IBM Security Verify Access
Version 10.0.3
December 2021

Auditing topics

Contents

Figures

Tables

Chapter 1. Auditing overview

Auditing is the process of maintaining detailed and secure logs of critical activities in a business environment.

These activities can be related to security, content management, business transactions, or other such activities.

For example, the following activities can be audited:

- Login failures
- Unauthorized access to protected resources
- Modification to security policy

Use the method that is provided in Chapter 4, "Native Security Verify Access auditing," on page 11 to manage audit events with the native Security Verify Access approach.

For information about managing statistical events, see "Working with local statistics" on page 37. For information about WebSEAL HTTP events, see "WebSEAL HTTP logging" on page 31.

Auditing versus diagnostics

Security Verify Access provides ways to collect events that you can use for diagnostic and auditing purposes of the servers. Events for diagnostics and auditing pertain to the operations of the servers.

To enable diagnostics and auditing, define which types of events to capture. You can write recorded events to one or a combination of the following files or devices:

- Log file.
- Standard output (STDOUT) device.
- Standard error (STDERR) device.

Beyond these destinations, when events are captured, they can be redirected to a remote authorization server or redirected to an application for processing.

Audit events

For auditing purposes, define which audit, statistic, or other type of events to capture.

You can use events to create snapshots of various server activities. You can record audit events by using the native Security Verify Access support.

To configure auditing events, define stanza entries in the configuration files. Depending on your approach, you define different stanza entries in different configuration files.

Use the following guidelines for defining the auditing configuration:

- For audit events, define `logcfg` entries in the `[aznapi-configuration]` stanza of the server configuration file.
- For HTTP request events, define entries in the `[aznapi-configuration]` and `[logging]` stanzas of the WebSEAL configuration files for HTTP events that you want to record.

Diagnostic events

For diagnostic information, define which message events and which trace events to capture. These events can help you troubleshoot problems.

To configure diagnostic events, you must define statements in the server-specific routing files. Each server has an associated routing file. The statements in these routing files are for both message events and trace events. You define the statements for message events by severity level. You can define the statements for trace events by trace level and optionally by component.

For more information about message and trace events, see the Troubleshooting topics in the IBM Knowledge Center.

Audit trails

IT organizations can use information that is contained in audit trails to help them show compliance with government regulations such as the following regulations:

- Sarbanes-Oxley (SOX) Act.
- The Health Insurance Portability and Accountability Act (HIPAA).
- The Basel II international banking accord.

For these reasons, such audit trails must be sometimes maintained for years.

Audit trails are useful to check enforcement and effectiveness of IT controls, for accountability and vulnerability, and for risk analysis. IT organizations can also use auditing of security-related critical activities to aid in forensic investigations of security incidents.

When a security incident occurs, audit trails enable analysis of the history of activities that occurred before the security incident. This analysis might answer questions such as who did what, when, where, and how. Based on this analysis, appropriate corrective actions can be taken. For these reasons, audit trails must be archived and accessible for years.

Audit trails can be established in relational databases that are easily queried to generate reports. When audit trails are written to relational databases, reporting tools can be used to display reports. Reports can fall into the following categories:

- Trend reports provide summarized audit data that you can use to assess whether there is any long-term rise or fall in questionable activity. Trend reports can help provide a "security pulse" for an organization.
- Operational reports allow a detailed review of audit data to help determine the cause of a security incident.

Audit records for HTTP access

The generation of audit records for HTTP access to WebSEAL can use large quantities of disk space quickly. You can reduce the volume of audit events that are generated by using the following strategies:

- Generate events for unsuccessful HTTP accesses only.
- Selectively disable the generation of events by using attached protected object policies (POPs).

For details about reducing records by generating events for unsuccessful accesses only, see “Native auditing” on page 3 if you are using native Security Verify Access auditing.

For details about using POPs to selectively disable the generation of audit events, see “Disabling resource access events” on page 29.

Chapter 2. Overview of Security Verify Access event logging

For auditing and other serviceability purposes, Security Verify Access uses a structured hierarchy of events. This hierarchy is built dynamically and allows runtime-associations to be made between event categories and the log agents that record those events.

Figure 1 on page 3 shows the hierarchy of Security Verify Access events in the event pool.

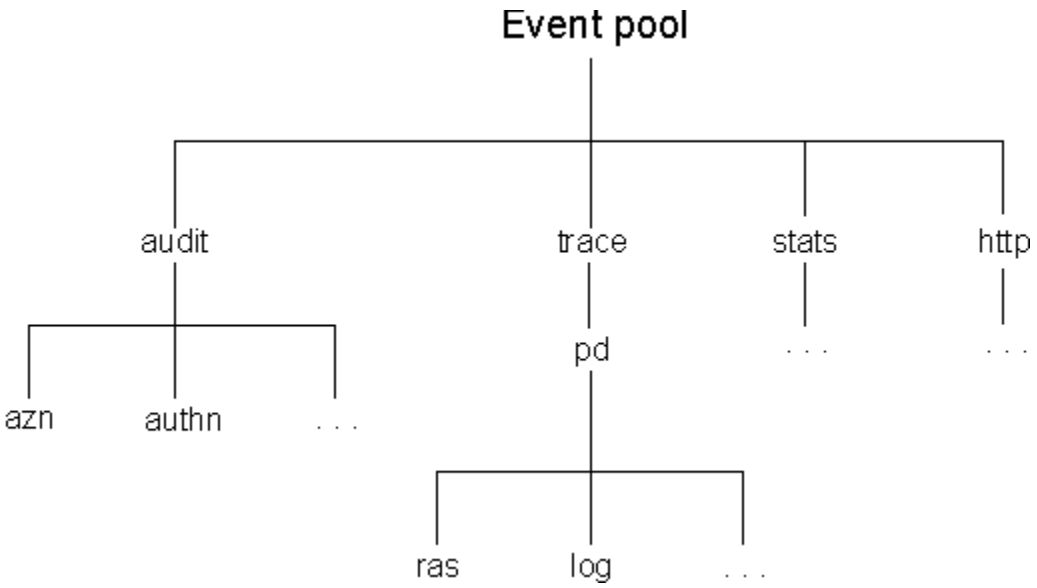

Figure 1. Event pool hierarchy

Natively, Security Verify Access generates and can record the following primary categories of events:

Audit events
For information about audit events, see "Audit event logging" on page 11.

