\(4\)](#) man page and “[How to Add an Audit Class](#)” on page 54.



---

**Caution** - The `all` class can generate large amounts of data and quickly fill disks. Use the `all` class only if you have extraordinary reasons to audit all activities.

---

## Audit Class Syntax

Events in an audit class can be audited for success, for failure, and for both.

- Without a prefix, a class of events is audited for success and for failure.
- With a plus (+) prefix, a class of events is audited for success only.
- With a minus (-) prefix, a class of events is audited for failure only.
- To modify a current preselection, add a caret (^) preceding a prefix or an audit flag. For example:
  - If `ot` is preselected for the system, and a user's preselection is `^ot`, that user is not audited for events in the other class.
  - If `+ot` is preselected for the system, and a user's preselection is `^+ot`, that user is not audited for successful events in the other class.

- If `-ot` is preselected for the system, and a user's preselection is `^-ot`, that user is not audited for failed events in the other class.

To review the syntax of audit class preselection, see the [audit\\_flags\(5\)](#) man page.

The audit classes and their prefixes can be specified in the following commands:

- As arguments to the `auditconfig` command options `-setflags` and `-setnaflags`.
- As values for the `p_flags` attribute to the `audit_syslog` plugin. You specify the attribute as an option to the `auditconfig -setplugin audit_syslog active` command.
- As values for the `-K audit_flags=always-audit-flags:never-audit-flags` option to the `useradd`, `usermod`, `roleadd`, and `rolemod` commands.
- As values for the `-always_audit` and `-never_audit` properties of the `profiles` command.

## Audit Plugins

Audit plugins specify how to handle the audit records in the audit queue. The audit plugins are specified by name: `audit_binfile`, `audit_remote`, and `audit_syslog`, as arguments to the `auditconfig -setplugin` command. The plugins can be further specified by the following attributes:

- `audit_binfile` plugin
  - `p_dir` attribute – Where to send binary data
  - `p_minfree` attribute – Minimum space remaining on a disk before the administrator is warned.
  - `p_fsize` attribute – Maximum size of an audit file.
- `audit_remote` plugin
  - `p_hosts` attribute – Remote authenticated audit server to which to send the binary audit data.
  - `p_retries` attribute – Number of attempts to make to reach a remote authenticated audit server.
  - `p_timeout` attribute – Number of seconds between attempts to reach a remote authenticated audit server.
- `audit_syslog` plugin
  - `p_flags` attribute – Selection of text summaries of audit records to be sent to `syslog`
- For all plugins, the maximum number of audit records that are queued for the plugin - `qsize` attribute

Refer to the [audit\\_binfile\(5\)](#), [audit\\_remote\(5\)](#), [audit\\_syslog\(5\)](#), and [auditconfig\(1M\)](#) man pages.

## Audit Remote Server

The Audit Remote Server (ARS) receives audit records over a secure link from audited systems and stores the records.

The reception relies on the following being configured:

- A Kerberos realm with specific audit principals and a GSS-API mechanism
- The ARS with at least one configured and active *connection group*
- At least one audited system in the connection group and a configured and active `audit_remote` plugin

A connection group is specified in the `group` property of the ARS. For file management, `group` can limit the size of an audit file and specify the minimum free space. The primary reason to specify different connection groups is to specify different storage locations on the ARS, as shown in [Example 27, “Streaming Audit Records to Different File Locations on the Same ARS,”](#) on page 85.

For more information about the ARS, see the [ars\(5\)](#) man page. For ARS configuration information, see the `-setremote` options in the [auditconfig\(1M\)](#) man page.

To configure the audited systems, see the [audit\\_remote\(5\)](#) man page and the `-setplugin` option in the [auditconfig\(1M\)](#) man page.

## Audit Policy

Audit policy determines whether additional information is added to the audit trail.

The following policies add tokens to audit records: `args`, `argv`, `group`, `path`, `seq`, `trail`, `windata_down`, `windata_up`, and `zonename`. The `windata_down` and `windata_up` policies are used by the Trusted Extensions feature of Oracle Solaris. For more information, see [Chapter 22, “Trusted Extensions and Auditing”](#) in *Trusted Extensions Configuration and Administration*.

The remaining policies do not add tokens. The `public` policy limits auditing of public files. The `perzone` policy establishes separate audit queues for non-global zones. The `ahlt` and `cnt` policies determine what happens when audit records cannot be delivered. For details, see [“Audit Policies for Asynchronous and Synchronous Events”](#) on page 118.

The effects of the different audit policy options are described in [“Understanding Audit Policy”](#) on page 34. For a description of audit policy options, see the `-setpolicy` option in the [auditconfig\(1M\)](#) man page. For a list of available policy options, run the command `auditconfig -lspolicy`. For the current policy, run the command `auditconfig -getpolicy`.

## Audit Policies for Asynchronous and Synchronous Events

Together, the `ahlt` policy and the `cnt` policy govern what happens when the audit queue is full and cannot accept more events.

---

**Note** - The `cnt` or `ahlt` policies are not triggered if the queue for at least one plugin can accept audit records.

---

The `cnt` and `ahlt` policies are independent and related. The combination of the policies has the following effects:

- `-ahlt +cnt` is the default policy that is shipped. This default allows the processing of an audited event even if the event cannot be logged.  
 The `-ahlt` policy states that if an audit record of an asynchronous event cannot be placed in the kernel audit queue, the system will count the events and continue processing.  
 The `+cnt` policy states that if a synchronous event arrives and the event cannot be placed in the kernel audit queue, the system will count the event and continue processing.  
 The `-ahlt +cnt` configuration is generally used at sites where processing must continue, even if continued processing could result in a loss of audit records. The `auditstat drop` field shows the number of audit records that are dropped in a zone.
- The `+ahlt -cnt` policy states that processing halts when an asynchronous event cannot be added to the kernel audit queue.  
 The `+ahlt` policy states that if an audit record of an asynchronous event cannot be placed in the kernel audit queue, all processing is stopped. The system will panic. The asynchronous event will not be in the audit queue and must be recovered from pointers on the call stack.  
 The `-cnt` policy states that if a synchronous event cannot be placed in the kernel audit queue, the thread that is attempting to deliver the event will be blocked. The thread is placed in a sleep queue until audit space becomes available. No count is kept. Programs might appear to hang until audit space becomes available.  
 The `+ahlt -cnt` configuration is generally used at sites where a record of every audit event takes precedence over system availability. The `auditstat wblk` field shows the number of times that threads were blocked.  
 However, if an asynchronous event occurs, the system will panic, leading to an outage. The kernel queue of audit events can be manually recovered from a saved crash dump. The asynchronous event will not be in the audit queue and must be recovered from pointers on the call stack.
- The `-ahlt -cnt` policy states that if an asynchronous event cannot be placed in the kernel audit queue, the event will be counted and processing will continue. When a synchronous event cannot be placed in the kernel audit queue, the thread that is attempting to deliver

the event will be blocked. The thread is placed in a sleep queue until audit space becomes available. No count is kept. Programs might appear to hang until audit space becomes available.

The `-ahlt -cnt` configuration is generally used at sites where the recording of all synchronous audit events takes precedence over some potential loss of asynchronous audit records. The `auditstat wblk` field shows the number of times that threads were blocked.

- The `+ahlt +cnt` policy states that if an asynchronous event cannot be placed in the kernel audit queue, the system will panic. If a synchronous event cannot be placed in the kernel audit queue, the system will count the event and continue processing.

## Process Audit Characteristics

The following audit characteristics are set at initial login:

- **Process preselection mask** – A combination of the system-wide audit mask and the user-specific audit mask, if a user audit mask has been specified. When a user logs in, the login process combines the preselected classes to establish the *process preselection mask* for the user's processes. The process preselection mask specifies the events that generate audit records.

In addition, as described in the [audit\\_flags\(5\)](#) man page, the preselection can specify auditing only successful events, auditing only failed events, or auditing all events.

