

Novell Storage Services™ Auditing Client Logger (VLOG) Utility Reference

Novell® Open Enterprise Server

2 SP2

April 29, 2010

www.novell.com



Legal Notices

Novell, Inc., makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc., reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc., makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc., reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. See the [Novell International Trade Services Web page \(http://www.novell.com/info/exports/\)](http://www.novell.com/info/exports/) for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Copyright © 2010 Novell, Inc. All rights reserved. No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of the publisher.

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the latest online documentation for this and other Novell products, see the [Novell Documentation Web page \(http://www.novell.com/documentation\)](http://www.novell.com/documentation).

Novell Trademarks

For Novell trademarks, see [the Novell Trademark and Service Mark list \(http://www.novell.com/company/legal/trademarks/tmlist.html\)](http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.

Contents

About This Guide	7
1 Overview of the NSS Auditing Client Logger (VLOG) Utility	9
1.1 Using VLOG with the NSS Auditing Engine	9
1.1.1 Logged Output	9
1.1.2 Paths to Include or Exclude	9
1.1.3 File System Events to Monitor	9
1.1.4 NSS, NCP, and CIFS Event Sub-Types to Monitor	10
1.1.5 VIGIL Events to Monitor	10
1.2 Using Auditing Client Applications with the NSS Auditing Engine	10
1.2.1 Novell Sentinel Log Manager	11
1.2.2 Third-Party Partner Applications	11
2 VLOG Utility Man Page	13
vlog	14

About This Guide

This reference guide describes the syntax and options for the Novell Storage Services (NSS) Auditing Client Logger (VLOG) utility for Novell Open Enterprise Server (OES) 2 Support Pack 2 (SP2) Linux. The VLOG utility is used with the NSS Auditing Engine, which is available in OES 2 SP2 Linux and later.

This guide includes the following sections:

- ♦ [Chapter 1, “Overview of the NSS Auditing Client Logger \(VLOG\) Utility,” on page 9](#)
- ♦ [Chapter 2, “VLOG Utility Man Page,” on page 13](#)

Audience

This guide is intended for system administrators or anyone who is responsible for auditing file system events on NSS file systems on OES 2 SP2 Linux servers.

Knowledge of the NSS file system is assumed. Some background knowledge of the host operating system is also assumed.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to www.novell.com/documentation/feedback.html and enter your comments there.

Documentation Updates

The VLOG man page, `vlog(8)`, is available on the server. Updates to the man page are delivered with any updates to the VLOG utility.

For the most recent version of the *OES 2 SP2: NSS Auditing Client Logger (VLOG) Reference*, visit the [Novell Open Enterprise Server 2 Documentation Web site \(http://www.novell.com/documentation/oes2/security.html\)](http://www.novell.com/documentation/oes2/security.html) under *Security*.

Additional Documentation

Information about the NSS Auditing Engine SDK (Software Development Kit) is available on the [NSS Auditing SDK Web site \(http://developer.novell.com/wiki/index.php/NSS_Auditing_SDK\)](http://developer.novell.com/wiki/index.php/NSS_Auditing_SDK).

Overview of the NSS Auditing Client Logger (VLOG) Utility

1

The Novell Storage Services (NSS) Auditing Client Logger (VLOG) utility for Novell Open Enterprise Server (OES) 2 Support Pack 2 (SP2) Linux is used with the NSS Auditing Engine (`/etc/init.d/novell-vigil`). The NSS Auditing Engine is installed by default when you install NSS on an OES 2 SP2 Linux or later server.

- ♦ [Section 1.1, “Using VLOG with the NSS Auditing Engine,” on page 9](#)
- ♦ [Section 1.2, “Using Auditing Client Applications with the NSS Auditing Engine,” on page 10](#)

1.1 Using VLOG with the NSS Auditing Engine

When VLOG is running, it intercepts, parses, filters, augments, and displays auditing records received from the NSS Auditing Engine (`vigil`). For information about configuring and using the VLOG utility, see [Chapter 2, “VLOG Utility Man Page,” on page 13](#).

The basic functionality includes:

- ♦ [Section 1.1.1, “Logged Output,” on page 9](#)
- ♦ [Section 1.1.2, “Paths to Include or Exclude,” on page 9](#)
- ♦ [Section 1.1.3, “File System Events to Monitor,” on page 9](#)
- ♦ [Section 1.1.4, “NSS, NCP, and CIFS Event Sub-Types to Monitor,” on page 10](#)
- ♦ [Section 1.1.5, “VIGIL Events to Monitor,” on page 10](#)

1.1.1 Logged Output

By default, `vlog` sends its output to `stdout` in an XML record format. VLOG also supports output in CSV (comma-separated values) format and SENT format (for Novell Sentinel/Log Manager products). For information, see [“VLOG Options” on page 15](#).

1.1.2 Paths to Include or Exclude

VLOG allows you to specify which files and directories are to be monitored. You can specify patterns for the file and directory names by using a defined set of search characters. You can specify which file paths are to be included or excluded. For information, see [“Path Element Options” on page 21](#). For examples of path patterns, see [“Path Element Examples” on page 22](#).