HTTP request events
For information about HTTP request events, see "WebSEAL HTTP logging" on page 31.

Statistical events
For information about statistical events, see "Working with local statistics" on page 37.

Trace events
For information about trace events, see the Troubleshooting topics in the Knowledge Center.

Native auditing

Auditing is defined as the logging of audit records. It includes the collection of data about system activities that affect the secure operation of the Security Verify Access server processes. Each Security Verify Access server can capture audit events whenever any security-related auditable activity occurs.

Auditing uses the concepts of a record, an audit event, and an audit trail. Each audited activity is called an *audit event*. The output of a specific server event is called a *record*. An *audit trail* is a collection of multiple records that document the server activity.

When configuring for auditing, think about the source of the events that you want to capture. Audit trail files can capture authorization, authentication, and management events that are generated by the Security Verify Access servers. There are multiple sources for auditing events that you want to gather. You can collect either a combination or all the different types of auditing events at the same time. Table 1 on page 4 shows some of the event types that can be used for native auditing.

Table 1. Categories and description of native audit events	
Event category	**Description**
audit.authz	Authorization events for WebSEAL servers
audit.azn	Authorization events for base servers
audit.authn	Authentication, credential acquisition authentication, password change, and logout events
audit.authn.successful	Successful authentication credential acquisition authentication, password change, and logout events
audit.authn.unsuccessful	Failed authentication credential acquisition authentication, password change, and logout events
audit.http	HTTP access events
audit.http.successful	Successful HTTP access events
audit.http.unsuccessful	Failed HTTP access events
audit.mgmt	Management events
http	HTTP logging information
http.clf	HTTP request information defined by the request-log-format configuration entry in the `[logging]` stanza. `clf` stands for common log format.
http.ref	HTTP Referrer header information
http.agent	HTTP User Agent head information

Statistics gathering

Security Verify Access servers provide a series of modules that can monitor and collect information about specific server activity. After enabling a module, you can display the statistical information that it gathered since it was enabled. In addition to displaying this information, you can direct these statistics to a log file.

You can work with statistics with the **`server task stats`** command or with stanza entries in the configuration file for the specific server.

When you display statistics, you see a snapshot of the statistics. These statistics provide a view of the recorded activity. If you capture statistics at regular intervals, you can determine trend analyses against the server activities.

For information about enabling and working with the statistics gathering modules, see "Working with local statistics" on page 37.

Logging process

Figure 2 on page 5 depicts the relationships among the steps in the logging process. The top part of the figure represents the code of a Security Verify Access server. The code contains probe points where events of specific types can be generated. Generated events are submitted to the server event pool for possible recording through a point of capture (event sink). The event pool defines the events category.

At run time, you can subscribe a log agent at any point in the event pool hierarchy. You can selectively record events that are generated at the probe points for the program. The middle part of the figure depicts subscription.

For example, you can subscribe to a remote client for capturing events. This client forwards the selected events to a remote authorization server.

The lower part of the figure depicts this remote server. Relayed events are placed in the event pool at the remote probe points for the authorization server.

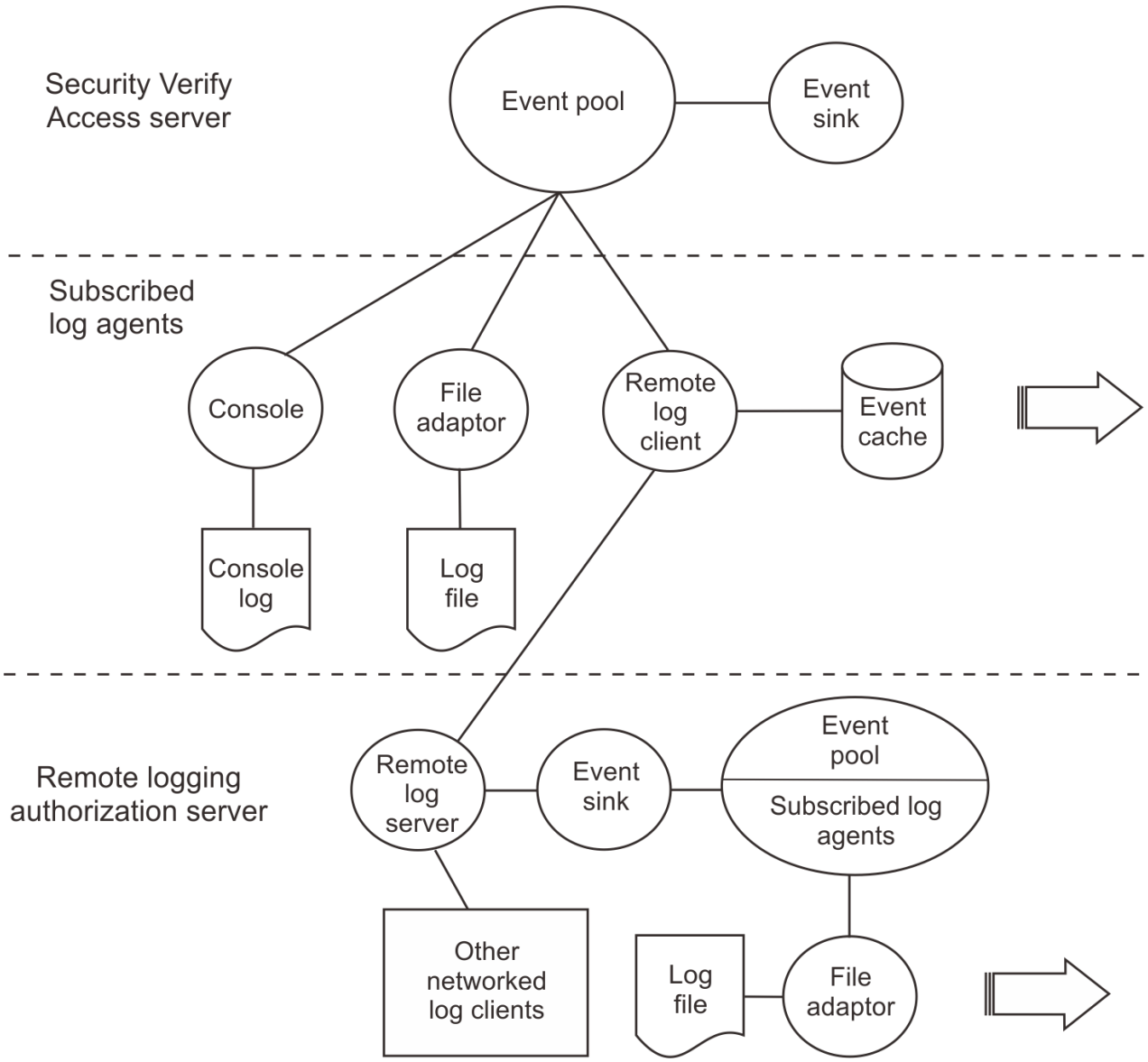

Figure 2. Application-specific probe points

Audit data in UTF-8 format

Security Verify Access produces audit data that uses UTF-8 encoding. When the operating system uses a non-UTF-8 code page, Security Verify Access converts the data to a format that matches the non-UTF-8 code page. In some cases, the conversion can result in data loss. For this reason, run Security Verify Access in an environment that uses UTF-8 encoded code pages.