The following algorithm describes how the system obtains the user's process preselection mask:

```
(system-wide default flags + always-audit-classes) - never-audit-classes
```

Add the system-wide audit classes from the results of the `auditconfig -getflags` command to the classes from the *always-audit-classes* value for the user's `always_audit` keyword. Then, from the total, subtract the classes from the user's *never-audit-classes*. See also the [audit\\_flags\(5\)](#) man page.

- **Audit user ID** – A process acquires an immutable audit user ID when the user logs in. This ID is inherited by all child processes that were started by the user's initial process. The audit user ID helps enforce accountability. Even after a user assumes a role, the audit user ID remains the same. The audit user ID that is saved in each audit record enables you to always trace actions back to the login user.
- **Audit session ID** – The audit session ID is assigned at login. This ID is inherited by all child processes.
- **Terminal ID** – For a local login, the terminal ID consists of the local system's IP address, followed by a device number that identifies the physical device on which the user logged in. Most often, the login is through the console. The number that corresponds to the console device is `0,0`. For a remote login, the terminal ID consists of the remote system's IP address followed by the remote port number and the local port number.

## Audit Trail

The *audit trail* contains binary audit files. The trail is created by the `audit_binfile` plugin. The audit service collects the records in the audit queue and sends them to the plugin, which writes them to disk.

## Conventions for Binary Audit File Names

The `audit_binfile` plugin creates binary audit files. Each binary 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 time stamps that indicate the time span are specified in Coordinated Universal Time (UTC) to ensure that they sort in proper order, even across time zones.

For more information, see the [audit.log\(4\)](#) man page. For examples of open and closed audit file names, see [“How to Clean Up a not\\_terminated Audit File” on page 99](#).

## Audit Record Structure

An audit record is a sequence of audit tokens. Each audit token contains event information such as user ID, time, and date. A header token begins an audit record, and an optional trailer token concludes the record. Other audit tokens contain information relevant to the audit event. The following figure shows a typical kernel audit record and a typical user-level audit record.

**FIGURE 3** Typical Audit Record Structures

header token	header token
arg token	subject token
data tokens	[other tokens]
subject token	return token
return token	



## Audit Record Analysis

Audit record analysis involves post-selecting records from the audit trail. You can use one of two approaches to parsing the binary data that was collected.

- You can use the `praudit` command. Options to the command provide different text output. For example, the `praudit` command provides XML for input into scripts and browsers. `praudit` output does not include fields whose sole purpose is to help to parse the binary data. Note that the order and format of `praudit` output is not guaranteed between Oracle Solaris releases.

For examples of `praudit` output, see [“Viewing the Contents of Binary Audit Files” on page 93](#).

For examples of `praudit` output for each audit token, see the individual tokens in [“Audit Token Formats” on page 121](#).

- You can write a program to parse the binary data stream. The program must take into account the variants of an audit record. For example, the `ioctl()` system call creates an audit record for “Bad file name.” This record contains different tokens from the `ioctl()` audit record for “Invalid file descriptor.”
  - For a description of the order of binary data in each audit token, see the `audit.log(4)` man page.
  - For manifest values, see the `/usr/include/bsm/audit.h` file.
  - To view the order of tokens in an audit record, use the `auditrecord` command. Output from the `auditrecord` command includes the different tokens for different manifest values. Square brackets ([]) indicate that an audit token is optional. For more information, see the `auditrecord(1M)` man page.

## Audit Token Formats

Each audit token has a token type identifier, which is followed by data that is specific to the token. The following table shows the token names with a brief description of each token. Obsolete tokens are maintained for compatibility with previous Solaris releases.

**TABLE 8** Audit Tokens for Auditing

Token Name	Description	For More Information
<code>acl</code>	Access Control Entry (ACE) and Access Control List (ACL) information	<a href="#">“acl Token” on page 123</a>
<code>arbitrary</code>	Data with format and type information	<a href="#">audit.log(4) man page</a>
<code>argument</code>	System call argument value	<a href="#">“argument Token” on page 123</a>
<code>attribute</code>	File vnode information	<a href="#">“attribute Token” on page 123</a>

Token Name	Description	For More Information
cmd	Command arguments and environment variables	<a href="#">“cmd Token” on page 124</a>
exec_args	Exec system call arguments	<a href="#">“exec_args Token” on page 124</a>
exec_env	Exec system call environment variables	<a href="#">“exec_env Token” on page 124</a>
exit	Program exit information	<a href="#">audit.log(4) man page</a>
file	Audit file information	<a href="#">“file Token” on page 125</a>
fmri	Framework Management Resource Indicator	<a href="#">“fmri Token” on page 125</a>
group	Process groups information	<a href="#">“group Token” on page 125</a>
header	Indicates start of audit record	<a href="#">“header Token” on page 125</a>
ip	IP header information	<a href="#">audit.log(4) man page</a>
ip address	Internet address	<a href="#">“ip address Token” on page 126</a>
ip port	Internet port address	<a href="#">“ip port Token” on page 126</a>
ipc	System V IPC information	<a href="#">“ipc Token” on page 126</a>
IPC_perm	System V IPC object access information	<a href="#">“IPC_perm Token” on page 127</a>
opaque	Unstructured data (unspecified format)	<a href="#">audit.log(4) man page</a>
path	Path information	<a href="#">“path Token” on page 127</a>
path_attr	Access path information	<a href="#">“path_attr Token” on page 128</a>
privilege	Privilege set information	<a href="#">“privilege Token” on page 128</a>
process	Process information	<a href="#">“process Token” on page 128</a>
return	Status of system call	<a href="#">“return Token” on page 128</a>
sequence	Sequence number	<a href="#">“sequence Token” on page 129</a>
socket	Socket type and addresses	<a href="#">“socket Token” on page 129</a>
subject	Subject information (same format as process)	<a href="#">“subject Token” on page 129</a>
text	ASCII string	<a href="#">“text Token” on page 130</a>
trailer	Indicates end of audit record	<a href="#">“trailer Token” on page 130</a>
use of authorization	Use of authorization	<a href="#">“use of authorization Token” on page 130</a>
use of privilege	Use of privilege	<a href="#">“use of privilege Token” on page 131</a>
user	User ID and user name	<a href="#">“user Token” on page 131</a>
xclient	X client identification	<a href="#">“xclient Token” on page 131</a>
zonename	Name of zone	<a href="#">“zonename Token” on page 131</a>
Trusted Extensions tokens	label and X Window System information	<a href="#">“Trusted Extensions Audit Reference” in <i>Trusted Extensions Configuration and Administration</i></a>

The following tokens are obsolete:

- liaison
- host
- tid

For information about obsolete tokens, see the reference material for the release that included the token.

An audit record always begins with a header token, which indicates where the audit record begins in the audit trail. In the case of attributable events, the subject and the process tokens refer to the values of the process that caused the event. In the case of non-attributable events, the process token refers to the system.

## acl Token

The `acl` token uses different formats to record information about Access Control Entries (ACEs) for a ZFS file system and about Access Control Lists (ACLs) for a legacy UFS file system.

When the `acl` token is recorded for a UFS file system, the `praudit` command shows the fields as follows:

```
<acl type="1" value="root" mode="6"/>
```

When the `acl` token is recorded for a ZFS dataset, the `praudit` command shows the fields as follows:

```
<acl who="root" access_mask="default" flags="-i,-R" type="2"/>
```

## argument Token

The `argument` token contains information about the arguments to a system call: the argument number of the system call, the argument value, and an optional description. This token allows a 32-bit integer system-call argument in an audit record.

The `praudit` command shows the fields of the `argument` token as follows:

```
<argument arg-num="2" value="0x5401" desc="cmd"/>
```

## attribute Token

The `attribute` token contains information from the file `vnode`.

The `attribute` token usually accompanies a `path` token. The `attribute` token is produced during path searches. If a path-search error occurs, `vnode` is not available to obtain the necessary file information. Therefore, the `attribute` token is not included as part of the audit record. The `praudit` command shows the fields of the `attribute` token as follows:

```
<attribute mode="20620" uid="root" gid="tty" fsid="0" nodeid="9267" device="108233"/>
```

## cmd Token

The cmd token records the list of arguments and the list of environment variables that are associated with a command.

