Linux Audit Quick Start

SUSE Linux Enterprise 11 SP1

NOVELL® QUICK START CARD

Linux audit allows you to comprehensively log and track access to files, directories, and resources of your system, as well as trace system calls. It enables you to monitor your system for application misbehavior or code malfunctions. By creating a sophisticated set of rules including file watches and system call auditing, you can make sure that any violation of your security policies is noted and properly addressed.

To set up Linux audit on your system, proceed as follows:

1. Stop the default audit daemon with the `rcauditd stop` command.
2. Adjust the system configuration for audit and enable audit.
3. Configure the audit daemon.
4. Determine which system components to audit and set up audit rules.
5. Optionally configure plugin applications you intend to use with the audit dispatcher.
6. Start the audit daemon after you have completed the configuration of the audit system using the `rcauditd start` command.
7. Determine which reports to run and configure these reports.
8. Analyze the audit logs and reports.
9. (Optional) Analyze individual system calls with autrace.

Users Entitled to Work with Audit
The audit tools, configuration files, and logs are only available to `root`. This protects audit from ordinary users of the system. To manipulate any aspect of audit, you must be logged in as `root`.

Enabling Audit

Your first task for enabling audit is to activate system call auditing, since system call auditing capabilities are needed even when you are only configuring plain file or directory watches:

Enabling System Call Auditing for One Session Only
Enable with `auditctl -e 1` and disable with `auditctl -e 0`. These settings are valid for the current session only, and do not survive a reboot.

Enabling System Call Auditing Permanently
Permanently enable audit contexts for system calls by changing `AUDITD_DISABLE_CONTEXTS` in `/etc/sysconfig/auditd` from `yes` to `no`. To permanently disable audit contexts for system calls, revert this setting to `yes`. THis configuration will be applied with the next start of the audit daemon.

Configuring Audit

The configuration of the audit daemon is contained in the `/etc/audit/auditd.conf` configuration file. The default settings as shipped with SUSE Linux Enterprise should be sufficient for most setups.

```
log_file = /var/log/audit/audit.log
log_format = RAW
log_group = root
priority_boost = 4
flush = INCREMENTAL
freq = 20
num_logs = 4
disp_qos = lossy
dispatcher = /usr/sbin/audispd
name_format = NONE
#name = mydomain
max_log_file = 5
max_log_file_action = ROTATE
space_left = 75
space_left_action = SYSLOG
action_mail_acct = root
admin_space_left = 50
admin_space_left_action = SUSPEND
disk_full_action = SUSPEND
disk_error_action = SUSPEND
#tcp_listen_port =
tcp_listen_queue = 5
#tcp_client_ports = 1024-65535
tcp_client_max_idle = 0
```

Most of the settings in this file apply the audit log files and how the logging is done. The most important settings apply to the actions the daemon should take when encountering certain critical conditions or errors (system low on disk space, system out of disk space, or disk error) and when to warn the administrator about these conditions. These actions are customizable and range from a mere warning in syslog to a complete halt of the system. For more information about `/etc/audit/auditd.conf`, refer to *The Linux Audit Framework* manual and the manual page of `auditd.conf` (`auditd.conf(8)`).

Setting Up Audit Rules

Audit rules are used to specify which components of your system are audited. There are three basic types of audit rules:

- Basic audit system parameters
- File and directory watches
- System call audits

Before creating an audit rule set and before rolling it out to your system, carefully determine which components to audit. Extensive auditing causes a substantial logging load. Make sure that your system provides enough disk space to store large audit logs and test your audit rule set extensively before rolling it out to production.

Audit rules can either be passed to the audit system by the command line using `auditctl` or bundled into a rules file located under `/etc/audit/audit.rules` that is read during the start of the audit daemon:

```
# basic audit system parameters
-D
-b 8192
-f 1
-e 1

# some file and directory watches
-w /var/log/audit/
-w /etc/audit/auditd.conf -p rxwa
-w /etc/audit/audit.rules -p rxwa

-w /etc/passwd -p rwxa
-w /etc/sysconfig/

# an example system call rule
-a entry,always -S umask
```

The basic audit system parameters include a rule to delete any preexisting rules (`-D`) (to avoid clashes with the new rules), a rule that sets the number of outstanding audit buffers (`-b`), the failure flag (`-f`), and the enable flag (`-e`):

`-b`
: Depending on the audit load of your system, increase or decrease the number of outstanding audit buffers. If there are no more buffers left, the kernel checks the failure flag for action.

`-f`
: The failure flag controls the kernel's reaction to critical errors. Possible values are `0` (silent), `1` (printk, print a failure message), and `2` (panic, bring the system down—no clean shutdown and risk of data loss or corruption).