1.1.3 File System Events to Monitor

VLOG can be configured to log various file system events on files and directories that are reported by the NSS Auditing Engine, including:

- ♦ delete
- ♦ create

- ♦ open
- ♦ close
- ♦ rename
- ♦ link
- ♦ metadata modified
- ♦ trustee added or removed
- ♦ inherited rights modified

For information, see [“Event Types” on page 26](#) and [“Event Type Examples” on page 26](#).

1.1.4 NSS, NCP, and CIFS Event Sub-Types to Monitor

These NSS file system events can be audited by NSS, NCP (NetWare Core Protocol), and CIFS sub-types. For information, see [“Event Sub-Types NSS, NCP, and CIFS” on page 27](#) and [“Event Sub-Type Examples” on page 28](#).

1.1.5 VIGIL Events to Monitor

VLOG can also be configured to report various events internal to the NSS Auditing Engine, referred to as VIGIL events, such as:

- ♦ Starting or stopping the `vigil.ko` kernel module
- ♦ Starting or stopping the `vigil.ncp.ko` kernel module
- ♦ Starting or stopping the `vigil.nss.ko` kernel module
- ♦ Starting or stopping the `vigil.cifs.ko` kernel module
- ♦ Starting or stopping the Auditing Client (an internal construct of the NSS Auditing Engine)
- ♦ Starting or stopping the Auditing Client User (an internal construct of the NSS Auditing Engine)
- ♦ Rolling the audit record log file over to a new file when the log reaches an administrator-specified maximum size

For information, see [“Patterns for Filtering Records of Type VIGIL” on page 17](#) and [“Examples for Filtering VIGIL Events” on page 19](#).

1.2 Using Auditing Client Applications with the NSS Auditing Engine

Some auditing client applications, such as Novell Sentinel Log Manager and various third-party products, can access audited events that are reported by the NSS Auditing Engine. Information about the NSS Auditing Engine Software Developer Kit (SDK) is available on the [NSS Auditing SDK Web site \(http://developer.novell.com/wiki/index.php/NSS_Auditing_SDK\)](http://developer.novell.com/wiki/index.php/NSS_Auditing_SDK).

- ♦ [Section 1.2.1, “Novell Sentinel Log Manager,” on page 11](#)
- ♦ [Section 1.2.2, “Third-Party Partner Applications,” on page 11](#)

1.2.1 Novell Sentinel Log Manager

Novell Sentinel Log Manager can be used to collect and report on event logs from the NSS Auditing Client Logger utility. Novell Sentinel Log Manager runs on a 64-bit SUSE Linux Enterprise Server (SLES) 11 host. You can download Novell Sentinel Log Manager from the [Novell Download Web site \(http://download.novell.com/Download?buildid=woGGwp3Mab4~\)](http://download.novell.com/Download?buildid=woGGwp3Mab4~). A 90-day evaluation license is available on the download site. For installation and usage instructions, see the [Novell Sentinel Log Manager Documentation Web site \(http://www.novell.com/documentation/novelllogmanager10/\)](http://www.novell.com/documentation/novelllogmanager10/).

1.2.2 Third-Party Partner Applications

As of the OES 2 SP2 Linux release, the following Novell partners are developing applications for use with the NSS Auditing Engine:

- ♦ Blue Lance
- ♦ NetVision
- ♦ Symantec

VLOG Utility Man Page

2

This section provides the syntax, options, and examples for the Novell Storage Services (NSS) Auditing Client Logger (VLOG) utility for Novell Open Enterprise Server (OES) 2 Support Pack 2 (SP2) Linux. This information is also available on the server as the `vlog(8)` man page.

- ♦ [“Synopsis” on page 14](#)
- ♦ [“Availability” on page 14](#)
- ♦ [“Syntax” on page 14](#)
- ♦ [“Description” on page 15](#)
- ♦ [“VLOG Options” on page 15](#)
- ♦ [“Filtering Records” on page 17](#)
- ♦ [“Patterns for Filtering Records of Type VIGIL” on page 17](#)
- ♦ [“Patterns for Filtering Records of Type NSS, NCP, and CIFS” on page 19](#)
- ♦ [“Filter Pattern Examples” on page 28](#)
- ♦ [“Troubleshooting” on page 29](#)

vlog(8)

Name

vlog - The Novell Storage Services Auditing Client Logger utility.

Synopsis

```
/opt/novell/vigil/bin/vlog [OPTIONS]
```

or information about options, see [“VLOG Options” on page 15](#).

Availability

Novell Open Enterprise Server 2 Support Pack 3 Linux or later. It is also available as a patch (released in April 2010) to Novell Open Enterprise Server 2 Support Pack 2 Linux.