The praudit command shows the fields of the cmd token. The following example is a truncated cmd token. The line is wrapped for display purposes.

```
<cmd><arg>WINDOWID=6823679</arg>  
<arg>COLORTERM=gnome-terminal</arg>  
<arg>...LANG=C</arg>...<arg>HOST=machine1</arg>  
<arg>LPDEST=printer1</arg>...</cmd>
```

## exec\_args Token

The exec\_args token records the arguments to an exec() system call.

The praudit command shows the fields of the exec\_args token as follows:

```
<exec_args><arg>/usr/bin/sh</arg><arg>/usr/bin/hostname</arg></exec_args>
```

---

**Note** - The exec\_args token is output only when the argv audit policy option is active.

---

## exec\_env Token

The exec\_env token records the current environment variables to an exec() system call.

The praudit command shows the fields of the exec\_env token. The line in the following example is wrapped for display purposes.

```
<exec_env><env>_/usr/bin/hostname</env>  
<env>LANG=C</env><env>PATH=/usr/bin</env>  
<env>LOGNAME=jdoe</env><env>USER=jdoe</env>  
<env>DISPLAY=:0</env><env>SHELL=/bin/csh</env>  
<env>HOME=/home/jdoe</env><env>PWD=/home/jdoe</env><env>TZ=US/Pacific</env>  
</exec_env>
```

---

**Note** - The exec\_env token is output only when the arge audit policy option is active.

---

## file Token

The file token is a special token that marks the beginning of a new audit file and the end of an old audit file as the old file is deactivated. The initial file token identifies the previous file in the audit trail. The final file token identifies the next file in the audit trail. These tokens link successive audit files into one audit trail.

The `praudit` command shows the fields of the file token. The line in the following example is wrapped for display purposes.

```
<file iso8601="2009-04-08 14:18:26.200 -07:00">
/var/audit/machine1/files/20090408211826.not_terminated.machine1</file>
```

## fmri Token

The fmri token records the use of a fault management resource indicator (FMRI). For more information, see the [fmri\(5\)](#) man page.

The `praudit` command shows the content of the fmri token as follows:

```
<fmri service_instance="svc:/system/cryptosvc"</fmri>
```

## group Token

The group token records the group entries from the process's credential. The group token is output only when the group audit policy option is active.

The `praudit` command shows the fields of the group token as follows:

```
<group><gid>staff</gid><gid>other</gid></group>
```

## header Token

The header token is special in that it marks the beginning of an audit record. The header token combines with the trailer token to bracket all the other tokens in the record.

Infrequently, a header token can include one or more event modifiers:

- fe indicates a failed audit event
- fp indicates the failed use of privilege

- `na` indicates a non-attributable event

```
header,52,2,system booted,na,mach1,2011-10-10 10:10:20.564 -07:00
```

- `rd` indicates that data is read from the object
- `sp` indicates the successful use of privilege

```
header,120,2,exit(2),sp,mach1,2011-10-10 10:10:10.853 -07:00
```

- `wr` indicates that data is written to the object

The `praudit` command displays the header token as follows:

```
header,756,2,execve(2),,machine1,2010-10-10 12:11:10.209 -07:00
```

The `praudit` command displays the fields of the header token at the beginning of the audit record. The line in the following example is wrapped for display purposes.

```
<record version="2" event="execve(2)" host="machine1"
iso8601="2010-10-10 12:11:10.209 -07:00">
```

## ip address Token

The `ip address` token contains an Internet Protocol address (IP address). The IP address can be displayed in IPv4 or IPv6 format. The IPv4 address uses 4 bytes. The IPv6 address uses 16 bytes to describe the address type, and 16 bytes to describe the address.

The `praudit` command shows the content of the `ip address` token as follows:

```
<ip_address>machine1</ip_address>
```

## ip port Token

The `ip port` token contains the TCP or UDP port address.

The `praudit` command displays the `ip port` token as follows:

```
ip port,0xf6d6
```

## ipc Token

The `ipc` token contains the System V IPC message handle, semaphore handle, or shared-memory handle that is used by the caller to identify a particular IPC object.

The IPC object identifiers violate the context-free nature of the audit tokens. No global “name” uniquely identifies IPC objects. Instead, IPC objects are identified by their handles. The handles are valid only during the time that the IPC objects are active. However, the identification of IPC objects should not be a problem. The System V IPC mechanisms are seldom used, and the mechanisms all share the same audit class.

The following table shows the possible values for the IPC object type field. The values are defined in the `/usr/include/bsm/audit.h` file.

**TABLE 9** Values for the 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

The `praudit` command shows the fields of the `ipc` token as follows:

```
<IPC ipc-type="shm" ipc-id="15"/>
```

## IPC\_perm Token

The `IPC_perm` token contains a copy of the System V IPC access permissions. This token is added to audit records that are generated by IPC shared-memory events, IPC semaphore events, and IPC message events.

The `praudit` command shows the fields of the `IPC_perm` token. The line in the following example is wrapped for display purposes.

```
<IPC_perm uid="jdoe" gid="staff" creator-uid="jdoe"
creator-gid="staff" mode="100600" seq="0" key="0x0"/>
```

The values are taken from the `IPC_perm` structure that is associated with the IPC object.

## path Token

The `path` token contains access path information for an object.

The `praudit` command shows the content of the `path` token as follows:

```
<path>/export/home/srv/.xsession-errors</path>
```

## path\_attr Token

The `path_attr` token contains access path information for an object. The access path specifies the sequence of attribute file objects below the path token object. Systems calls such as `openat()` access attribute files. For more information about attribute file objects, see the [fsattr\(5\)](#) man page.

The `praudit` command displays the `path_attr` token as follows:

```
path_attr,1,attr_file_name
```

## privilege Token

The privilege token records the use of privileges on a process. The privilege token is not recorded for privileges in the basic set. If a privilege has been removed from the basic set by administrative action, then the use of that privilege is recorded. For more information about privileges, see “[Process Rights Management](#)” in *Securing Users and Processes in Oracle Solaris 11.3*.

The `praudit` command shows the fields of the privilege token.

```
<privilege set-type="Inheritable">ALL</privilege>
```

## process Token

The process token contains information about a user who is associated with a process, such as the recipient of a signal.

The `praudit` command shows the fields of the process token. The line in the following example is wrapped for display purposes.

```
<process audit-uid="-2" uid="root" gid="root" ruid="root"  
rgid="root" pid="567" sid="0" tid="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 return token is always returned as part of kernel-generated audit records for system calls. In application auditing, this token indicates exit status and other return values.

The `praudit` command displays the return token for a system call as follows:

```
return,failure: Operation now in progress,-1
```

The `praudit` command shows the fields of the return token as follows:

```
<return errval="failure: Operation now in progress" retval="-1/">
```

## sequence Token

The sequence token contains a sequence number. The sequence number is incremented every time an audit record is added to the audit trail. The sequence token is output only when the `seq` audit policy option is active. This token is useful for debugging.

The `praudit` command shows the content of the sequence token:

```
<sequence seq-num="1292" />
```

## socket Token

The socket token contains information that describes an Internet socket. In some circumstances, the token includes only the remote port and remote IP address.

The `praudit` command displays this instance of the socket token as follows:

```
socket,0x0002,0x83b1,localhost
```

The expanded token adds information, including socket type and local port information.

The `praudit` command displays this instance of the socket token as follows. The line in the following example is wrapped for display purposes.

```
<socket sock_domain="0x0002" sock_type="0x0002" lport="0x83cf"  
laddr="example1" fport="0x2383" faddr="server1.Subdomain.Domain.COM"/>
```

## subject Token

The subject token describes a user who performs or attempts to perform an operation. The format is the same as the process token.

The subject token is always returned as part of kernel-generated audit records for system calls. The `praudit` command displays the subject token as follows:

```
subject,jdoe,root,root,root,root,1631,1421584480,8243 65558 machine1
```

The `praudit` command shows the fields of the subject token. The line in the following example is wrapped for display purposes.