When the operating system does not use a UTF-8 code page, the conversion to UTF-8 can result in data loss. When data loss occurs, the log file contains a series of question mark (?) characters at the location where the data conversion was problematic.

When running in a non-UTF-8 locale, use the `UTF8FILE` type in the routing file. For more information about the `UTF8FILE` type, see Chapter 6, “Routing files,” on page 183.

Chapter 3. Configuring auditing on the appliance

Use the Audit Configuration feature to enable logging of audit events.

Before you begin

Depending on the required audit configuration, you might need the following information to complete the auditing configuration:

- If you plan to use a syslog server on a remote machine, ensure that you have the information of the location of the syslog server.
- If you plan to use a TLS type protocol, ensure that the server certificate was imported into the chosen certificate database.
- If you plan to use client certificate to authenticate to the syslog server, ensure that the certificate is trusted by the syslog server. The certificate must be imported into the chosen certificate database.

About this task

IBM Security Verify Access provides the capability of collecting and processing system log (syslog) messages. Enable the feature by completing the steps in the audit configuration page to use a common auditing configuration that is used by all runtime components.

Procedure

1. From the top menu, select **Monitor Analysis and Diagnostics** > **Logs** > **Audit Configuration**.
2. Select **Enable audit log**.
3. Specify the location of the syslog server.

   On this appliance

   Audit events are sent to a syslog server on this appliance. If you select the local syslog server, no additional mandatory configuration is needed. If you want to tune the default configuration settings, proceed to step “5” on page 8.

   Note: If you configure auditing to use a local syslog server, see Viewing application log files, to view the audit logs.

   On a remote machine

   Audit events are sent to a syslog server on a remote machine. If you select a syslog server on a remote machine, you might need to specify some or all of the following information:

Table 2. Syslog server remote machine configuration values.

Field	Default Values	Description
Host	None	Specifies the host name of the syslog server.
Port	514	Specifies the port of the syslog server.
Protocol	UDP **Note:** Though UDP is the default value, use TLS. TLS is the preferred protocol for production environments.	Specifies the type of transport protocol to use to transmit syslog messages.

Table 2. Syslog server remote machine configuration values. (continued)		
Field	**Default Values**	**Description**
Certificate database (truststore)	None	Specifies the truststore to use to validate the certificate of the syslog server. This field is enabled only when the transport layer protocol type selected is TLS.
Enable client certificate authentication	Disabled	If enabled, the client is able to do client certificate authentication during the SSL handshake upon server request.
Certificate database (keystore)	None	Specifies the keystore to use for client certificate authentication. This field is enabled only when the enable client certificate authentication is selected.
Certificate label	None	Specifies the personal certificate to use for client certificate authentication. This field is enabled only when the enable client certificate authentication is selected.
Enable disk failover	Disabled	If enabled, audit events are logged to a local disk file when an error occurs during the SSL connection to the remote syslog server. **Note:** If you enable disk failover the audit events are logged to local disk files that follow the naming pattern **`ISVAAudit0.log.nn`**, where **nn** is a number that uniquely identifies a local disk file. The local disk file can be viewed at the same location as the local syslog server audit logs.

4. If you choose to use default values for tuning, you can complete the configuration by clicking **Save**. Otherwise, proceed with the subsequent steps. If you want to discard the changes you made, click **Refresh**.
5. Click **Tuning**. Provide the following information:

Table 3. Audit tuning values		
Field	**Default Value**	**Description**
Event Queue Size	**`1000`**	Specifies the maximum number of audit events that the event queue can hold. Syslog messages are queued in the memory before they are sent to the syslog server.

Table 3. Audit tuning values (continued)		
Field	**Default Value**	**Description**
Queue Full Timeout (seconds)	**-1**	Specifies the number of seconds to wait before an incoming event is discarded when the queue is full. A value of **0** indicates that new events are discarded immediately if the queue is full. A value of **-1** indicates that new events wait perpetually for the queue to have a vacancy.
Sender Threads	**1**	Specifies the number of sender threads, which drain the audit events from the queue to send to the syslog server.
Error Retry Count	**2**	Specifies the number of times the syslog client tries to establish a connection with the server again if it fails in the first attempt.

6. Click **Save**. Otherwise, click **Refresh** to discard the changes you made.

Chapter 4. Native Security Verify Access auditing

Audit event logging

To enable logging, define entries in the configuration file.

Procedure

1. Specify the type of audit event.
2. Specify the location of the audit log.

   Note: On Windows operating systems, newly created files are given "Full Control" permissions or inherit permissions from the parent directory. To protect audit files from possible tampering, manually modify the permission settings to "Read & Execute" on newly created files and on any parent directory.
3. Specify the maximum file size.
4. Specify the file flush interval.

Log agents

With event logging, the concept of a *log agent* includes capturing events that are redirected to destinations other than the local file system. Event logging uses the following types of log agents, each agent represents an audit trail:

- "Sending events to the console" on page 14
- "Configuring file log agents" on page 15
- "Configuring remote log agents" on page 20

Configuring audit events

Independent of the logging agent, configure which audit events to capture by using the `logcfg` entry.