Syntax

Prior to running vlog, the NSS Auditing Engine (`/etc/init.d/novell-vigil`) should be started. To check the status of the engine, issue the following command (as the `root` user) in a terminal console:

```
/etc/init.d/novell-vigil status
```

If the status is not “Running”, the engine should be started by issuing the following command (as the `root` user) in a terminal console:

```
/etc/init.d/novell-vigil start
```

Run the NSS Auditing Client Logger (`vlog`) utility in a terminal console (generally as the `root` user).

```
/opt/novell/vigil/bin/vlog [OPTIONS]
```

Stopping `vlog` requires a `SIGTERM` signal. This can be done by issuing a `Ctrl+C` in the terminal where `vlog` is running, or by using the `kill` or `killall` command. For example, to kill all instances of `vlog`, enter the following in a terminal console:

```
killall -s SIGTERM vlog
```

IMPORTANT: If `vlog` terminates because of a `SIGTERM` signal, it instructs the NSS Auditing Engine (`vigil`) to discontinue sending auditing data records to the specific instance of `vlog`.

If the application is terminated without a `SIGTERM` signal (such as closing the window in which `vlog` is running), the NSS Auditing Engine does not discontinue sending auditing records to the `vlog` instance. Because an auditing record log entry is sent by appending it to the log file instance, the log file continues to grow. Over time, the audit log files can grow to fill all available disk space. See [“Troubleshooting” on page 29](#) for further details.

Description

Intercepts, parses, filter, augments, and displays auditing records received from the NSS Auditing Engine (`vigil`).

VLOG Options

By default, `vlog` sends its output to `stdout` in an XML record format. The following options modify `vlog`'s default behavior:

[-a, --asciiOut]

Print Unicode 16 (NSS paths) in '\uXXXX' format.

[-c, --clientKey] CLIENT-KEY

Debugging option. (Do not use.)

[-C, --clientName] CLIENT-NAME

Debugging option. (Do not use.)

The specified client name can be from 1 to 15 characters. Longer names might be generated by `vlog` if the `[-C, --clientName]` option is not used. The client name must be unique on the system. The specified client name is added to `CLIENT_`.

For example, if you provide a 15-character client name of `jasonjames12345`, `vlog` creates a client name of `CLIENT_jasonjames12345`.

[-f, --format] [NUL, XML, CSFV, SENT]

Sets the output format using the specified format as follows:

NUL

No output.

XML

(Default) Extensible Markup Language (XML) format.

CSV

Comma Separated Values (CSV) format.

SENT

Format compatible with Novell Sentinel/Log Manager products.

[-F, --filterFile] FILE-PATH

Specifies a filter file that contains include and exclude filter patterns to be applied to the auditing records that are received from the NSS Auditing Engine (`vigil`). See [“Filtering Records” on page 17](#) for further details.

[-h, --help]

Causes `vlog` to display a help page, then terminate.

[-l, --size] BYTES

Debugging option. (Do not use.)

[-o, --outputFile] FILE-PATH

Redirects output from `stdout` to the specified FILE-PATH.

[-p, --pattern] FILTER-PATTERN

Specifies a filter pattern to be applied to auditing records received from the NSS Auditing Engine (vigil). See [“Filtering Records” on page 17](#) for further details.

[-r, --recCnt] NUMBER-OF-RECORDS

Debugging option. (Do not use.)

[-u, --userKey] USER-KEY

Debugging option. (Do not use.)

[-U, --userName] USER-NAME

Debugging option. (Do not use.)

[-v, --incverbose]

Debugging option. Increments the vlog’s (error and warning) verbose level by one, and reports the verbose level. Use the -v option to specify a verbose level. Verbose messages are sent to stderr.

[-V, --setverbose] VERBOSE-LEVEL

Debugging option. Sets vlog’s (error and warning) verbose level to the specified value. Verbose messages are sent to stderr. Each level includes the messages of the lower levels.

Verbose Levels	Description
60 through 69	vlog debugging messages.
50 through 59	Filter-parsing notes.
40 through 49	Program-warning notes (mostly due to filter pattern issues).
30 through 39	Filter-related notes (why audit records match, or are excluded, by filters).
20 through 29	Internal modes are noted (mostly cursor-file related modes).
10 through 19	Configuration changes are noted (as per command-line options).
0 through 9	Normal (default). Only fatal errors are emitted.
-10 through -1	Silent. No error, or other, messages emitted.

For example, to set the verbose level to 22, enter

```
/opt/novell/vigil/bin/vlog -V 22
```

The verbose messages for fatal errors, configuration changes, and internal modes are sent to stderr.

--filterTest

Filter pattern debugging option. Causes vlog to validate filter patterns and terminate. Filter patterns specified with the [-p, --pattern] and [-F, --filterFile] options are validated.

Filtering Records

The `vlog` application supports filtering of events, as they are received from the NSS Auditing Engine (`vigil`), by using filter patterns. Filter patterns are rules for filtering events. You can use either of the following methods to specify filter patterns:

- ♦ A filter file of filter patterns (consisting of one filter pattern per line) can be specified with the `[-F, --filterFile]` command line option. This option must be followed by a `[path/]filename`.