```
<subject audit-uid="jdoe" uid="root" gid="root" ruid="root"
rgid="root" pid="1631" sid="1421584480" tid="8243 65558 machine1"/>
```

## text Token

The text token contains a text string.

The `praudit` command shows the content of the text token as follows:

```
<text>booting kernel</text>
```

## trailer Token

The two tokens, header and trailer, are special in that they distinguish the beginning and end points of an audit record and bracket all the other tokens. A header token begins an audit record. A trailer token ends an audit record. The trailer token is an optional token, and is added as the last token of each record only when the `trail` audit policy option has been set.

When an audit record is generated with trailers turned on, the `auditreduce` command can verify that the trailer token correctly points back to the record header. The trailer token supports backward seeks of the audit trail.

The `praudit` command displays the trailer token as follows:

```
trailer,136
```

## use of authorization Token

The `use of authorization` token records the use of authorizations.

The `praudit` command displays the `use of authorization` token as follows:

```
use of authorization,solaris.role.delegate
```

## use of privilege Token

The `use of privilege` token records the use of privileges.

The `praudit` command shows the fields of the `use of privilege` token as follows:

```
<use_of_privilege result="successful use of priv">proc_setid</use_of_privilege>
```

## user Token

The `user` token records the user name and user ID. This token is present if the user name is different from the caller.

The `praudit` command shows the fields of the `user` token as follows:

```
<user uid="123456" username="tester1"/>
```

## xclient Token

The `xclient` token contains the number of the client connection to the X server.

The `praudit` command shows the content of the `xclient` token as follows:

```
<X_client>15</X_client>
```

## zonename Token

The `zonename` token records the zone in which the audit event occurred. The string “global” indicates audit events that occur in the global zone.

The `praudit` command shows the content of the `zonename` token as follows:

```
<zone name="graphzone"/>
```



## Glossary

---

<b>Access Control List (ACL)</b>	A list associated with a file that contains information about which users or groups have permission to access or modify the file. An access control list (ACL) provides finer-grained file security than traditional UNIX file protection provides. For example, an ACL enables you to allow group read access to a file, while allowing only one member of that group to write to the file.
<b>admin principal</b>	A user principal with a name of the form <i>username/admin</i> (as in <i>jdoe/admin</i> ). An admin principal can have more privileges (for example, to change policies) than a regular user principal. See also <a href="#">principal name</a> and <a href="#">user principal</a> .
<b>algorithm</b>	A cryptographic algorithm. This is an established, recursive computational procedure that encrypts or hashes input.
<b>asynchronous audit event</b>	Asynchronous events are the minority of system events. These events are not associated with any process, so no process is available to be blocked and later woken up. Initial system boot and PROM enter and exit events are examples of asynchronous events.
<b>audit files</b>	Binary audit logs. Audit files are stored separately in an audit file system.
<b>audit policy</b>	The global and per-user settings that determine which audit events are recorded. The global settings that apply to the audit service typically affect which pieces of optional information are included in the audit trail. Two settings, <code>cnt</code> and <code>ahlt</code> , affect the operation of the system when the audit queue fills. For example, audit policy might require that a sequence number be part of every audit record.
<b>audit trail</b>	The collection of all audit files from all systems.
<b>authentication</b>	The process of verifying the claimed identity of a principal.
<b>authorization</b>	<ol style="list-style-type: none"><li>1. In Kerberos, the process of determining if a principal can use a service, which objects the principal is allowed to access, and the type of access that is allowed for each object.</li><li>2. In user rights management, a right that can be assigned to a role or user (or embedded in a rights profile) for performing a class of operations that are otherwise prohibited by security policy. Authorizations are enforced at the user application level, not in the kernel.</li></ol>
<b>basic set</b>	The set of privileges that are assigned to a user's process at login. On an unmodified system, each user's initial inheritable set equals the basic set at login.

<b>client</b>	<p>Narrowly, a system process that makes use of a network service on behalf of a user; for example, an application that uses <code>rlogin</code>. In some cases, a network server can itself be a client of some other server or service.</p> <p>More broadly, a system that a) receives a Kerberos credential, and b) makes use of a service that is provided by a network server.</p> <p>Informally, a principal that makes use of a service.</p>
<b>clock skew</b>	<p>The maximum amount of time that the internal system clocks on all systems that are participating in the Kerberos authentication system can differ. If the clock skew is exceeded between any of the participating systems, requests are rejected. Clock skew can be specified in the <code>krb5.conf</code> file.</p>
<b>credential</b>	<p>An information package that includes a ticket and a matching session key. Used to authenticate the identity of a principal. See also <a href="#">session key</a>.</p>
<b>GSS-API</b>	<p>The Generic Security Service Application Programming Interface. A network layer that provides support for various modular security services, including the Kerberos service. GSS-API provides for security authentication, integrity, and privacy services. See also <a href="#">authentication</a>, <a href="#">integrity</a>, and <a href="#">privacy</a>.</p>
<b>host</b>	<p>A system that is accessible over a network.</p>
<b>host principal</b>	<p>A particular instance of a service principal in which the principal (signified by the primary name <code>host</code>) is set up to provide a range of network services, such as <code>ftp</code>, <code>rcp</code>, or <code>rlogin</code>. An example of a host principal is <code>host/central.example.com@EXAMPLE.COM</code>. See also <a href="#">server principal</a>.</p>
<b>inheritable set</b>	<p>The set of privileges that a process can inherit across a call to <code>exec</code>.</p>
<b>instance</b>	<p>In the context of network communications, an instance is the second part of a principal name. An instance qualifies the principal's primary. In the case of a service principal, the instance is required. The instance is the host's fully qualified domain name, as in <code>host/central.example.com</code>. For user principals, an instance is optional. Note, however, that <code>jdoe</code> and <code>jdoe/admin</code> are unique principals. See also <a href="#">primary</a>, <a href="#">principal name</a>, <a href="#">service principal</a>, and <a href="#">user principal</a>.</p>
<b>integrity</b>	<p>A security service that, in addition to user authentication, provides for the validity of transmitted data through cryptographic checksumming. See also <a href="#">authentication</a> and <a href="#">privacy</a>.</p>
<b>KDC</b>	<p>Key Distribution Center. A system that has three Kerberos V5 components:</p> <ul style="list-style-type: none"><li>■ Principal and key database</li><li>■ Authentication service</li><li>■ Ticket-granting service</li></ul> <p>Each realm has a master KDC and should have one or more slave KDCs.</p>

<b>Kerberos</b>	<p>An authentication service, the protocol that is used by that service, or the code that is used to implement that service.</p> <p>The Kerberos implementation in Oracle Solaris that is closely based on Kerberos V5 implementation.</p> <p>While technically different, “Kerberos” and “Kerberos V5” are often used interchangeably in the Kerberos documentation.</p> <p>Kerberos (also spelled Cerberus) was a fierce, three-headed mastiff who guarded the gates of Hades in Greek mythology.</p>
<b>key</b>	<ol style="list-style-type: none"><li>1. Generally, one of two main types of keys:<ul style="list-style-type: none"><li>■ A <i>symmetric key</i> – An encryption key that is identical to the decryption key. Symmetric keys are used to encrypt files.</li><li>■ An <i>asymmetric key</i> or <i>public key</i> – A key that is used in public key algorithms, such as Diffie-Hellman or RSA. Public keys include a private key that is known only by one user, a public key that is used by the network server or general resource, and a private-public key pair that combines the two. A private key is also called a <i>secret</i> key. The public key is also called a <i>shared</i> key or <i>common</i> key.</li></ul></li><li>2. An entry (principal name) in a keytab file. See also <a href="#">keytab file</a>.</li><li>3. In Kerberos, an encryption key, of which there are three types:<ul style="list-style-type: none"><li>■ A <i>private key</i> – An encryption key that is shared by a principal and the KDC, and distributed outside the bounds of the system. See also <a href="#">private key</a>.</li><li>■ A <i>service key</i> – This key serves the same purpose as the private key, but is used by network servers and services. See also <a href="#">service key</a>.</li><li>■ A <i>session key</i> – A temporary encryption key that is used between two principals, with a lifetime limited to the duration of a single login session. See also <a href="#">session key</a>.</li></ul></li></ol>
<b>keystore</b>	<p>A keystore holds passwords, passphrases, certificates, and other authentication objects for retrieval by applications. A keystore can be specific to a technology, or a location that several applications use.</p>
<b>keytab file</b>	<p>A key table file that contains one or more keys (principals). A system or service uses a keytab file in the much the same way that a user uses a password.</p>
<b>master KDC</b>	<p>The main KDC in each realm, which includes a Kerberos administration server, <code>kadmin</code>, and an authentication and ticket-granting daemon, <code>krb5kdc</code>. Each realm must have at least one master KDC, and can have many duplicate, or slave, KDCs that provide authentication services to client systems.</p>
<b>mechanism</b>	<ol style="list-style-type: none"><li>1. A software package that specifies cryptographic techniques to achieve data authentication or confidentiality. Examples: Kerberos V5, Diffie-Hellman public key.</li></ol>

2. In the Cryptographic Framework feature of Oracle Solaris, an implementation of an algorithm for a particular purpose. For example, a DES mechanism that is applied to authentication, such as CKM\_DES\_MAC, is a separate mechanism from a DES mechanism that is applied to encryption, CKM\_DES\_CBC\_PAD.

**nonattributable audit event** An audit event whose initiator cannot be determined, such as the AUE\_BOOT event.

**NTP** Network Time Protocol. Software from the University of Delaware that enables you to manage precise time or network clock synchronization, or both, in a network environment. You can use NTP to maintain clock skew in a Kerberos environment. See also clock skew.

**PAM** Pluggable Authentication Module. A framework that allows for multiple authentication mechanisms to be used without having to recompile the services that use them. PAM enables Kerberos session initialization at login.

**passphrase** A phrase that is used to verify that a private key was created by the passphrase user. A good passphrase is 10-30 characters long, mixes alphabetic and numeric characters, and avoids simple prose and simple names. You are prompted for the passphrase to authenticate use of the private key to encrypt and decrypt communications.

**policy** Generally, a plan or course of action that influences or determines decisions and actions. For computer systems, policy typically means security policy. Your site's security policy is the set of rules that define the sensitivity of the information that is being processed and the measures that are used to protect the information from unauthorized access. For example, security policy might require that systems be audited, that devices must be allocated for use, and that passwords be changed every six weeks.

See [audit policy](#).

