

NAME

auditd.conf – audit daemon configuration file

DESCRIPTION

The file */etc/audit/auditd.conf* contains configuration information specific to the audit daemon. Each line should contain one configuration keyword, an equal sign, and then followed by appropriate configuration information. All option names and values are case insensitive. The keywords recognized are listed and described below. Each line should be limited to 160 characters or the line will be skipped. You may add comments to the file by starting the line with a '#' character.

local_events

This yes/no keyword specifies whether or not to include local events. Normally you want local events so the default value is yes. Cases where you would set this to no is when you want to aggregate events only from the network. At the moment, this is useful if the audit daemon is running in a container. This option can only be set once at daemon start up. Reloading the config file has no effect.

log_file This keyword specifies the full path name to the log file where audit records will be stored. It must be a regular file. The default path is */var/log/audit/audit.log* if not explicitly set.

write_logs

This yes/no keyword determines whether or not to write logs to the disk. Normally you want this so the default is yes.

log_format

The log format describes how the information should be stored on disk. There are 2 options: raw and enriched. If set to *RAW*, the audit records will be stored in a format exactly as the kernel sends it. The *ENRICHED* option will resolve all uid, gid, syscall, architecture, and socket address information before writing the event to disk. This aids in making sense of events created on one system but reported/analyzed on another system. The *NOLOG* option is now deprecated. If you were setting this format, now you should set the *write_logs* option to no.

log_group

This keyword specifies the group that is applied to the log file's permissions. The default is root. The group name can be either numeric or spelled out.

priority_boost

This is a non-negative number that tells the audit daemon how much of a priority boost it should take. The default is 4. No change is 0.

flush

Valid values are *none*, *incremental*, *incremental_async*, *data*, and *sync*. If set to *none*, no special effort is made to flush the audit records to disk. If set to *incremental*, Then the *freq* parameter is used to determine how often an explicit flush to disk is issued. The *incremental_async* parameter is very much like *incremental* except the flushing is done asynchronously for higher performance. The *data* parameter tells the audit daemon to keep the data portion of the disk file sync'd at all times. The *sync* option tells the audit daemon to keep both the data and meta-data fully sync'd with every write to disk. The default value is *incremental_async*.

freq

This is a non-negative number that tells the audit daemon how many records to write before issuing an explicit flush to disk command. This value is only valid when the *flush* keyword is set to *incremental* or *incremental_async*.

num_logs

This keyword specifies the number of log files to keep if rotate is given as the *max_log_file_action*. If the number is < 2, logs are not rotated. This number must be 999 or less. The default is 0 - which means no rotation. As you increase the number of log files being rotated, you may need to adjust the kernel backlog setting upwards since it takes more time to rotate the files. This is typically done in */etc/audit/audit.rules*. If log rotation is configured to occur, the daemon will check for excess logs and remove them in effort to keep disk space available. The excess log check is only

done on startup and when a reconfigure results in a space check.

name_format

This option controls how computer node names are inserted into the audit event stream. It has the following choices: *none*, *hostname*, *fqd*, *numeric*, and *user*. *None* means that no computer name is inserted into the audit event. *hostname* is the name returned by the `gethostname` syscall. The *fqd* means that it takes the *hostname* and resolves it with `dns` for a fully qualified domain name of that machine. *Numeric* is similar to *fqd* except it resolves the IP address of the machine. In order to use this option, you might want to test that `'hostname -i'` or `'domainname -i'` returns a numeric address. Also, this option is not recommended if `dhcp` is used because you could have different addresses over time for the same machine. *User* is an admin defined string from the *name_option*. The default value is *none*.

name This is the admin defined string that identifies the machine if *user* is given as the *name_format* option.

max_log_file

This keyword specifies the maximum file size in mebibytes (MiB). When this limit is reached, it will trigger a configurable action. The value given must be numeric.

max_log_file_action

This parameter tells the system what action to take when the system has detected that the max file size limit has been reached. Valid values are *ignore*, *syslog*, *exec*, *suspend*, *rotate* and *keep_logs*. If set to *ignore*, the audit daemon does nothing. *syslog* means that it will issue a warning to `syslog`. *exec /path-to-script* will execute the script. You cannot pass parameters to the script. The script is also responsible for telling the auditd daemon to resume logging once its completed its action. This can be done by adding `auditdctl --signal resume` to the script. Also note that logging is stopped when this script runs. Whatever it does needs to be real quick because events are backing up in the kernel. The script **MUST** delete or rename `/var/log/audit/audit.log` or when logging resumes, it will retrigger executing the script. If the script fails to execute, auditing is suspended until it is told to resume by sending it a signal. *suspend* will cause the audit daemon to stop writing records to the disk. The daemon will still be alive. The *rotate* option will cause the audit daemon to rotate the logs. It should be noted that logs with higher numbers are older than logs with lower numbers. This is the same convention used by the `logrotate` utility. The *keep_logs* option is similar to *rotate* except it does not use the *num_logs* setting. This prevents audit logs from being overwritten. The effect is that logs accumulate and are not deleted – which will trigger the *space_left_action* if the volume fills up. This is best used in combination with an external script used to archive logs on a periodic basis.