A filter file can contain comment lines. Comment lines begin with a pound sign (`#`) or a double forward slash (`//`).

- ♦ Individual filter patterns can be specified with the `[-p, --pattern]` command line option. This option must be followed by a quoted filter pattern.

There are two kinds of patterns that can be specified from a filter file by using the `[-F, --filterFile]` option, or specified individually in the command by using the `[-p, --pattern]` option.

- ♦ Patterns for filtering records of type VIGIL
- ♦ Patterns for filtering records of type NSS, NCP, and CIFS

Each of these pattern types are discussed below.

Patterns for Filtering Records of Type VIGIL

Records of type VIGIL represent operations internal to the NSS Auditing Engine. By default, records of type VIGIL are not filtered from `vlog`'s output.

- ♦ [“Filter Syntax for Type VIGIL Records” on page 17](#)
- ♦ [“Filter Keywords for Type VIGIL Records” on page 18](#)
- ♦ [“Examples for Filtering VIGIL Events” on page 19](#)

Filter Syntax for Type VIGIL Records

The general pattern for filtering records of type VIGIL is:

```
:[+ or -]KEYWORD [[+ or -]KEYWORD]
```

A pattern used to filter records of type VIGIL has a colon (`:`) as the first character of the pattern.

The colon is followed by one or more keywords that represent records that are to be included or excluded from the `vlog` output. Multiple keyword entries are separated by a space. Keywords are applied in the order that they appear in the filter pattern.

The specified keyword causes specific records of type VIGIL to be included or excluded from the output. Each keyword is preceded by an exclude/include character that indicates whether the records that match the specified pattern should be excluded or included in the `vlog` output. A minus (`-`) character indicates that the records that are represented by the keyword that follows it should be excluded from the `vlog` output. A plus (`+`) character indicates that the records that are represented by the keyword that follows it should be included in the `vlog` output.

Filter Keywords for Type VIGIL Records

The keywords for VIGIL record types are as follows:

START

Each time the `vigil.ko` kernel module is loaded, a “Start” record is sent to all auditing clients.

STOP

Each time the `vigil.ko` kernel module is unloaded, a “Stop” record is sent to all auditing clients.

NCP_START

Each time the `vigil.ncp.ko` kernel module is loaded, an “NCP started” record is sent to all auditing clients.

NCP_STOP

Each time the `vigil.ncp.ko` kernel module is unloaded, an “NCP stopped” record is sent to all auditing clients.

NSS_START

Each time the `vigil.nss.ko` kernel module is loaded, an “NSS started” record is sent to all auditing clients.

NSS_STOP

Each time the `vigil.nss.ko` kernel module is unloaded, an “NSS stopped” record is sent to all auditing clients.

CIFS_START

Each time the `vigil.cifs.ko` kernel module is loaded, a “CIFS started” record is sent to all auditing clients.

CIFS_STOP

Each time the `vigil.cifs.ko` kernel module is unloaded, a “CIFS stopped” record is sent to all auditing clients.

CLIENT_START

Each time a new Auditing Client (an internal NSS Auditing Engine construct) is activated, a “Client started” record is sent to all auditing clients.

CLIENT_STOP

Each time a new Auditing Client (an internal NSS Auditing Engine construct) is deactivated, a “Client stopped” record is sent to all auditing clients.

USER_START

Each time a new Auditing Client User (an internal NSS Auditing Engine construct) is activated, a “User started” record is sent to all auditing clients.

USER_STOP

Each time a new Auditing Client User (an internal NSS Auditing Engine construct) is deactivated, a “User stopped” record is sent to all auditing clients.

ROLL

The NSS Auditing Engine (`vigil`) appends auditing records to a file in a directory specified by the auditing client application. The auditing client application can also specify the maximum size of the file in which auditing records are placed, and can optionally specify that when the file maximum size has been reached, the NSS Auditing Engine creates a new file (in the specified directory) and begins appending audit records to the new file.

When the NSS Auditing Engine creates a new file (in which audit records will be placed), it generates a “Roll” audit record, and appends it (as the last record) to the previously used file. The record contains the full (Linux) path of the newly created file, where audit record processing should continue. Roll records are sent only to the specifically affected auditing client, not to all auditing clients.

ALL

Used to indicate all records of type VIGIL.

Examples for Filtering VIGIL Events

The following are examples of how records of type VIGIL might be filtered from the `vlog` output by specifying individual patterns at the command line prompt:

```
/opt/novell/vigil/bin/vlog -p":-all"
```

Specifies a filter pattern that excludes all records of type VIGIL from the `vlog` output.

```
/opt/novell/vigil/bin/vlog -p":-all +roll"
```

Specifies a filter pattern that excludes all records of type VIGIL from the `vlog` output, except Roll records, which are shown in the `vlog` output.

```
/opt/novell/vigil/bin/vlog -p":-roll -user_stop -user_start"
```

Specifies a filter pattern that excludes Roll records, User stopped records, and User started records from the `vlog` output.