**primary** The first part of a principal name. See also [instance](#), [principal name](#), and [realm](#).

**principal** 1. A uniquely named client/user or server/service instance that participates in a network communication. Kerberos transactions involve interactions between principals (service principals and user principals) or between principals and KDCs. In other words, a principal is a unique entity to which Kerberos can assign tickets. See also [principal name](#), [service principal](#), and [user principal](#).

2. (RPCSEC\_GSS API) See [server principal](#).

**principal name** 1. The name of a principal, in the format *primary/instance@REALM*. See also [instance](#), [primary](#) and [realm](#).

2. (RPCSEC\_GSS API) See [server principal](#).

**privacy** A security service, in which transmitted data is encrypted before being sent. Privacy also includes data integrity and user authentication. See also [authentication](#), [integrity](#), and [service](#).



<b>private key</b>	A key that is given to each user principal, and known only to the user of the principal and to the KDC. For user principals, the key is based on the user's password. See also <a href="#">key</a> .
<b>private-key encryption</b>	In private-key encryption, the sender and receiver use the same key for encryption. See also <a href="#">public-key encryption</a> .
<b>privilege</b>	<ol style="list-style-type: none"><li>1. In general, a power or capability to perform an operation on a computer system that is beyond the powers of a regular user. Superuser privileges are all the <a href="#">rights</a> that superuser is granted. A privileged user or privileged application is a user or application that has been granted additional rights.</li><li>2. A discrete right on a process in an Oracle Solaris system. Privileges offer a finer-grained control of processes than does <code>root</code>. Privileges are defined and enforced in the kernel. Privileges are also called <i>process privileges</i> or <i>kernel privileges</i>. For a full description of privileges, see the <a href="#">privileges(5)</a> man page.</li></ol>
<b>privilege set</b>	<p>A collection of privileges. Every process has four sets of privileges that determine whether a process can use a particular privilege. See <a href="#">inheritable set</a> set.</p> <p>Also, the <a href="#">basic set</a> set of privileges is the collection of privileges that are assigned to a user's process at login.</p>
<b>privileged application</b>	An application that can override system controls. The application checks for security attributes, such as specific UIDs, GIDs, authorizations, or privileges.
<b>privileged user</b>	A user who is assigned rights beyond the rights of regular user on a computer system.
<b>profile shell</b>	In rights management, a shell that enables a role (or user) to run from the command line any privileged applications that are assigned to the role's rights profiles. The profile shell versions correspond to the available shells on the system, such as the <code>pfbash</code> version of <code>bash</code> .
<b>public object</b>	A file that is owned by the <code>root</code> user and readable by the world, such as any file in the <code>/etc</code> directory.
<b>public-key encryption</b>	An encryption scheme in which each user has two keys, one public key and one private key. In public-key encryption, the sender uses the receiver's public key to encrypt the message, and the receiver uses a private key to decrypt it. The Kerberos service is a private-key system. See also <a href="#">private-key encryption</a> .
<b>realm</b>	<ol style="list-style-type: none"><li>1. The logical network that is served by a single Kerberos database and a set of Key Distribution Centers (KDCs).</li><li>2. The third part of a principal name. For the principal name <code>jdoh/admin@CORP.EXAMPLE.COM</code>, the realm is <code>CORP.EXAMPLE.COM</code>. See also <a href="#">principal name</a>.</li></ol>
<b>rights</b>	An alternative to the all-or-nothing superuser model. User rights management and process rights management enable an organization to divide up superuser's privileges and assign them to users or roles. Rights in Oracle Solaris are implemented as kernel privileges, authorizations,

and the ability to run a process as a specific UID or GID. Rights can be collected in a [rights profile](#) and a [role](#).

<b>rights profile</b>	Also referred to as a profile. A collection of security overrides that can be assigned to a role or user. A rights profile can include authorizations, privileges, commands with security attributes, and other rights profiles that are called supplementary profiles.
<b>role</b>	A special identity for running privileged applications that only assigned users can assume.
<b>secret key</b>	See <a href="#">private key</a> .
<b>Secure Shell</b>	A special protocol for secure remote login and other secure network services over an insecure network.
<b>security attributes</b>	Overrides to security policy that enable an administrative command to succeed when the command is run by a user other than superuser. In the superuser model, the <code>setuid root</code> and <code>setgid</code> programs are security attributes. When these attributes are applied to a command, the command succeeds no matter who runs the command. In the privilege model, kernel privileges and other <a href="#">rights</a> replace <code>setuid root</code> programs as security attributes. The privilege model is compatible with the superuser model, in that the privilege model also recognizes the <code>setuid</code> and <code>setgid</code> programs as security attributes.
<b>security mechanism</b>	See <a href="#">mechanism</a> .
<b>security policy</b>	See <a href="#">policy</a> .
<b>security service</b>	See <a href="#">service</a> .
<b>server</b>	A principal that provides a resource to network clients. For example, if you <code>ssh</code> to the system <code>central.example.com</code> , then that system is the network server that provides the <code>ssh</code> service. See also <a href="#">service principal</a> .
<b>server principal</b>	(RPCSEC_GSS API) A principal that provides a service. The server principal is stored as an ASCII string in the form <code>service@host</code> .
<b>service</b>	<ol style="list-style-type: none"><li>1. A resource that is provided to network clients, often by more than one network server. For example, if you <code>rlogin</code> to the system <code>central.example.com</code>, then that system is the server that provides the <code>rlogin</code> service.</li><li>2. A security service (either integrity or privacy) that provides a level of protection beyond authentication. See also <a href="#">integrity</a> and <a href="#">privacy</a>.</li></ol>
<b>service key</b>	An encryption key that is shared by a service principal and the KDC, and is distributed outside the bounds of the system. See also <a href="#">key</a> .
<b>service principal</b>	A principal that provides Kerberos authentication for a service or services. For service principals, the primary name is a name of a service, such as <code>ftp</code> , and its instance is the fully

qualified host name of the system that provides the service. See also [host principal](#) and [user principal](#).

<b>session key</b>	A key that is generated by the authentication service or the ticket-granting service. A session key is generated to provide secure transactions between a client system and a service. The lifetime of a session key is limited to a single login session. See also <a href="#">key</a> .
<b>single-system image</b>	A single-system image is used in Oracle Solaris auditing to describe a group of audited systems that use the same naming service. These systems send their audit records to a central audit server, where the records can be compared as if the records came from one system.
<b>slave KDC</b>	A copy of a master KDC, which is capable of performing most functions of the master. Each realm usually has several slave KDCs (and only one master KDC). See also <a href="#">KDC</a> and <a href="#">master KDC</a> .
<b>synchronous audit event</b>	The majority of audit events. These events are associated with a process in the system. A non-attributable event that is associated with a process is a synchronous event, such as a failed login.
<b>user principal</b>	A principal that is attributed to a particular user. A user principal's primary name is a user name, and its optional instance is a name that is used to describe the intended use of the corresponding credentials (for example, jdoe or jdoe/admin). Also known as a user instance. See also <a href="#">service principal</a> .



# Index

---

## Numbers and Symbols

- + (plus sign) in audit class prefixes, 86, 115
- (minus sign)
  - audit class prefix, 115
- [] (square brackets)
  - auditrecord output, 121
- ^ (caret)
  - audit class prefix modifier, 115
  - in audit class prefixes, 45

## A

- a option
  - auditrecord command, 89
- A option
  - auditreduce command, 98
- acl audit token
  - format, 123
- active audit policy
  - temporary audit policy, 48
- adding
  - audit classes, 54, 54
  - audit file systems, 74
  - audit policy, 48
  - auditing
    - of individual users, 45, 108
    - of zones, 27
  - plugins
    - auditing, 80, 82, 86
  - temporary audit policy, 50
- administering auditing
  - audit -s command, 42, 69
  - audit -t command, 42
  - audit classes, 15
  - audit events, 14
  - audit files, 93
  - audit records, 16
  - audit trail overflow prevention, 100
  - audit\_remote plugin, 80, 82
  - audit\_syslog plugin, 86
  - auditconfig command, 42, 44
  - auditreduce command, 97
  - configuring, 42
  - cost control, 36
  - description, 23
  - disabling, 42
  - efficiency, 38
  - enabling, 42
  - in zones, 26, 28, 65, 114
  - plugins, 80, 82
  - policy, 48
  - praudit command, 93
  - queue controls, 51
  - reducing space requirements, 37
  - refreshing, 69
  - rights profiles required, 114
  - reports, 24
- ahlt audit policy
  - description, 34
  - setting, 50