When using the Security Verify Access approach, define the `logcfg` entry in any or all the following locations:

- The `[aznapi-configuration]` stanza of the policy server `ivmgrd.conf` configuration file
- The `[ivacld]` stanza of the authorization server `ivacld.conf` configuration file
- The `[aznapi-configuration]` stanza of a WebSEAL server `webseald.`*`instance`*`.conf` configuration file
- The `[aznapi-configuration]` stanza of the Plug-in for Web Servers pdwebpi.conf configuration file
- The `[aznapi-configuration]` stanza of the resource manager `aznAPI.conf` configuration file

Defining logcfg entries

When you define the `logcfg` entry in a configuration file, use the following general format (on a single line) to specify audit event logging:

```
logcfg = category:{stdout|stderr|file|remote}
[[parameter[=value]],
[parameter[=value]]],
...,
[parameter[=value]]]
```

To enable the recording of audit events, associate an event category with a log agent (`file` or `remote`) or associate an event category with a console destination (`stdout` or `stderr`).

When you define the parameters for any `logcfg` entry, be aware of the following conditions:

- Parameters can be specified in any sequence
- Parameter names are not case-sensitive
- Parameter names can be shortened to any unambiguous name
- Parameters differ by log agent
- Parameters are optional

Events for a category are inclusive of all subcomponents in the hierarchy. That is, a `foo.bar.fred` event is captured when the `foo.bar` category is defined.

You can attach multiple log agents to the same category. For example, the following configuration:

- Captures authorization audit events (category `audit.azn`) and uses a file agent to copy these events to the `audit.azn` file.
- Uses a pipe agent to relay these same events to the `analyse.exe` program.

```
[ivacld]
logcfg = audit.azn:file path=audit.azn
```

Parameters for the logcfg entry

The available parameters for the `logcfg` stanza entry differ by log agent.

Table 4 on page 12 shows which parameters are available for the `EventPool` category and the following log agents:

- File log agent
- Pipe log agent
- Remote agent
- Remote syslog agent

Table 4 on page 12 does not show the console log agent. The console log agent does not support parameters. For more information, see “Sending events to the console” on page 14.

Table 4. Available parameters for the logcfg stanza entry

Parameter	EventPool category	File log agent	Pipe log agent	Remote log agent	Remote syslog agent
buffer_size		Yes		Yes	
compress				Yes	
dn				Yes	
error_retry				Yes	Yes
flush_interval	Yes	Yes	Yes	Yes	Yes
hi_water	Yes	Yes	Yes	Yes	Yes
log_id		Yes			Yes
max_event_len					Yes
mode		Yes			
path		Yes	Yes	Yes	Yes
port				Yes	Yes
queue_size	Yes	Yes	Yes	Yes	Yes
rebind_retry				Yes	Yes

Table 4. Available parameters for the logcfg stanza entry (continued)					
Parameter	**EventPool category**	**File log agent**	**Pipe log agent**	**Remote log agent**	**Remote syslog agent**
rollover_size		Yes			
server				Yes	Yes
ssl_keyfile					Yes
ssl_label					Yes
ssl_stashfile					Yes
ssl_protocols					Yes

Configuring the event pool

Events are passed to subscribed log agents asynchronously from the application-level requests that construct the events. All events enter the common propagation queue before they are forwarded to the subscribed log agents.

The propagation queue is configurable. To configure the propagation queue, define the `logcfg` stanza entry with `EventPool` as the category name and specifies the configuration parameters without specifying a log agent.

Manage the propagation queue to support the configuration of log agents. For example, limit the amount of memory that is used to queue events for a remote log agent. To limit the amount of memory that is used, constrain the propagation queue with the `queue_size` parameter:

```
[aznapi-configuration]
logcfg = EventPool queue_size=number,hi_water=number,
     flush_interval=number_seconds
logcfg = category:remote buffer_size=number,path=pathname,
     server=hostname,queue_size=number
```

You can define the following parameters for pipe log agents:

flush_interval

Configure the `flush_interval` parameter to limit the amount of time that events can remain in the propagation queue. Specify the time in seconds. Assume that the size of the queue does not reach the high water mark within the specified interval. In this case, events in the queue are forwarded to the log agents.

The default value is `10` seconds. Specifying a value of `0` is equivalent to setting the value to `600` seconds.

hi_water

Configure the `hi_water` parameter to indicate the threshold where events in the propagation queue are forwarded to the log agents. Assume that the size of the queue does not reach this high water mark within the defined flush interval. In this case, events in the queue are forwarded to the log agents.

The default value is calculated as two-thirds of the configured queue size. If the queue size is `0` (unlimited), the high water mark is set to `100` events. If the high water mark is `1` event, each event in the queue is forwarded immediately to the log agents.

Setting a low value for the high water mark can have an adverse effect on performance.

queue_size

Because each event in the propagation queue consumes memory, configure the `queue_size` parameter to define the maximum number of events that the propagation queue can hold. If the maximum size is reached, the event-producing thread is blocked until space is available in the queue.

Blocking corresponds to throttling back the performance of the event-producing thread to a rate that can be consumed by the logging threads.

The default value is `0`. Specifying a value of `0` indicates that no size limit is enforced on the propagation queue. The propagation queue can grow to an unmanageable size when:

- You use the default value, and
- The logging threads cannot process events as they enter the propagation queue.

Sending events to the console

Logging to the console is the easiest event logging option to configure. Associate an output destination of standard out or standard error with the category of events in the event pool to capture:

```
[ivmgrd]
logcfg = category:{stdout|stderr}
```

Logging to the console does not use any queuing. The events are written to the console as they are received from the propagation queue. Depending on the queue settings, events might be delayed in the propagation queue.

If you are using console output and running a server in the foreground for debugging purposes, you might want to set the propagation queue settings accordingly. For example, set the `hi_water` parameter to a low value.

Sending events to standard error

You might configure event logging to standard error.

Procedure

1. Edit the appropriate server configuration file. Each server provides its own stanza entry values in its configuration file.
2. Locate the stanza that contains the `logcfg` entries.
3. Define the `logcfg` entry and specify the event category to log and the destination of standard error.

   ```
   logcfg = category:stderr
   ```

4. Save and exit the configuration file.

Example

For example, to capture all audit events to standard error, define the following entry in the configuration file:

```
[ivmgrd]
logcfg = audit:stderr
```

Sending events to standard output

You might capture event logging to standard output.

Procedure

1. Edit the appropriate server configuration file. Each server provides its own stanza entry values in its configuration file.
2. Locate the stanza that contains the `logcfg` entries.
3. Define the `logcfg` entry and specify the event category to log and the destination of standard output.