Keywords are applied in the order that they appear in the filter pattern. For example, the following patterns are not equivalent:

```
/opt/novell/vigil/bin/vlog -p":-all +roll"
```

Specifies a filter pattern that excludes all records of type VIGIL, but then allows the Roll record. Of all the VIGIL type records, only the Roll events are output.

```
/opt/novell/vigil/bin/vlog -p":"+roll -all"
```

Specifies a filter pattern that allows the Roll record, but then excludes all records of type VIGIL. No VIGIL type records (of any event type) are output.

Patterns for Filtering Records of Type NSS, NCP, and CIFS

Records of type NSS, NCP, and CIFS represent operations on files.

- ♦ [“Filter Syntax for Type NSS, NCP, and CIFS Records” on page 20](#)
- ♦ [“Negation Element Options” on page 20](#)
- ♦ [“Path Element Options” on page 21](#)
- ♦ [“Event Element Options” on page 26](#)

Filter Syntax for Type NSS, NCP, and CIFS Records

`[negation_element]path_element (event [event...])`

Patterns for filtering records of type NSS, NCP and CIFS consist of three elements in the following order:

1. **Negation Element:** Indicates whether records that match the specified filter patterns that follow are to be included or excluded from the auditing log. The negation element, if present, is immediately followed by the path element.
2. **Path Element:** A filename-matching pattern or directory-name-matching pattern that specifies directories or files to include or exclude from the audit log. The path element is delimited from the event element by a [space or tab] character.
3. **Event Element:** A list of NSS file system events enclosed in parenthesis to include or exclude from the audit log.

The options for each of the elements is described below.

Negation Element Options

The negation element is a single character that is used to indicate whether the specified filter patterns are to be included or excluded from the auditing log.

The negation element is a single character that precedes the path element:

! (exclamation mark character)

Used to negate the filter patterns specified in a filter file when the command line filter file option `[-F, --filterFile]` is used.

- (minus character)

Used to exclude (negate) a filter pattern that is specified on the command line when the filter pattern option `[-p, --pattern]` is used.

+ (plus character)

Used to include (non-negate) a filter pattern that is specified on the command line when the filter pattern option `[-p, --pattern]` is used.

The negation element, if present, is immediately followed by the path element.

Audit records that match a negated filter pattern are excluded from the `vlog` output. Specifically, `vlog` uses the following logic to include and exclude audit records from being output:

1. If there are no include (non-negated) filter patterns specified (either in the filter pattern `[-p, --pattern]` option or the filter file `[-F, --filterFile]` option), include all audit records in the output.
2. If one or more filter patterns are specified:
 - a. If the audit record does not match any of the include (non-negated) filter patterns, do not output the record.
 - b. If the audit record is not excluded by 2a above, and if the audit record matches any of the exclude (negated) filter patterns, do not output the record.

Path Element Options

The path element of the filter pattern is a filename-matching pattern or directory-name-matching pattern that specifies directories or files to include or exclude from the audit log.

The path element immediately follows the negation element (if present).

The path element is delimited from event element by a [space or tab] character. Thus, if a path element contains a space or tab character, you must do one of the following: Enclose the path element in quotation marks (such as "VOL1:/xyz dir/"), or escape each space or tab character within the path element by preceding it with a backslash (\) character (such as VOL1:/xyz\ dir/).

The event element follows the path element delimiter (space character or tab character).

- ♦ [“Path Element Wildcard Characters” on page 21](#)
- ♦ [“Path Element Examples” on page 22](#)

Path Element Wildcard Characters

The syntax for the filename and directory name pattern allows for the following wildcard characters:

?

Using the question mark (?) wildcard matches any single character, except for a forward slash character (/).

*

Using the single asterisk (*) wildcard matches any sequence of zero or more characters, except for a forward slash character (/).

**

Using the double asterisk (**) wildcard matches any sequence of zero or more characters, including a forward slash character (/).

[chars]

Using the [chars] wildcard matches any single character in chars. If chars contains a sequence of the form a-b, then any character between a and b (inclusive) will match. The forward slash (/) cannot be specified within chars. A leading or trailing minus character (-) simply includes the minus as one of the characters in the group.

\x

Using the backslash before a character matches the character specified, except for the forward slash character (/).

{a,b,...}

Using the list of strings matches any of the strings a, b, and so on.

NOTE: Generally, the forward slash character (/) must be matched explicitly. The only exception is in the use of a double asterisk (**).

For path element examples, see [“Path Element Examples” on page 22](#).

Path Element Examples

This section provides examples of path elements and a description of how each might be applied.

/a[-e]?/joke

Filename	Matches (Yes or No)
/a-h/joke	Yes.
/adh/joke	No. The [e-] group only includes "e" and "-", not "d".
/aeh/joke	Yes.

/a[c\ -e]?/joke

Filename	Matches (Yes or No)
/a-h/joke	Yes.
/ach/joke	Yes.
/adh/joke	No. The [c\ -e] group does not include "d".
/aeh/joke	Yes.

/a[d-f]?/joke

Filename	Matches (Yes or No)
/aez/joke	Yes.
/agf/joke	No. Need character from [d-f] group.