  - with cnt policy, 118
- all audit class
  - caution for using, 115
- always-audit* classes
  - process preselection mask, 119
- archiving
  - audit files, 100
- arge audit policy
  - and exec\_env token, 124
  - description, 34
  - setting, 59
- argument audit token

- format, 123
- argv audit policy
  - and exec\_args token, 124
  - description, 34
  - setting, 58
- ARS *See* audit remote server
- asynchronous audit events, 118, 118
- attribute audit token, 123
- audit -s command, 42, 69, 69
- audit -t command, 42
- audit characteristics
  - audit user ID, 119
  - processes, 119
  - session ID, 119
  - terminal ID, 119
  - user process preselection mask, 119
- audit classes
  - adding, 54
  - configuration, 115
  - cusa, 51
  - description, 12, 14
  - displaying defaults, 40
  - exceptions to system-wide settings, 16
  - mapping events, 16
  - modifying default, 54
  - overview, 15
  - post-selection, 13
  - prefixes, 115
  - preselecting
    - effect on public objects, 14
    - for failure, 46, 86, 87
    - for success, 46, 86, 87
    - for success and failure, 44
  - preselection, 14
  - process preselection mask, 119
  - replacing, 44
  - syntax, 115, 115
  - user exceptions, 45
- audit command
  - disabling audit service, 42
  - options, 112
  - refreshing audit service, 69
- Audit Configuration rights profile, 114
  - configuring audit policy, 48
  - displaying auditing defaults, 40
  - preselecting audit classes, 44
- Audit Control rights profile, 114
  - disabling audit service, 42
  - enabling audit service, 42
  - refreshing audit service, 69
- audit directory
  - creating file systems for, 74
- audit event-to-class mappings
  - changing, 55
- audit events
  - asynchronous, 118
  - audit\_event file and, 14
  - changing class membership, 55
  - description, 14
  - mapping to classes, 16
  - removing from audit\_event file, 62
  - selecting from audit trail, 91
  - selecting from audit trail in zones, 114
  - summary, 12
  - synchronous, 118
  - viewing from binary files, 93
- audit file system
  - description, 12
- audit files
  - combining, 97
  - compressing on disk, 63
  - copying messages to single file, 93
  - creating summary files, 92, 92, 93
  - effects of Coordinated Universal Time (UTC), 98
  - limiting size of, 108
  - managing, 100
  - printing, 96
  - reading with praudit , 93
  - reducing size of, 97
  - reducing space requirements, 37
  - reducing storage-space requirements, 38
  - setting aside disk space for, 74
  - time stamps, 120
  - ZFS file systems, 63, 74
- audit flags
  - summary of, 12
- audit logs, 11
  - See also* audit files
  - comparing binary and text summaries, 18
  - configuring, 73
  - configuring text summary audit logs, 86
  - modes, 18

- audit plugins
  - audit\_binfile plugin, 51, 77
  - audit\_remote plugin, 80, 82
  - audit\_syslog plugin, 86
  - description, 13
  - qsize attribute, 51
  - summary of, 112, 116
- audit policy
  - audit tokens from, 117
  - defaults, 34
  - description, 13
  - displaying defaults, 40
  - effects of, 34
  - public, 35
  - setting, 48
  - setting ahl\_t, 50
  - setting arge, 59
  - setting argv, 58
  - setting in global zone, 26, 114
  - setting perzone, 50
  - that does not affect tokens, 117
  - tokens added by, 117
- audit preselection mask
  - modifying for existing users, 61
  - modifying for individual users, 45
- audit queue
  - events included, 16
- audit queue controls
  - displaying defaults, 40
  - getting, 51
- audit records
  - converting to readable format, 96
  - copying to single file, 93
  - description, 13
  - displaying, 93
  - displaying definitions of
    - procedure, 89
  - displaying formats of a program, 90
  - displaying formats of an audit class, 90
  - displaying in XML format, 94
  - event modifiers, 125
  - events that generate, 22
  - format, 120
  - formatting example, 90
  - merging, 97
  - overview, 16
  - policies that add tokens to, 117
  - reducing audit file size, 97
  - sequence of tokens, 120
  - /var/adm/auditlog file, 87
- audit remote server
  - managing, 21
  - overview, 17
  - summary of, 117
- Audit Review rights profile, 114
- audit service, 11
  - See also* auditing
  - audit trail creation, 120
  - configuring policy, 48
  - configuring queue controls, 51
  - defaults, 111
  - disabling, 42
  - enabling, 42
  - policy, 34
  - refreshing the kernel, 69
  - troubleshooting, 104
- audit session ID, 119
  - overview, 11
- audit tokens, 11
  - See also* individual audit token names
  - added by audit policy, 117
  - audit record format, 120
  - description, 13, 17
  - format, 121
  - list of, 121
  - xclient token, 131
- audit trail
  - adding disk space, 77
  - analysis costs, 36
  - cleaning up not\_terminated files, 99
  - creating summary files, 92, 92
  - description, 13
  - effect of audit policy, 34
  - monitoring in real time, 38
  - overview, 24
  - preventing overflow, 100
  - reducing size of, 63, 106
  - selecting events from, 91
  - sending files to remote repository, 80, 82
  - viewing events from, 93
  - viewing events from different zones, 114

- audit user ID
  - mechanism, 119
  - overview, 11
- audit.notice entry
  - syslog.conf file, 87
- audit\_binfile plugin, 17
  - getting attributes, 78, 79, 79
  - limiting audit file size, 78
  - removing queue size, 79
  - setting attributes, 77
  - setting free space warning, 80
  - specifying time for log rotation, 78
- audit\_class file
  - adding a class, 54
  - troubleshooting, 55
- audit\_event file
  - changing class membership, 55
  - description, 14
  - removing events safely, 62
- audit\_flags keyword, 45
  - specifying user exceptions to audit preselection, 45
  - use, 116
  - using caret (^) prefix, 46
- audit\_remote plugin, 17
  - configuring, 82
  - getting attributes, 80, 82
  - setting attributes, 80, 82
  - troubleshooting audit queue overfull, 81
- audit\_syslog plugin, 17
  - setting attributes, 86
- audit\_warn script
  - configuring, 53
  - description, 113
- auditconfig command
  - adding audit file systems, 77
  - audit classes as arguments, 16
  - configuring policy, 48
  - configuring queue controls, 51
  - description, 113
  - displaying audit defaults, 40
  - getplugin option, 80, 82, 86
  - policy options, 48
  - preselecting audit classes, 44
  - queue control options, 51
  - sending files to remote repository, 80, 82
  - setflags option, 44
  - setnaflags option, 44
  - setplugin option, 80, 82, 86
  - setting active audit policy, 50
  - setting audit policy, 58
  - setting audit policy temporarily, 50
  - setting audit\_binfile attributes, 77
  - setting audit\_remote attributes, 80, 82
  - setting system-wide audit parameters, 16
  - viewing default audit preselection, 44
- auditd daemon
  - refreshing audit service, 70
- auditing
  - adding audit flags to a group of users, 48
  - all commands by users, 57
  - analysis, 24
  - Audit Remote Server (ARS), 21
  - configuring
    - all zones, 42
    - global zone, 50
    - identically for all zones, 66
    - per zone, 68
  - configuring in global zone, 28
  - customizing, 56
  - default configuration, 39
  - defaults, 111
  - determining if running, 104
  - disabling, 42
  - enabling, 42
  - finding changes to specific files, 59
  - getting queue controls, 51
  - local definition, 13
  - logins, 109
  - man page summaries, 112
  - planning, 27
  - planning in zones, 28, 28
  - plugin modules, 17
  - plugin to Oracle Audit Vault and Database Firewall, 24
  - post-selection definition, 13
  - preselection definition, 14
  - remote definition, 14
  - removing user-specific audit flags, 48
  - reports, 24
  - rights profiles for, 114
  - setting queue controls, 51



- sftp file transfers, 64
- troubleshooting, 103
- troubleshooting praudit command, 97
- updating information, 69, 69
- users only, 47
- zones and, 26, 114
- auditlog file
  - text audit records, 87
- auditrecord command
  - [ ] (square brackets) in output, 121
  - description, 113
  - displaying audit record definitions, 89
  - example, 90
  - listing all formats, 89
  - listing formats of class, 90
  - listing formats of program, 90
  - optional tokens ([ ]), 121
- auditreduce command
  - A option, 98
  - b option, 92
  - c option, 93, 93
  - C option, 99
  - cleaning up audit files, 99
  - d option, 93
  - description, 113
  - e option, 92
  - examples, 97
  - filtering options, 91
  - M option, 99
  - merging audit records, 97
  - O option, 92, 97, 99
  - selecting audit records, 91
  - time stamp use, 98
  - trailer tokens, and, 130
  - using lowercase options, 91
  - using uppercase options, 98
- auditstat command
  - description, 113

## B

- b option
  - auditreduce command, 92
- binary and remote records, 19

## C

- c option
  - auditrecord command, 90
  - auditreduce command, 93
- C option
  - auditreduce command, 99
- caret (^)
  - in audit class prefixes, 45
  - using prefix in audit\_flags value, 46
- changing
  - audit\_class file, 54
  - audit\_event file, 55
  - auditing defaults, 44
- classes *See* audit classes
- cleaning up
  - binary audit files, 99
- cmd audit token, 124
- cnt audit policy
  - description, 35
  - with ahl\_t policy, 118
- combining audit files
  - auditreduce command, 97
  - from different zones, 114
- compressing
  - audit files on disk, 63
- configuration decisions
  - auditing
    - file storage, 32
    - policy, 34
    - remote file storage, 33
    - who and what to audit, 30
    - zones, 28
- configuration files
  - auditing, 112
- configured audit policy
  - permanent audit policy, 48
- configuring
  - active audit policy, 50
  - ahl\_t audit policy, 50
  - audit classes, 44
  - audit logs task map, 73
  - audit policy, 48
  - audit policy temporarily, 50
  - audit queue controls, 51
  - audit service policy, 48

- audit trail overflow prevention, 100
- audit\_class file, 54
- audit\_event file, 55
- audit\_warn script, 53
- auditing, 42
- auditing in zones, 26, 114
- auditing reports, 24
- auditing task map, 42
- identical auditing for non-global zones, 66
- per-zone auditing, 68
- permanent audit policy, 48
- perzone audit policy, 50
- space for audit trail, 77
- temporary audit policy, 48
- text summaries of audit records, 86
- converting
  - audit records to readable format, 96