   ```
   logcfg = category:stdout
   ```

4. Save and exit the configuration file.

Results

To capture all audit events to standard output, define the following entry in the configuration file:

```
[ivmgrd]
logcfg = audit:stdout
```

Configuring file log agents

To record events in a file, specify a log file configuration as follows:

```
[ivacld]
logcfg = category:file path=file_pathname, flush_interval=num_seconds,
rollover_size=number,max_rollover_files=number,log_id=logid,
queue_size=number,hi_water=number,buffer_size=number,mode={text|binary}
```

Parameter names can be shortened to any unambiguous name. For example, the `hi_water` parameter can be shortened to `hi`.

A file is opened only one time. The file opens according to the options in the first configuration entry that is processed when:

- Multiple configuration entries exist.
- You want to selectively capture events to the same file.
- You want to capture events at different points of the event pool hierarchy.

After a file was opened, further file configurations can use the following shorthand notation to record events to the same file:

```
[ivacld]
logcfg = category:file log_id=logid
```

Writing to a file can be a slow operation relative to the tasks that are generating events. Therefore, events are posted to a file log agent through a second level of queuing. This second level of event queuing is configured like the central event propagation queue, but has different default values.

Parameters for file log agents

You can define the following parameters for file log agents:

buffer_size

Reduce memory fragmentation and improve the performance of writing to a file by:

- Not queuing many small events individually to the file log agent.
- Buffering events into blocks of a nominated size before queuing for writing.

The `buffer_size` parameter specifies the maximum size message that the program attempts to construct by combining smaller events into a large buffer.

Buffers consist of only an integral number of events; events are not split across buffers. If any individual event exceeds that maximum configured size, the large event is recorded in a buffer of its own, exceeding the configured value. The default buffer size for logging to a file is `0` bytes. This value prevents buffering and each event is handled individually.

If a value is specified for the `buffer_size` parameter, events are packed into buffers of that size before queuing to the file log agent.

For example, around 10 events are packed into each buffer that is written to the file when:

- The value for the `buffer_size` parameter is set to 2 KB.
- Events are assumed to be about 256 bytes.

This process reduces the number of disk input/outputs (I/Os) that are made while logging to 10 percent of the equivalent non-buffering case.

A default queue size of 200 also consumes around 10 times the memory of a default configuration that did no buffering if:

- The buffer size was 2 KB.
- The event size was around 200 bytes.

This size is because the maximum queue size value has not been changed. However, the size of events being queued has increased tenfold.

flush_interval

The `flush_interval` parameter is a multiuse parameter.

Ensure that stream buffers are flushed to disk regularly. Configure the frequency with which the server asynchronously forces a flush of the file stream to disk. To configure this frequency, use the `flush_interval` parameter. The value that is defined for this parameter is `0`, `< 0`, or the flush interval in seconds.

Specifying a value of `0` results in the flushing of the buffer every 600 seconds.

Specifying a value of `< 0` results in the absolute value that is used as the asynchronous flush frequency. However, a stream flush is also forced synchronously after each record is written.

Events are consolidated into large buffers that is based on the value of the `buffer_size` parameter. However, the `flush_interval` parameter also might affect the size of buffer written. When a flush is scheduled, an in-memory, partially filled buffer is also queued for writing before it completes the buffer fill.

The event queue is triggered for processing at the flush interval rate. The trigger enables processing of events that were waiting for longer than the scheduled flush time. Such processing applies to a scenario when the queue does not reach the high water mark between scheduled flushes.

hi_water

Processing of the event queue is scheduled regularly at the configured flush interval. It also is triggered asynchronously by the queue size that reaches a high water mark on the event queue.

The default value is two-thirds of the maximum configured queue size. If the maximum queue size is zero, the high water mark is set to a default of `100`.

The transaction rates and the values of these options determine the maximum amount of memory that is consumed by enabling event logging to file.

If the event queue high water mark is set to 1, every event queued is relayed to the log agent as soon as possible. This setting is not optimal. Use it if you want to ensure that events get to disk as fast as possible. Doing so adversely impacts overall performance.

log_id

An open log file is associated with a short name identifier to facilitate the recording of events from different categories to the same file.

Use the `log_id` parameter to set the log file identifier (ID) explicitly; otherwise, it is given a default value. If the `path` parameter is specified, the default value is the configured path name. If the `path` parameter is not specified, the log ID defaults to the domain component of the event category being captured. For example:

```
logcfg = audit.azn:file
```

implies

```
log_id=audit
```

To capture events to a common file, set the log file ID to a suitable value in a fully optioned file configuration. Then, use the shorthand configuration variant to capture events from additional categories as shown:

```
[aznapi-configuration]
logcfg = audit.azn:file path=audit.log,
     rollover_size=-1,flush_interval=20,log_id=audit,
     ...
logcfg = audit.authn:file log_id=audit
```

Because of the default rules, this configuration is also equivalent to the following specification:

```
[aznapi-configuration]
logcfg = audit.azn:file path=audit.log,
     rollover_size=-1,
     ...
logcfg = audit.authn:file
```

If you construct a configuration where the log ID value does not match any open log file, no events are captured. For example, the following configuration does not record any events because the configuration line that initializes the log file was commented out:

```
[ivacld]
#logcfg = audit.azn:file path=azn.log,log_id=azn,...
logcfg = audit.authn:file log_id=azn
```

mode

Configure the `mode` parameter to open a file in either text or binary mode. For example:

```
[aznapi-configuration]
logcfg = audit.azn:file
...
mode={text|binary},
...
```

Text mode is deprecated on AIX, Linux, and Solaris operating systems. Binary mode on a Windows operating system writes the log file in an AIX®, Linux®, or Solaris-compatible format.

path

The path specifies the name and location of a log file. There is no default value, because the value of the `log_id` parameter takes precedence. An example for the WebSEAL audit trail file on AIX, Linux, and Solaris operating systems is as follows:

```
[aznapi-configuration]
logcfg = category:file path=audit.log
```

The directory portion of this path must exist. The log file is created if it does not exist.

queue_size

There is a delay between events being placed on the queue and the file log agent removing them. The `queue_size` parameter specifies the maximum size to which the queue is allowed to grow.

Consider that a new event is ready to be placed on the queue. Then, if the queue reaches the maximum size, the requesting thread is blocked until space is available in the queue. This process causes the performance of the event propagation thread to slow down to that of the file logging thread.