/a[d-fs-u]?/joke

Filename	Matches (Yes or No)
/aef/joke	Yes.
/aft/joke	Yes.

/a[def]?/joke

Filename	Matches (Yes or No)
/ad/joke	No. No character matches the "?".
/aef/joke	Yes.
/agf/joke	Yes.

/a[def][hij]?/joke

Filename	Matches (Yes or No)
/afh/joke	No. No character matches the "?".
/afhz/joke	Yes.
/agfh/joke	No. Need character from [def] group.

/a[e-]?/joke

Filename	Matches (Yes or No)
/a-h/joke	Yes.
/aeh/joke	Yes.
/afh/joke	No. The [e-] group only includes "e" and "-", not "f".

/a*

Filename	Matches (Yes or No)
/a/b/c/d	No. "*" does not match the "/" character.

/a*?/a

Filename	Matches (Yes or No)
/a/a	No. The "?" in the pattern does not match "/".
/ab/a	Yes.
/abb/a	Yes.

/a*?\a

Filename	Matches (Yes or No)
/ab/a	No. An escaped "/" in the pattern cannot match "/".

/a**

Filename	Matches (Yes or No)
/a/b/c/d	Yes.

/a?/a**

Filename	Matches (Yes or No)
/a/a	No. No character matches the "?".
/a/b/c/a	Yes.

/a/**

Filename	Matches (Yes or No)
/a/b/c/d	No. Must end with the "/" character.
/a/b/c/d/	Yes.

/a*/

Filename	Matches (Yes or No)
/a/b/c/d	No. "*" does not match the "/" character.

/a*/b/c//e/f**

Filename	Matches (Yes or No)
/a/b/b/c/d/d/d/d/e/f	Yes.
/a/b/b/c/d/d/d/d/e/f/e/f/e/f	Yes.
/a/b/b/c/d/d/d/e/f/	No. Must end with the "f" character.
/a/b/c/d/e/e/e/e/f	No. Need something between a/ and /b/c.
/a/xxx/b/d/e/e/e/e/f	No. /a/xxx/b must be followed by /c.

/a*/b/c//e/f/**

Filename	Matches (Yes or No)
/a/b/b/c/d/d/d/d/e/f	No. Must end with the "/" character.

/a*/d/e

Filename	Matches (Yes or No)
/a/d/e/	No. Must end with the "e" character.
/a/def/d/e	Yes.
/a/def/d/e/	No. Must end with the "e" character.

abc,{,,,{,x,,},,,,}def

Filename	Matches (Yes or No)
abc,def	Yes.

abc,}{,,,{,x,,},,,,}def

Filename	Matches (Yes or No)
abc,}}def	Yes.

abc{,,,{,x,,},,,,}def

Filename	Matches (Yes or No)
abcdef	Yes.
abcxdef	Yes.
abcydef	No. Nothing after "abc" allows "y".

abc{}def

Filename	Matches (Yes or No)
abcdef	Yes.

abc*{def,xyz,hij,{a*[d-g],b*[7-9]}z}qrt*m

Filename	Matches (Yes or No)
abcadzqrtm	Yes.
abcaezqrtm	Yes.
abcb6zqrtm	No. Nothing after "abcd" would allow "6".
abcb8zqrtm	Yes.
abcdefadzqrtm	Yes.
abcdefdefqrtm	Yes.
abcdefqrtm	Yes.
abchijqrtm	Yes.
abcxyqrtm	No. Nothing after "abc" would allow "xyq".
abcxyzdefqrtm	Yes.
abcxyzqrtm	Yes.

For simplicity, the NSS volume name was omitted from the above examples. NSS audit records emitted by the NSS Auditing Engine (*vigil*) have paths that include the NSS volume name. In order to match these NSS audit records, the NSS volume name must be specified as part of the path element, similar to the following:

VOL1:/a*/a

Filename	Matches (Yes or No)
VOL1:/ab/a	Yes.

VOL2:/a*/a

Filename	Matches (Yes or No)
VOL1:/abb/a	No.

VOL2:/a*/a**

Filename	Matches (Yes or No)
VOL2:/a/b/c/a	Yes.

Event Element Options

The event element consists of a list of events enclosed in parentheses. The events listed in the parentheses are delimited by a [space or tab] character.

The event element follows the path element delimiter (space character or tab character).

- ♦ [“Event Types” on page 26](#)
- ♦ [“Event Type Examples” on page 26](#)
- ♦ [“Event Sub-Types NSS, NCP, and CIFS” on page 27](#)
- ♦ [“Event Sub-Type Examples” on page 28](#)

Event Types

Valid event options are:

DELETE
CREATE
OPEN
CLOSE
RENAME
MODIFYMETADATA
ADDTRUSTEE
REMOVETRUSTEE
SETINHERITEDRIGHTS
LINK

The asterisk character (*) can be used to specify all events.

The exclamation mark (!) or minus (-) characters can precede an event name to exclude (negate) events of that type.