- Coordinated Universal Time (UTC)
  - time stamp use in auditing, 98, 120
- copying audit records to single file, 93
- cost control
  - and auditing, 36
- creating
  - audit trail, 120
  - rights profile for a group of users, 48
  - storage for binary audit files, 74
- cusa audit class, 51

## D

- d option
  - auditreduce command, 92, 93
- debugging sequence number, 129
- defaults
  - audit service, 111
- deleting
  - archived audit files, 101
  - audit files, 97
  - not\_terminated audit files, 99
- determining
  - audit ID of a user, 61
  - whether auditing is running, 104
- disabling
  - audit policy, 48
  - audit service, 42

- disk space requirements
  - audit files, 37, 74
- displaying
  - audit policies, 49
  - audit policy defaults, 40
  - audit queue controls, 40, 51
  - audit record definitions, 89
  - audit records, 93
  - audit records in XML format, 94
  - auditing defaults, 40
  - definition of audit records, 89
  - exceptions to system-wide auditing, 40
  - selected audit records, 97

## E

- e option
  - auditreduce command, 92
- efficiency
  - auditing and, 38
- enabling
  - audit service, 42
- environment variables
  - audit token for, 124
  - presence in audit records, 34, 122
- /etc/security/audit\_event file
  - audit events and, 14
- /etc/syslog.conf file
  - auditing and, 87, 113
- event
  - description, 14
- event modifiers
  - audit records, 125
- exec\_args audit token
  - argv policy and, 124
  - format, 124
- exec\_env audit token
  - format, 124

## F

- failure and success events
  - audit class prefix, 115
- fe audit event modifier, 125
- file audit token

- format, 125
- file transfers
  - auditing, 64
- file vnode audit token, 123
- files, 14
  - See also* audit files
  - audit\_class, 113
  - audit\_event, 113
  - auditing modifications to, 59
  - public objects, 14
  - syslog.conf, 113
- flags line
  - process preselection mask, 119
- fmri audit token
  - format, 125
- format of audit records
  - auditrecord command, 90
- fp audit event modifier, 125
- ftp command
  - logging file transfers, 64

**G**

- group audit policy
  - and group token, 35, 125
  - description, 35
- group audit token
  - format, 125
  - group policy, and, 125

**H**

- h option
  - auditrecord command, 89
- hard disk
  - space requirements for auditing, 37
- header audit token
  - event modifiers, 125
  - format, 125
  - order in audit record, 125

**I**

- IDs
  - audit

- mechanism, 119
- overview, 11
- audit session, 119
- Internet-related audit tokens
  - ip address token, 126
  - ip port token, 126
  - socket token, 129
- ip address audit token
  - format, 126
- ip port audit token
  - format, 126
- ipc audit token, 126
- IPC type field values (ipc token), 126
- IPC\_perm audit token
  - format, 127

## L

- limiting
  - audit file size, 108
- local auditing, 13
- log files
  - audit records, 18, 96
  - configuring for audit service, 86
  - syslog audit records, 113
  - /var/adm/messages, 103
  - /var/log/syslog, 103
- logadm command
  - archiving text summary audit files, 101
- logging
  - ftp file transfers, 64
- logging in
  - auditing logins, 109
- lspolicy option
  - auditconfig command, 48

## M

- M option
  - auditreduce command, 99
- man pages
  - audit service, 112
- managing
  - audit files, 97, 100
  - audit records task map, 97

- audit trail overflow, 100
- auditing in zones, 26, 114
- mappings
  - events to classes (auditing), 16
- mask (auditing)
  - description of process preselection, 119
- merging
  - binary audit records, 97
- minus sign (-)
  - audit class prefix, 115
- modifying
  - user security attributes, 45
- monitoring
  - audit trail in real time, 38

## N

- na audit event modifier, 126
- naming conventions
  - audit files, 120
- never-audit* classes
  - process preselection mask, 119

## O

- O option
  - auditreduce command, 92, 99
- o option
  - auditreduce command, 97
- Oracle Audit Vault and Database Firewall
  - plugging in auditing, 24
- overflow prevention
  - audit trail, 100

## P

- p option
  - auditrecord command, 90
- path audit policy
  - description, 35
- path audit token
  - format, 127
- path\_attr audit token, 128
- permanent audit policy
  - configured audit policy, 48

- perzone audit policy
  - description, 35
  - setting, 50
  - using, 29, 68, 114
  - when to use, 26
- planning
  - auditing, 27
  - auditing in zones, 28
- plugins
  - auditing, 17
- plus sign (+) in audit class prefixes, 86, 115
- policies
  - for auditing, 34
  - that add tokens to audit records, 117
- post-selection in auditing, 13
- praudit command
  - converting audit records to readable format, 96
  - description, 113
  - piping auditreduce output to, 96
  - using in a script, 96
  - viewing audit records, 93
  - XML format, 94
- prefixes for audit classes, 115
- preselecting
  - audit classes, 44
- preselection in auditing, 14
- preselection mask (auditing)
  - description, 119
- preventing audit trail overflow, 100
- printing
  - audit log, 96
- privilege audit token, 128
- process audit characteristics
  - audit session ID, 119
  - audit user ID, 119
  - process preselection mask, 119
  - terminal ID, 119
- process audit token
  - format, 128
- process preselection mask
  - description, 119
- processing time costs of audit service, 36
- public audit policy
  - description, 35
  - read-only events, 35
- public directories

- auditing, 14
- public objects
  - auditing, 14

## Q

- qsize attribute
  - audit plugins, 51

## R

- rd audit event modifier, 126
- readable audit record format
  - converting audit records to, 96
- reducing
  - audit file size, 97
  - disk space required for audit files, 63
  - storage-space requirements for audit files, 38
- refreshing audit service, 69
- remote auditing, 14
- removing
  - audit events from audit\_event file, 62
  - user-specific auditing, 48
- replacing preselected audit classes, 44
- return audit token
  - format, 128
- rights
  - audit profiles, 114
- rights profiles
  - for audit service, 114

## S

- s option
  - audit command, 42, 69, 69
- scripts
  - audit\_warn script, 53, 113
  - monitoring audit files example, 38
  - processing praudit output, 96
- security
  - auditing and, 11, 21
- selecting
  - audit classes, 44
  - audit records, 91
  - events from audit trail, 91

- seq audit policy
  - and sequence token, 35, 129
  - description, 35
- sequence audit token
  - and seq audit policy, 129
  - format, 129
- session ID
  - audit, 119
- setflags option
  - auditconfig command, 44
- setnaflags option
  - auditconfig command, 44
- setplugin option
  - auditconfig command, 80, 82, 86
- setpolicy option
  - auditconfig command, 48
- setting
  - arge policy, 59
  - argv policy, 58
  - audit policy, 48
  - audit queue controls, 51
- sftp command
  - auditing file transfers, 64
- size of audit files
  - reducing, 97
  - reducing storage-space requirements, 38
- SMF
  - auditd service, 111
- socket audit token, 129
- sp audit event modifier, 126
- square brackets ([])
  - auditrecord output, 121
- starting auditing, 42
- storage costs and auditing, 37
- storage overflow prevention
  - audit trail, 100
- storing
  - audit files, 32, 74
  - audit files remotely, 33
- subject audit token
  - format, 129
- success and failure events
  - audit class prefix, 115
- svcadm command
  - restarting, 87

- syslog records, 20
- syslog.conf file
  - and auditing, 113
  - audit.notice level, 87
- system calls
  - argument audit token, 123
  - exec\_args audit token, 124
  - exec\_env audit token, 124
  - return audit token, 128
- System V IPC
  - ipc audit token, 126
  - IPC\_perm audit token, 127

## T

- t option
  - audit command, 42
- tail command
  - example of use, 38
- task maps
  - configuring audit logs, 73
  - configuring auditing, 42
  - managing audit records, 97
  - planning auditing, 27
- TCP addresses, 126
- temporary audit policy
  - active audit policy, 48
  - setting, 50
- terminal ID
  - audit, 119
- text audit token
  - format, 130
- time stamps
  - audit files, 120
- trail audit policy
  - and trailer token, 36
  - description, 36
- trailer audit token
  - format, 130
  - order in audit record, 130
  - praudit display, 130
- troubleshooting
  - active plugin, 105
  - audit classes
    - customized, 55, 106

- auditing, 103
- praudit command, 97
- too many audit records in queue, 81

## U

- UDP
  - addresses, 126
  - using for remote audit logs, 18
- use of authorization audit token, 130
- use of privilege audit token, 131
- user audit token, 131
- user ID
  - audit ID and, 119
- user ID and audit ID, 11
- User Security rights profile
  - modifying audit preselection for users, 45
- user\_attr database
  - listing user exceptions to audit preselection, 45
- user\_attr file
  - exceptions to system-wide audit classes, 16
- userattr command
  - displaying exceptions to system-wide auditing, 40
- usermod command
  - audit\_flags keyword, 45
  - exceptions to system-wide auditing, 16
  - specifying user exceptions to audit preselection, 45
  - using caret (^) prefix for audit\_flags exception, 46
- users
  - auditing all commands, 57
  - auditing individual users, 47
  - creating rights profile for a group, 48
  - modifying audit preselection mask of, 45
  - removing audit flags, 48

## V

- /var/adm/auditlog file
  - text audit records, 87
- /var/adm/messages file
  - troubleshooting auditing, 103
- /var/log/syslog file
  - troubleshooting auditing, 103
- variables

- adding to audit record, 34, 124
- auditing those associated with a command, 124
- viewing
  - audit record definitions, 89
  - binary audit files, 93
  - XML audit records, 94
- vnode audit token
  - format, 123

## W

- wr audit event modifier, 126

## X

- xclient audit token, 131
- XML format
  - audit records, 94

## Z

- ZFS File System Management rights profile
  - creating audit file systems, 74
- ZFS file systems
  - creating for binary audit files, 74
- ZFS Storage Management rights profile
  - creating pools for audit files, 74
- zonename audit policy
  - description, 36
  - using, 29, 114
- zonename audit token, 131
- zones
  - auditing and, 26, 114
  - configuring auditing in global zone, 50
  - perzone audit policy, 26, 29, 114
  - planning auditing in, 28
  - zonename audit policy, 29, 114