`-e`
: If set to `1`, this enables audit and audit contexts for system calls. Setting it to `2` does the same, but also locks down the configuration. Set to `0`, audit is disabled. This flag is used to enable or disable audit temporarily.

File system watches can be added whenever you want to track files or directories for unauthorized access. Typical examples would include watching the audit configuration and logs and user and security databases. Use permission filtering to focus on those system calls requesting the permissions in which you are interested:

```
-w /etc/audit/audit.rules -p rxwa
```

The `-p` flag enables permission filtering. This example has permission filtering turned on for read, write, execute, and attribute change permissions.

Note the following limitations to file system watches:

- Directory watches produce less verbose logs than exact file watches. When in need of detailed file-related records, enable separate file watches for all files of interest.
- Pathname globbing of any kind is not supported by audit. Always use the exact pathnames.
- Auditing can only be performed on existing files. Any files added while the audit daemon is already running are ignored until the audit rule set is updated to watch the new files.

Assigning keys to your audit rules helps you to identify any records related to this rule in the logs. An example rule plus key:

```
-w /var/log/audit/ -k LOG_audit
```

The `-k` option attaches a text string to any event that is recorded in the logs due to this rule. Using the `ausearch` log analyzer, you can easily filter for any events related to this particular rule.

A sample system call audit rule could look like the following:

```
-a entry,always -S umask
```

This adds the rule to the system call entry list (`-a`) and logs an event whenever this system call is used (`entry,always`). The `-S` option precedes the actual system call, `umask` in this example. Using `-F`, you could add optional filtering to this rule. For more information about audit rules, refer to *The Linux Audit Framework* and the manual page of `auditctl` (`auditctl(8)`).

Generating Reports

Every audit event is recorded in the audit log, `/var/log/audit/audit.log`. To avoid having to read the raw audit log, configure custom audit reports with aureport and run them regularly. Use the aureport tool to create various types of reports filtering for different fields of the audit records in the log. The output of any `aureport` command is printed in column format and can easily be piped to other commands for further processing. Because the aureport commands are scriptable, you can easily create custom report scripts to run at certain intervals to gather the audit information for you.

`aureport --summary`
Run this report to get a rough overview of the current audit statistics (events, logins, processes, etc.). To get detailed information about any of the event categories listed, run individual reports for the event type.

`aureport --success`
Run this report to get statistics of successful events on your system. This report includes the same event categories as the summary report. To get detailed information for a particular event type, run the individual report adding the `--success` option to filter for successful events of this type, for example, `aureport -f --success` to display all successful file-related events.

`aureport --failed`
Run this report to get statistics of failed events on your system. This report includes the same event categories as the summary report. To get detailed information for a particular event type, run the individual report adding the `--failed` option to filter for failed events of this type, such as `aureport -f --failed` to display all failed file-related events.

`aureport -l`
Run this command to generate a numbered list of all login-related events. The report includes date, time, audit ID, host and terminal used, as well as name of the executable, success or failure of the attempt, and an event ID.

`aureport -p`
Run this report to generate a numbered list of all process-related events. This command generates a numbered list of all process events including date, time, process ID, name of the executable, system call, audit ID, and event number.

`aureport -f`
Run this report to generate a numbered list of all file-related events. This command generates a numbered list of all process events including date, time, process ID, name of the executable, system call, audit ID and event number.

`aureport -u`
Run this report to find out which users are running what executables on your system. This command generates a numbered list of all user-related events including date, time, audit ID, terminal used, host, name of the executable, and an event ID.

Use the `-ts` and `-te` (for start time and end time) options with any of the above commands to limit your reports to a certain time frame. Use the `-i` option with any of these commands to transform numeric entities to human-readable text. The following command creates a file report for the time between 8 am and 5:30 pm on the current day and converts numeric entries to text.

```
aureport -ts 8:00 -te 17:30 -f -i
```

Analyzing Audit Log Files and Reports

While aureport helps you generate custom reports focusing on a certain area, ausearch helps you find the detailed log entry of individual events:

`ausearch -a` *`audit_event_id`*
: Run this search to view all records carrying a particular audit event ID. Each audit event message is logged, along with a message ID consisting of a UNIX epoch time stamp plus a unique event ID, separated by a colon. All events that are logged from one application's system call have the same event ID. For example, use `ausearch -a 1234` to display all audit events carrying this audit event ID. As one application's system call may trigger several events to be logged, you are likely to retrieve more than one record from the log.