The event options can occur in any order in the parenthesis. When parsing an event list, a list of included (non-negated) events is first created, then the excluded (negated) events (that is, those with the [! or -] exclusion character in front of them) are removed from that list.

Event Type Examples

The following are examples of event element patterns and a description of the results you can expect for each one:

(*)

Includes all elements.

(OPEN CLOSE RENAME)

Includes only the OPEN, CLOSE, and RENAME events.

(* !OPEN)

Includes all events except OPEN.

This list could also have been specified as **(!OPEN *)**. All excluded (negated) events are removed after first creating a list of events that are included (non-negated).

(OPEN CLOSE !RENAME)

Includes only the OPEN and CLOSE events.

The **!RENAME** is not necessary here because it was never included. Typically, you add the excluded events in the list only when you use the asterisk (*) to specify all events.

(!* OPEN CLOSE)

Excludes all events. No events are included because all events were excluded.

This example illustrates that you can put the **[! or -]** character in front of the asterisk (*).

The OPEN and CLOSE event names have no effect because all excluded (negated) events are processed after the included (non-negated) events. In effect, this list includes the OPEN and CLOSE events, then excludes all events.

(OPEN !OPEN CLOSE)

Includes only the CLOSE event.

The OPEN and !OPEN cancel each other out.

Event Sub-Types NSS, NCP, and CIFS

Each event in the list can also specify a list of event sub-types to include or exclude. These are specified in parentheses immediately after the operation name. Event sub-types are delimited by a space or tab.

Valid event sub-types are:

NSS

NCP

CIFS

The asterisk (*) can be used to include all event sub-types.

If no sub-types are specified, they are all included by default.

The exclamation mark (!) or minus (-) characters can precede a sub-type to exclude it.

The sub-types can occur in any order in the parentheses. When parsing a sub-type list, a list of included (non-negated) sub-types is created first, then the excluded (negated) sub-types (that is, those with the **[! or -]** exclusion character in front of them) are removed from that list.

Event Sub-Type Examples

This section provides examples of event element patterns and a description of how each might be applied.

(*)

Matches all events.

(OPEN)

Matches all `OPEN` events (including the `NSS`, `NCP`, and `CIFS` sub-types).

(OPEN CLOSE)

Matches only the `OPEN` and `CLOSE` events (including the `NSS`, `NCP`, and `CIFS` sub-types).

(OPEN(NSS) CLOSE(NSS))

Matches only the `NSS OPEN` and `NSS CLOSE` events.

(* !OPEN)

Matches all events (including the `NSS`, `NCP`, and `CIFS` sub-types) except the `OPEN` event.

(* !OPEN(NSS) !CLOSE(NSS))

Matches all events except the `NSS OPEN` and `NSS CLOSE` events.

(* !OPEN(* !NSS) !CLOSE(* !NSS))

Matches all events except the non-`NSS OPEN` and non-`NSS CLOSE` events.

Filter Pattern Examples

This section provides examples of filter patterns and a description of how each might be applied. These examples are specific to entries in the filter file (when using the `[-F, --filterFile]` option).

VOL1:/abc/* (*)

Include: Matches all events on any file in the `VOL1:/abc` directory. The events on files in subdirectories are not included.

VOL1:/abc/ (*)**

Include: Matches all events on any file in the `VOL1:/abc` directory and any of its subdirectories.

VOL1:/abc/* (OPEN CLOSE)

Include: Matches only `OPEN` and `CLOSE` events on any file in the `VOL1:/abc` directory. The events on files in subdirectories are not included.

VOL1:/abc/* (* !OPEN !CLOSE)

Include: Matches all events except the `OPEN` and `CLOSE` events on any file in the `VOL1:/abc` directory.

VOL1:/abc/* (OPEN(NSS) CLOSE(NSS))

Include: Matches only `NSS OPEN` and `NSS CLOSE` events on any file in the `VOL1:/abc` directory.

VOL1:/abc/* (* !OPEN(NSS) !CLOSE(NSS))

Include: Matches all events except the NSS OPEN and NSS CLOSE events on any file in the VOL1:/abc directory.

VOL1:/abc/* (* !OPEN(* !NSS) !CLOSE(* !NSS))

Include: Matches all events except the non-NSS OPEN and non-NSS CLOSE events on any file in the VOL1:/abc directory.

!VOL1:/abc/def (*)

Exclude: Matches all events for the VOL1:/abc/def file. This exclusion causes all events on the VOL1:/abc/def file to be dropped (assuming that they had been included by an include rule).

!VOL1:/abc/def (OPEN)

Exclude: Matches OPEN events for the VOL1:/abc/def file. This exclusion effectively excludes OPEN events on the VOL1:/abc/def file.

VOL1:/abc/def (* !OPEN)

Include: Matches all events except the OPEN event on the VOL1:/abc/def file. This is another way to exclude OPEN events on the VOL1:/abc/def file.