Limiting the queue size for the log agent must be configured with setting the queue size for the central event propagation queue. Unless the event propagation defined by the `queue_size` parameter is constrained appropriately, memory usage can still grow without bounds.

```
[aznapi-configuration]
logcfg = audit.azn:file
...
queue_size=number_events,
...
```

The default value is `0`. Specifying a value of `0` indicates that no limit is enforced on the growth of the unprocessed event queue. Correspondingly, the event propagation thread is not constrained by the speed of the logging thread. The unrecorded event queue can grow to an unmanageable size if:

- You are using the default.
- Events are being generated faster than they can be recorded to file.

rollover_size

Configure the `rollover_size` parameter to specify the maximum size to which a log file can grow. The default value is `2000000` bytes.

When the size of a log file reaches the specified rollover threshold, the existing file is backed up. The back-up happens to a file of the same name with the current date and time stamp appended. A new log file is then started.

The possible rollover size values are interpreted as follows:

- If the `rollover_size` value is less than zero, a new log file is created:
  - With each invocation of the process, and
  - Every 24 hours since that instance.
- If the `rollover_size` value is equal to zero, the log file grows until it reaches 2 GB and then rolls over. If a log file exists at startup, new data is appended to it.
- If the `rollover_size` value is greater than zero, the log file grows until it reaches the lesser of the following values and then rolls over:
  - The specified value
  - 2 GB

  If a log file exists at startup, new data is appended to it.

max_rollover_files

Configure the `max_rollover_files` parameter to specify the maximum number of rollover files to be kept on disk.

When the number of rollover log files reaches the specified threshold, the oldest log file is deleted.

The value of this configuration parameter is interpreted as follows:

- If the `max_rollover_files` value is blank or not specified, then no rollover files are deleted.
- If the `max_rollover_files` value is equal to zero, then only the current log file is kept, and all rollover log files are deleted.
- If the `max_rollover_files` value is greater than zero, then only that number of rollover log files are kept. When the number of rollover log files exceeds max_rollover_files, the oldest log file is deleted.

Sending events to a log file

You might configure Security Verify Access to send event records to a log file.

Before you begin

Before you begin this task, review the information in "Configuring file log agents" on page 15.

Procedure

1. Edit the appropriate server configuration file. Each server provides its own stanza entry values in its configuration file.
2. Locate the stanza that contains the `logcfg` entries.
3. Specify that the category is to send event records to a log file by using the following format:

```
category:file
```

For example, a category might be to audit authorization events (`audit.azn`):

```
logcfg=audit.azn:file
```

4. Specify the path to the log file:

```
path=fully_qualified_path
```

The default directories are:

AIX, Linux, and Solaris operating systems
`audit.log`

The file can be found by navigating to **Reverse Proxy** > **Manage** > **Logging**.

Windows operating systems
`audit.log`

The file can be found by navigating to **Reverse Proxy** > **Manage** > **Logging**.

The default file name depends on the type of logging being completed, such as `audit.log`

5. Specify the identifier for the log file:

```
log_id=logid
```

Use the `log_id` parameter to set the log file identifier (ID) explicitly; otherwise, it is given a default value. If the `path` parameter is specified, the default value is the configured path name. If the `path` parameter is not specified, the log ID defaults to the domain component of the event category that is being captured. For example, `logcfg=audit.azn:file` implies `log_id=audit`.

6. Specify the maximum size of the log file:

```
rollover_size= value
```

By default is `rollover_size=2000000`.

The rollover size values are interpreted as:

- If less than zero, a new log file is created with each invocation of the process and every 24 hours from that instance.
- If equal to zero, no rollover is completed, and the log file grows indefinitely. If a log file exists, new data is appended to it.
- If greater than zero, a rollover is completed when a log file reaches the configured threshold value. If a log file exists at startup, new data is appended to it.

7. Specify the maximum number of rollover log files:

```
max_rollover_files= value
```

The rollover size values are interpreted as:

- If the value is blank or not specified, no rollover files are deleted.
- If equal to zero, only the current log file is kept, and all rollover log files are deleted.
- If greater than zero, only that number of rollover log files are kept. When the number of rollover log files exceeds `max_rollover_files`, the oldest log file is deleted.

8. Specify the maximum size of the buffer:

```
buffer_size={0|number_kb}
```

By default, the buffer size for logging to a file is `0` bytes, This buffer size prevents buffering so that each event is handled individually. If a value other than `0` is specified, events are packed into buffers of that size before queuing to the file log agent.

Buffers consist of only an integral number of events; events are not split across buffers. If any individual event exceeds that maximum configured size, the large event is recorded in a buffer of its own, exceeding the configured value.

9. Specify the maximum number of events to queue in memory:

   ```
   queue_size={0|number_events}
   ```

   By default, the queue size is `0`. A zero queue size means that no limit is enforced on the growth of the unprocessed event queue. The requesting thread is blocked until space is available in the queue when:

   - The `queue_size` is defined as any valid value except `0`.
   - The number of events in the queue reaches the defined queue size.
   - A new event is ready to be placed on the queue.

10. Specify the event queue high water mark:

    ```
    hi_water={0|1|number}
    ```

    By default, the event queue high water mark value is two-thirds of the maximum configured queue size.

    If the maximum queue size is `0`, the high water mark is set to a default of `100`. The transaction rates and the values of these options determine the maximum amount of memory that is consumed by enabling event logging to file.

    If the event queue high water mark is set to `1`, every event queued is relayed to the log agent as soon as possible. This setting is not optimal.

11. Specify the frequency for flushing log file buffers:

    ```
    flush_interval={0|number_seconds}
    ```

12. Specify the file mode:

    ```
    mode={text|binary}
    ```

    Binary mode on a Windows operating system writes the log file in an AIX, Linux, or Solaris-compatible format.

    Text mode is deprecated on AIX, Linux, and Solaris operating systems.

13. Save and exit the configuration file.

Example

For example, to configure a file log agent to capture authorization events, the following sample shows the `logcfg` entry:

```
[aznapi-configuration]
logcfg=audit.azn:file path=audit.log,
flush_interval=20,rollover_size=2000000,log_id=audit,queue_size=200,
hi_water=100,buffer_size=2,mode=text
```

Tuning the buffer size with the queue size and the event queue high water mark can improve performance.