verify_email

This option determines if the email address given in *action_mail_acct* is checked to see if the domain name can be resolved. This option must be given before *action_mail_acct* or the default value of *yes* will be used.

action_mail_acct

This option should contain a valid email address or alias. Addresses may use RFC 5233 style sub-addressing with the '+' character. The allowed characters are alphanumerics, '@', '.', '-', '_', and '+'. The default address is `root`. If the email address is not local to the machine, you must make sure you have email properly configured on your machine and network. This option requires that `/usr/lib/sendmail` exists on the machine. Many clients provide it as a symlink.

space_left

If the free space in the filesystem containing *log_file* drops below this value, the audit daemon takes the action specified by *space_left_action*. If the value of *space_left* is specified as a whole number, it is interpreted as an absolute size in mebibytes (MiB). If the value is specified as a number between 1 and 99 followed by a percentage sign (e.g., 5%), the audit daemon calculates the absolute size in mebibytes based on the size of the filesystem containing *log_file*. (E.g., if the filesystem containing *log_file* is 2 gigabytes in size, and *space_left* is set to 25%, then the audit daemon sets *space_left* to approximately 500 mebibytes. Note that this calculation is performed

when the audit daemon starts, so if you resize the filesystem containing *log_file* while the audit daemon is running, you should send the audit daemon SIGHUP to re-read the configuration file and recalculate the correct percentage.

space_left_action

This parameter tells the system what action to take when the system has detected that it is starting to get low on disk space. Valid values are *ignore*, *syslog*, *rotate*, *email*, *exec*, *suspend*, and *single*. If set to *ignore*, the audit daemon does nothing. *syslog* means that it will issue a warning to syslog. *rotate* will rotate logs, losing the oldest to free up space. *Email* means that it will send a warning to the email account specified in *action_mail_acct* as well as sending the message to syslog. *exec /path-to-script* will execute the script. You cannot pass parameters to the script. The script is also responsible for telling the auditd daemon to resume logging once its completed its action. This can be done by adding service auditd resume to the script. *suspend* will cause the audit daemon to stop writing records to the disk. If the script fails to execute, the audit daemon will be left suspended until its told to resume by sending it a signal. The *single* option will cause the audit daemon to put the computer system in single user mode. Except for rotate, it will perform this action just one time. The previously available *halt* option, which would cause the audit daemon to shut down the computer system, has been deprecated and should no longer be used. It was determined that halting the system at this stage could lead to unintended consequences and is considered a bad action if selected.

Disk space notifications follow a three-stage progression. The *space_left_action* is the low water mark and serves as the first warning that disk space is running low. Halting at this stage is not recommended, as it prevents administrators from taking corrective action. The next stage, *admin_space_left_action*, indicates an emergency level where immediate action is required to free up disk space. Administrators should configure critical responses for this level. Finally, the *disk_full_action* occurs when the disk is completely full. At this stage, the system may have already halted, and preemptive measures configured in earlier stages will determine the system's behavior.

admin_space_left

This is a numeric value in mebibytes (MiB) that tells the audit daemon when to perform a configurable action because the system **is running low** on disk space. This should be considered the last chance to do something before running out of disk space. The numeric value for this parameter should be lower than the number for *space_left*. You may also append a percent sign (e.g. 1%) to the number to have the audit daemon calculate the number based on the disk partition size.

admin_space_left_action

This parameter tells the system what action to take when the system has detected that it **is low on disk space**. Valid values are *ignore*, *syslog*, *rotate*, *email*, *exec*, *suspend*, *single*, and *halt*. If set to *ignore*, the audit daemon does nothing. *syslog* means that it will issue a warning to syslog. *rotate* will rotate logs, losing the oldest to free up space. *Email* means that it will send a warning to the email account specified in *action_mail_acct* as well as sending the message to syslog. *exec /path-to-script* will execute the script. You cannot pass parameters to the script. The script is also responsible for telling the auditd daemon to resume logging once its completed its action. This can be done by adding service auditd resume to the script. *Suspend* will cause the audit daemon to stop writing records to the disk. The daemon will still be alive. The *single* option will cause the audit daemon to put the computer system in single user mode. The *halt* option will cause the audit daemon to shutdown the computer system. Except for rotate, it will perform this action just one time.