!VOL1:/abc/def (* !OPEN)

Exclude: Matches all events except the OPEN event on the VOL1:/abc/def file. This effectively excludes all events except the OPEN event on the VOL1:/abc/def file (assuming they had been included by an include rule).

"VOL1:/xyz dir/*" (*)

Include: Matches all events on any file in the "VOL1:/xyz dir" directory. Quotation marks are used to enclose the pattern when the path contains spaces. Otherwise, the space in "xyz dir" is treated as a delimiter between the pattern and the events.

VOL1:/xyz\ dir/* (*)

Include: Matches all events on any file in the "VOL1:/xyz dir". A backslash (\) is used to escape the space character in the directory name "xyz dir". Otherwise, the space in "xyz dir" is treated as a delimiter between the pattern and the operations.

Troubleshooting

Orphaned Auditing Client

The NSS Auditing Engine (*vigil*) implements an interface that allows user-space applications (such as *vlog*) to establish Auditing Clients. When doing so, the user-space application (such as *vlog*) specifies various parameters, such as a directory in the Linux file system where the auditing record log files are placed.

After an Auditing Client has been established, the NSS Auditing Engine (*vigil*) architecture has been designed to store the auditing records in files in the directory specified by the auditing application (such as *vlog* or Novell Sentinel).

Records are stored in the files until the Auditing Engine is instructed to stop, or until the NSS Auditing Engine is stopped. If the auditing application terminates (perhaps unexpectedly), and the NSS Auditing Engine is therefore not instructed to stop sending records to the Auditing Client's directory, the NSS Auditing Engine continues to store auditing records in the Auditing Client's specified directory.

An Auditing Client that does not have a live user-space application associated with it is called an Orphaned Auditing Client. The architecture of the NSS Auditing Engine supports this mode of operation. This mode facilitates the continued collection of auditing data, even if the auditing application temporarily fails. The NSS Auditing Engine architecture assumes that the auditing application will eventually be restarted, and will then re-connect to the auditing stream.

The default configuration of the `vlog` application does not attempt to re-connect to an orphaned Auditing Client from a previous failed `vlog` session. If `vlog` is not properly terminated by the `SIGTERM` signal, an Orphaned Auditing Client is created.

IMPORTANT: If Orphaned Auditing Clients are not stopped, they continue until they fill the Linux file system partition with auditing data.

You can use one of the following methods to eliminate Orphaned Auditing Clients: Start and stop (or restart) the NSS Auditing Engine, or stop a specific instance of the Auditing Client. Each method is described below.

Method 1: Stop and Start (or Restart) the NSS Auditing Engine

To do this, enter the following commands as the `root` user at a terminal console prompt:

```
./etc/init.d/novell-vigil stop  
./etc/init.d/novell-vigil start
```

Or you can enter the following command to restart the engine:

```
./etc/init.d/novell-vigil restart
```

This method stops all Auditing Clients, including those that were not associated with the `vlog` application. This might be undesirable because some auditing records of file-system events will not be logged to the various auditing applications.

Method 2: Stop a Specific Auditing Client Instance

By default, all active Auditing Clients for the NSS Auditing Engine can be listed by listing the directory content of the `/sys/audit/vigil` directory.

For example, enter the following command as the `root` user at a terminal console prompt:

```
ll /sys/audit/vigil
```

All active Auditing Clients are represented in the listing as directories named "`CLIENT_*`". Using the `[-C, --clientName]` option, `vlog` Auditing Clients can be given a name such as "`JOHN`", and the specific entry in the `/sys/audit/vigil/` directory will be "`CLIENT_JOHN`". If the `[-C, --clientName]` option is not specified, `vlog` generates a random Auditing Client name. Generated name entries are prefixed with "`CLIENT_VLOG_`", followed by the process ID that created the client, followed by a numeric value that represents the date and time that the specific Auditing Client was started.

For example, a `vlog` generated client name might be:

```
CLIENT_VLOG_31691-1264200226
```

In order to stop a specific instance of a `vlog` Orphaned Auditing Client, that client's entry in `/sys/audit/vigil/` must be accurately identified. The `root` user can stop a specific instance of an auditing client by writing a `STOP` command to the client's `CONTROL` file (found in that client's directory). The `ClientKey` must also be specified as an additional credential. This limits closing an Auditing Client to a `root` user who knows the Auditing Client's `ClientKey`. The `ClientKey` can be specified by using `vlog`'s `[-c, --clientKey]` option. If the `[-c, --clientKey]` option is not specified, `vlog` uses the default client key "Zarahemla".

For example, enter the following command as the `root` user at a terminal console prompt:

```
echo 'CLOSE ClientKey="Zarahemla"' > \  
> /sys/audit/vigil/CLIENT_VLOG_31691-1264200226/CONTROL
```

IMPORTANT: When an Auditing Client is closed, the client's directory is removed. You should not close an Auditing Client if the client directory is the current working directory.

Authors

Copyright 2009–2010, Novell, Inc. All rights reserved. <http://www.novell.com>

See Also

To report problems with this software or its documentation, visit <http://bugzilla.novell.com>