Configuring remote log agents

Configure the remote log agent to send events to a remote authorization server for recording. For example:

```
[aznapi-configuration]
logcfg = category:remote buffer_size=size,
     compress={yes|no},error_retry=timeout,path=name,
     flush_interval=number_seconds,rebind_retry=timeout,
```

```
server=hostname,port=number,dn=identity,
queue_size=number,hi_water=number
```

Parameter names can be shortened to any unambiguous name. For example, the `hi_water` parameter can be shortened to `hi`.

Requests to log an event remotely are accepted on a best effort basis only. If the remote authorization server is not available, captured events are cached locally and relayed at a later date, if and when the server becomes available.

Only one remote logging connection is established to a remote authorization server. Consider the case where multiple configuration entries are made to:

- Selectively capture events,
- Capture events at different points of the event pool hierarchy, and
- To the same remote server.

Then, the remote connection is established according to the options of the first remote configuration entry processed. Multiple remote connections can be configured to log to different remote authorization servers.

Events received at the remote authorization server are placed in the event pool of that server. The events are placed in a different location from where they were originally captured on the client system. All events entering a host through the remote logging service are placed in a category constructed in the following manner:

```
remote.client-category-domain.hostname.program
```

Note: The short name version of the host name is shown in some of the examples, however, the fully qualified host name is often required. To obtain system configuration information, you can use the **gethostbyname** command.

To relay events remotely on host `amazon`, you might use this example:

```
[aznapi-configuration] logcfg = audit:remote buffer=2000,compress=y,
error=2,path=remote.cache,rebind=600,server=timelord,port=7136
```

Parameters for remote log agents

You can define the following parameters for remote log agents:

buffer_size

To reduce network traffic, events are buffered into blocks of the nominated size before relaying to the remote server. The `buffer_size` parameter specifies the maximum size message that the local program attempts to construct by combining smaller events into a large buffer. Buffers consist only of an integral number of events; events are not split across buffers. If any individual event exceeds that maximum configured size, the large event is sent in a buffer of its own, exceeding the configured value.

The default value is `1024` bytes.

compress

Security Verify Access events are principally text messages. To reduce network traffic, use the `compress` parameter to compress buffers before transmission and expand on reception.

The default value is `no`.

dn

To establish mutual authentication of the remote server, a distinguished name (DN) must be configured. The DN can be checked against the name that is returned in the remote server's certificate.

The default value is a null string. Explicitly specifying an empty string or using the default value enables the logging client to request a remote server connection with any server that is listening.

Specifying a value for the dn parameter limits successful connection to a specific server, such as:

```
dn="cn=ivacld/timelord.testnet.tivoli.com,o=policy director,c=us"
```

A distinguished name must be specified as a string that is enclosed by double quotation marks.

error

If a send to a remote service fails, the system tries again. Before the system tries again, the system waits for the error retry timeout in seconds. If the attempt to try again fails:

- The link is recorded.
- The given event and future events are saved.

Events are saved in the local event cache file until the remote service is available again.

The default value is 2 seconds.

flush_interval

Events can sit in memory for a long time if:

- Events are being consolidated into large buffers.
- There is less logging activity.

Further, events can sit in memory before being:

- Forwarded to the remote server.
- Written to the cache file.

The `flush_interval` parameter limits the time that a process waits to fill a consolidation buffer.

The default value is `20` seconds. A flush interval of `0` is not allowed. Specifying a value of `0` results in the buffer being flushed every 600 seconds.

hi_water

The `hi_water` parameter for a remote logging connection is like the one specified for logging to a file.

path

Configure the `path` parameter to specify the location of a cache file on the local host. The cache file name defaults to `./`*`server`*`.cache`, where *server* is the name of the remote server that is being logged to.

If the running process cannot establish communication with the remote server, or the link fails during operation, event recording switches to storing events in the specified file. The switch lasts until the server becomes available again. When the server is available, events are drained from the disk cache and relayed to the remote server.

For example, suppose that the path value is as follows:

```
path=pdmgrd_remote.cache
```

The log file is created if it does not exist. The size of this file is not bound, and it does not have any rollover capability. If a remote server is not accessible for sufficient time, you might run out of disk space.

port

Configure the `port` parameter to specify the port that the remote authorization server listens on for remote logging requests.

The default value is port 7136.

queue_size

The `queue_size` parameter for a remote logging connection is like the one specified for logging to a file.

rebind_retry

If the remote authorization server is unavailable, the log agent attempts to rebind to this server at this frequency in number of seconds.

```
rebind_retry=number_seconds
```

The default rebind retry timeout value is `300` seconds.

server

The remote logging services are offered by the authorization service. The `server` parameter nominates the hosts to which the authorization server process is bound for event recording.

```
server=hostname
```

Sending events to a remote authorization server

You might configure Security Verify Access to send event records to a remote authorization server.

Before you begin

Before you begin this task, review the information in "Configuring remote log agents" on page 20.

Procedure

1. Edit the appropriate server configuration file. Each server provides its own stanza entry values in its configuration file.
2. Locate the stanza that contains the `logcfg` entries.
3. Specify that the category is to send event records to a remote server using the format *`category`*`:remote`.

   For example, a category might be to audit authorization events (`audit`) :

   ```
   logcfg=audit:remote
   ```

4. Specify the maximum buffer size. This buffer size is the maximum size message that the local program attempts to construct by combining smaller events into a large buffer:

   ```
   buffer_size={0|number_bytes}
   ```

   If a *number_bytes* value is specified, events are packed into buffers of that size before being relayed to the remote server. By default, the buffer size before relaying to the remote server is `1024` bytes.

   Buffers consist of only an integral number of events; events are not split across buffers. If any individual event exceeds that maximum configured size, the large event is recorded in a buffer of its own, exceeding the configured value.
5. Specify the frequency for flushing log file buffers:

   ```
   flush_interval={0|number_seconds}
   ```

   The `flush_interval` parameter limits the time that a process waits to fill a consolidation buffer.

   By default, the flush interval value is `20` seconds. A flush interval of `0` is not allowed. Specifying a value of `0` results in the buffer being flushed every 600 seconds.
6. Specify the maximum number of events to queue:

   ```
   queue_size={0|number_events}
   ```

By default, the queue size is 0. A zero queue size means that no limit is enforced on the growth of the unprocessed event queue. The requesting thread is blocked until space is available in the queue if:

- The maximum value for *number_events* is specified.
- The maximum value for *number_events* is reached.
- A new event is ready to be placed on the queue.