`ausearch -ul` *`login_id`*
: Run this search to view records associated with a particular login user ID. It displays any records related to the user login ID specified, provided that user had been able to log in successfully. For example, use `ausearch -ul root` to list all processes owned by the given login user ID.

`ausearch -k` *`key`*
: Run this search to find records that contain a certain key assigned in the audit rule set. For example, use `ausearch -k CFG_etc` to display any records containing the `CFG_etc` key.

`ausearch -m` *`message_type`*
: Run this search to find records related to a particular message type. Examples of valid message types include `PATH`, `SYSCALL`, `USER_LOGIN`. Invoking `ausearch -m` without a message type displays a list of all message types.

`ausearch -f` *`filename`*
: Run this search to find records containing a certain filename. For example, run `ausearch -f /foo/bar` for all records related to the `/foo/bar` file. Using the filename alone would work as well, but using relative paths would not.

`ausearch -p` *`process_id`*
: Run this to search for records related to a certain process ID. For example, use `ausearch -p 13368` to search for all records related to this process ID.

Use the `-ts` and `-te` (start time and end time) options with any of these commands to limit your reports to a certain time frame. Use the `-i` option with any of these to transform numeric entities to human readable text. The following command searches for any file event related to `audit.log` that took place any time between 8 am and 5:30 pm on the current day and converts numeric entries to text.

```
ausearch -ts 8:00 -te 17:30 -f audit.log -i
```

Analyzing Individual System Calls

Perform dedicated audits of individual processes using the autrace command. autrace works similarly to the strace command, but gathers slightly different information. The output of autrace is written to `/var/log/audit/audit.log` and does not look any different from the standard audit log entries.

When performing an autrace on a process, make sure that any audit rules are purged from the queue to avoid having these rules clash with the ones that autrace adds. Delete the audit rules with the `auditctl -D` command.

```
autrace /usr/bin/less /etc/sysconfig/auditd

Waiting to execute: /usr/bin/less
Cleaning up...
No rules
Trace complete. You can locate the records
with 'ausearch -i -p 7642'
```

Always use the full path to the executable with autrace. After the trace is complete, autrace provides you with the event ID of the trace, so you can analyze the entire data trail with ausearch. To restore the audit system to use the audit rule set again, just restart the audit daemon by calling `rcauditd restart`.

Audit Tool Set

`auditctl`
: Controls the audit system. Check the audit daemon's status and rule set, delete rules, or create new ones.

`aureport`
: Create various types of reports from the audit daemon logs.

`ausearch`
: Create custom queries to search the audit daemon logs.

`autrace`
: Add audit rules to trace a process. Similar to `strace`.

`rcauditd`
: Controls the audit init script.

Files

`/etc/audit/auditd.conf`
: Contains configuration options specific to the audit daemon, such as log file location, log rotation, maximum size of the log file, and various actions to be taken when the system starts to run low on disk space.

`/etc/sysconfig/auditd`
: Controls configuration aspects of auditd that are not covered in `/etc/audit/auditd.conf`, such as the locale to use with audit, the use of audit contexts with system calls, and if rules and watches should be deleted on shutdown of the system.

`/etc/audit/audit.rules`
Controls the rules auditd processes to track system calls and file and directory access.

`/var/log/audit/audit.log`
The audit log file.

For More Information

For a more detailed introduction to the Linux audit framework, refer to the *The Linux Audit Framework* manual that is available at `http://www.novell.com/documentation/sles11/http://www.novell.com/documentation/sled11/`.

Legal Notice

All content is copyright © 2006–2010 Novell, Inc. All rights reserved.

This manual is protected under Novell intellectual property rights. By reproducing, duplicating or distributing this manual you explicitly agree to conform to the terms and conditions of this license agreement.

This manual may be freely reproduced, duplicated and distributed either as such or as part of a bundled package in electronic and/or printed format, provided however that the following conditions are fulfilled:

That this copyright notice and the names of authors and contributors appear clearly and distinctively on all reproduced, duplicated and distributed copies. That this manual, specifically for the printed format, is reproduced and/or distributed for noncommercial use only. The express authorization of Novell, Inc must be obtained prior to any other use of any manual or part thereof.

For Novell trademarks, see the Novell Trademark and Service Mark list `http://www.novell.com/company/legal/trademarks/tmlist.html`. Linux* is a registered trademark of Linus Torvalds. All other third party trademarks are the property of their respective owners. A trademark symbol (®, ™ etc.) denotes a Novell trademark; an asterisk (*) denotes a third party trademark.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither Novell, Inc., SUSE LINUX Products GmbH, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

Created by SUSE® with XSL-FO