disk_full_action

This parameter tells the system what action to take when the system has detected that the partition to which log files are written has become full. Valid values are *ignore*, *syslog*, *rotate*, *exec*,

suspend, *single*, and *halt*. If set to *ignore*, the audit daemon will issue a syslog message but no other action is taken. *Syslog* means that it will issue a warning to syslog. *rotate* will rotate logs, losing the oldest to free up space. *exec /path-to-script* will execute the script. You cannot pass parameters to the script. The script is also responsible for telling the auditd daemon to resume logging once it's completed its action. This can be done by adding `service auditd resume` to the script. *Suspend* will cause the audit daemon to stop writing records to the disk. The daemon will still be alive. The *single* option will cause the audit daemon to put the computer system in single user mode. *halt* option will cause the audit daemon to shutdown the computer system.

disk_error_action

This parameter tells the system what action to take whenever there is an error detected when writing audit events to disk or rotating logs. Valid values are *ignore*, *syslog*, *exec*, *suspend*, *single*, and *halt*. If set to *ignore*, the audit daemon will not take any action. *Syslog* means that it will issue no more than 5 consecutive warnings to syslog. *exec /path-to-script* will execute the script. You cannot pass parameters to the script. *Suspend* will cause the audit daemon to stop writing records to the disk. The daemon will still be alive. The *single* option will cause the audit daemon to put the computer system in single user mode. *halt* option will cause the audit daemon to shutdown the computer system.

tcp_listen_port

This is a numeric value in the range 1..65535 which, if specified, causes auditd to listen on the corresponding TCP port for audit records from remote systems. The audit daemon may be linked with `tcp_wrappers`. You may want to control access with an entry in the `hosts.allow` and `deny` files. If this is deployed on a systemd based OS, then you may need to adjust the 'After' directive. See the note in the `auditd.service` file. Networking can be enabled by adding this option and sending auditd a SIGHUP. Changing the port or disabling networking requires restarting the daemon so that clients reconnect.

tcp_listen_queue

This is a numeric value which indicates how many pending (requested but unaccepted) connections are allowed. The default is 5. Setting this too small may cause connections to be rejected if too many hosts start up at exactly the same time, such as after a power failure. This setting is only used for aggregating servers. Clients logging to a remote server should keep this commented out. Changing this value while the daemon is running restarts the listener and drops any current connections.

tcp_max_per_addr

This is a numeric value which indicates how many concurrent connections from one IP address is allowed. The default is 1 and the maximum is 1024. Setting this too large may allow for a Denial of Service attack on the logging server. Also note that the kernel has an internal maximum that will eventually prevent this even if auditd allows it by config. The default should be adequate in most cases unless a custom written recovery script runs to forward unsent events. In this case you would increase the number only large enough to let it in too.

use_libwrap

This setting determines whether or not to use `tcp_wrappers` to discern connection attempts that are from allowed machines. Legal values are either *yes*, or *no*. The default value is *yes*.

tcp_client_ports

This parameter may be a single numeric value or two values separated by a dash (no spaces allowed). It indicates which client ports are allowed for incoming connections. If not specified, any port is allowed. Allowed values are 1..65535. For example, to require the client use a privileged port, specify `1-1023` for this parameter. You will also need to set the `local_port` option in the `auditd-remote.conf` file. Making sure that clients send from a privileged port is a security feature to prevent log injection attacks by untrusted users.

tcp_client_max_idle

This parameter indicates the number of seconds that a client may be idle (i.e. no data from them at all) before auditd complains. This is used to close inactive connections if the client machine has a

problem where it cannot shutdown the connection cleanly. Note that this is a global setting, and must be higher than any individual client heartbeat_timeout setting, preferably by a factor of two. The default is zero, which disables this check.

transport

If set to *TCP*, only clear text tcp connections will be used. If set to *KRB5*, then Kerberos 5 will be used for authentication and encryption. The default value is *TCP*. Changes to this option take effect only after restart so that clients can reconnect.

enable_krb5

This option is deprecated. Use the *transport* option above instead. If set to "yes", Kerberos 5 will be used for authentication and encryption. The default is "no". If this option is set to "yes" and it follows the transport option, it will override the transport setting. This would be the normal expected behavior for backwards compatibility.

krb5_principal

This is the principal for this server. The default is "auditd". Given this default, the server will look for a key named like *auditd/hostname@EXAMPLE.COM* stored in */etc/audit/audit.key* to authenticate itself, where hostname is the canonical name for the server's host, as returned by a DNS lookup of its IP address.

krb5_key_file

Location of the key for this client's principal. Note that the key file must be owned by root and mode 0400. The default is */etc/audit/audit.key*

distribute_network

If set to "yes", network originating events will be distributed to the audit dispatcher for processing. The default is "no".

q_depth

This is a numeric value that tells how big to make the internal queue of the audit event dispatcher. A bigger queue lets it handle a flood of events better, but could hold events that are not processed when the daemon is terminated. If you get messages in syslog about events getting dropped, increase this value. The default value is 2000.

overflow_action

This option determines how the daemon should react to overflowing its internal queue. When this happens, it means that more events are being received than it can pass along to child processes. This error means that it is going to lose the current event that it's trying to dispatch. This option has the following choices: *ignore*, *syslog*, *suspend*, *single*, and *halt*. If set to *ignore*, the audit daemon does nothing. *syslog* means that it will issue a warning to syslog. *suspend* will cause the audit daemon to stop sending events to child processes. The daemon will still be alive. The *single* option will cause the audit daemon to put the computer system in single user mode. *halt* option will cause the audit daemon to shutdown the computer system.

max_restarts

This is a non-negative number that tells the audit event dispatcher how many times it can try to restart a crashed plugin. The default is 10.

plugin_dir

This is the location that auditd will use to search for its plugin configuration files. The default directory is */etc/audit/plugins.d*

end_of_event_timeout

This is a non-negative number of seconds used by the userspace *auparse()* library routines and the *aureport(8)*, *ausearch(8)* utilities to consider an event is complete when parsing an event log stream. For an event stream being processed, if the time of the current event is over *end_of_event_timeout* seconds old, compared to co-located events, then the event is considered complete. The default is 2 seconds. See the NOTES section for more detail.

report_interval

This option specifies the interval between automatic state reports. The value is a time string composed of a number optionally followed by **m** for minutes, **h** for hours, **d** for days, or **M** for months. The default is 0 which disables periodic reporting. The largest value is 40 days. When set, auditd will periodically generate the state report written to `/run/audit/auditd.state`.

RELOADING

Most parameters can be changed while the daemon is running by sending **SIGHUP** to **auditd**. This can also be done with **auditctl --signal reload**. Options that are only read at startup include *local_events* and *verify_email*

NOTES

In a CAPP environment, the audit trail is considered so important that access to system resources must be denied if an audit trail cannot be created. In this environment, it would be suggested that `/var/log/audit` be on its own partition. This is to ensure that space detection is accurate and that no other process comes along and consumes part of it.

The flush parameter should be set to sync or data.

Max_log_file and num_logs need to be adjusted so that you get complete use of your partition. It should be noted that the more files that have to be rotated, the longer it takes to get back to receiving audit events. Max_log_file_action should be set to keep_logs.

Space_left should be set to a number that gives the admin enough time to react to any alert message and perform some maintenance to free up disk space. This would typically involve running the **aureport -t** report and moving the oldest logs to an archive area. The value of space_left is site dependent since the rate at which events are generated varies with each deployment. The space_left_action is recommended to be set to email. If you need something like an snmp trap, you can use the exec option to send one.

Admin_space_left should be set to the amount of disk space on the audit partition needed for admin actions to be recorded. Admin_space_left_action would be set to single so that use of the machine is restricted to just the console.

The disk_full_action is triggered when no more room exists on the partition. All access should be terminated since no more audit capability exists. This can be set to either single or halt.

The disk_error_action should be set to syslog, single, or halt depending on your local policies regarding handling of hardware malfunctions.

Specifying a single allowed client port may make it difficult for the client to restart their audit subsystem, as it will be unable to recreate a connection with the same host addresses and ports until the connection closure TIME_WAIT state times out.

Auditd events are made up of one or more records. The auditd system cannot guarantee that the set of records that make up an event will occur atomically, that is the stream will have interleaved records of different events, IE

```
event0_record0
event1_record0
event2_record0
event1_record3
event2_record1
event1_record4
event3_record0
```

The auditd system does not guarantee that the records that make up an event will appear in order. Thus, when processing event streams, we need to maintain a list of events with their own list of records hence List of List (LOL) event processing.

When processing an event stream we define the end of an event via

record type = AUDIT_EOE (audit end of event type record), or
record type = AUDIT_PROCTITLE (we note the AUDIT_PROCTITLE is always the last record),
or
record type = AUDIT_KERNEL (kernel events are one record events), or
record type < AUDIT_FIRST_EVENT (only single record events appear before this type), or
record type >= AUDIT_FIRST_ANOM_MSG (only single record events appear after this type), or
record type >= AUDIT_MAC_UNLBL_ALLOW && record type <= AUDIT_MAC_CALIPSO_DEL (these are also one record events), or
for the stream being processed, the time of the event is over end_of_event_timeout seconds old.

LOG ROTATION POLICY

By default, auditd uses size-based log rotation. If you prefer time-based rotation (e.g., hourly, daily, weekly, or custom schedule), refer to auditd.cron(5) for configuration details.

FILES

/etc/audit/auditd.conf
Audit daemon configuration file

SEE ALSO

auditd(8), **audisp-remote.conf(5)**, **auditd-plugins(5)**, **auditd.cron(5)**.

AUTHOR

Steve Grubb