7. Specify the event queue high water mark:

```
hi_water={0|1|number}
```

By default, the event queue high water mark value is a *number* that represents two-thirds of the maximum configured queue size.

If the maximum queue size is 0, the high water mark is set to a default of 100. The transaction rates and the values of these options determine the maximum amount of memory that is consumed by enabling event logging to file.

If the event queue high water mark is set to 1, every event queued is relayed to the log agent as soon as possible. This setting is not optimal.

8. Specify whether you want to compress buffers before transmission and expand on reception:

```
compress={yes|no}
```

By default, the compress value is no to disable.

9. Specify the time to wait whenever a send to a remote service fails and an error occurs:

```
error=seconds
```

By default, the error retry timeout is 2 seconds.

10. Specify the cache file location:

```
path=fully_qualified_path
```

The file name is *server_name*_remote.cache. For example: pdmgrd_remote.cache

The default file name depends on the type of logging being performed, such as audit.log

11. Specify the time between attempts to rebind (sign on):

```
rebind_retry=number_seconds
```

By default, the rebind retry timeout value is 300 seconds.

12. Specify the host name of the remote authorization server:

```
server=hostname
```

13. Specify the remote server port number:

```
port=authorization server port
```

By default, the port number value is 7136.

14. Specify the remote server distinguished name to establish mutual authentication of the remote server:

```
dn="distinguished_name"
```

The default value for the dn parameter is a null string. Explicitly specifying an empty string or using the default value enables the logging client to request a remote server connection with any server listening.

The dn parameter value limits a successful connection to a specific server, for example:

```
dn="cn=ivacld/timelord.tivoli.com,o=policy director,c=us"
```

A distinguished name must be specified as a string enclosed by double quotation marks.

15. Save and exit the configuration file.

Example

This example sends event records to the remote `timelord` server:

```
[aznapi-configuration] logcfg = audit:remote buffer=2000,compress=y,error=2
path=remote.cache,rebind=600,server=timelord,port=7136
dn="cn=ivacld/timelord.tivoli.com,o=policy director,c=us"
```

Configuring remote syslog agents

Use the `logcfg` entry to configure the remote syslog agent to send events to a remote syslog server for recording.

For example:

```
[aznapi-configuration]
logcfg = category:rsyslog,error_retry=timeout,log_id=id,
                    path=name,flush_interval=number_seconds,max_event_len=length,
                    rebind_retry=timeout,server=hostname,port=number,
                    ssl_keyfile=key_file,ssl_label=label,ssl_stashfile=stash_file,
                    queue_size=number,hi_water=number
```

The agent accepts requests to log an event remotely on a best effort basis only. If the remote syslog server is not available, the agent buffers events in a local cache file. When the server becomes available again, the agent sends the events to the server.

Caching does not occur if you configure the agent to use clear text communication with the syslog server. Clear text communication occurs over the User Datagram Protocol (UDP), which does not guarantee message delivery. In this configuration, the network layer does not notify the agent if the server does not receive the event. This means that events can be lost if the remote syslog server becomes unavailable.

Note: If you do not want to use clear text communication, you can configure SSL. For SSL communication, the agent uses the TLS Cipher Suite to encrypt the data.

Parameters for remote syslog agents

You can define the following parameters for remote syslog agents:

error_retry

If a message sent to a remote syslog service fails, the system tries again. Before trying again, the system waits for the **`error_retry`** timeout in seconds. If the next attempt fails, the agent saves the current event and future events in the local cache file until the remote service is available again.

The default value is 2 seconds.

flush_interval

Events can sit in memory for a long time if there is only a small amount of logging activity.

The **`flush_interval`** parameter limits the time a process waits to fill a consolidation buffer.

The default value is `20` seconds. You cannot use a flush interval of `0` seconds. If you specify a value of `0`, the agent flushes the buffer every 600 seconds.

hi_water

Processing of the event queue is scheduled regularly at the configured flush interval. It is also triggered asynchronously when the queue size reaches a high water mark on the event queue.

Use the **hi_water** parameter to define this high water mark. The default value is two-thirds of the maximum configured queue size. If the maximum queue size is zero, the high water mark is set to a default of 100.

The transaction rates and the values of these options determine the maximum amount of memory that the agent uses for logging events to file.

If the event queue high water mark is set to 1, WebSEAL relays every queued event to the log agent as soon as possible. This setting is not optimal. A setting of 1 ensures that events get to disk as fast as possible, but this configuration adversely impacts overall performance.

log_id

The **log_id** parameter defines the name of the application that the syslog agent includes in the messages sent to the remote syslog server. This field is mandatory.

max_event_len

The **max_event_len** parameter specifies the maximum length of an event that the syslog agent transmits to the remote syslog server.

If the event text is longer than the configured length, the agent truncates the message to the maximum event length. If the maximum event length is zero, the agent does not truncate the event text.

If you are using clear text communication to transmit the event, set the **max_event_len** parameter to a value less than the maximum transmission unit (MTU). That is, use a value less than the MTU for the network path to the server to avoid fragmentation of the event.

port

Configure the **port** parameter to specify the port that the remote syslog server listens on for remote logging requests.

The default port value is 514 for clear text communication and 6514 for SSL communication.

queue_size

There is a delay between placing events on the queue and their removal by the file log agent. The **queue_size** parameter specifies the maximum size of the queue. Consider that a new event is ready to be placed on the queue. If the queue reaches the maximum size, the requesting thread is blocked until space is available in the queue.

This process causes the performance of the event propagation thread to slow down to the speed of the file logging thread.

You must use the **queue_size** parameter to limit the central event propagation queue size. If not, memory usage by the log agent can grow without bounds.

```
[aznapi-configuration]
logcfg = audit.azn:rsyslog
...
queue_size=number_events,
...
```

The default value is 0. Specifying a value of 0 indicates that there is no limit to the growth of the unprocessed event queue. In this case, the speed of the logging thread does not constrain the event propagation thread. The unrecorded event queue can grow to an unmanageable size if:

- You are using the default value.
- Events are being generated faster than they can be recorded to file.

rebind_retry

If the remote syslog server is unavailable, the log agent attempts to rebind to this server at this frequency in number of seconds.

```
rebind_retry=number_seconds
```

The default **rebind_retry** timeout value is 300 seconds.

server

The remote logging services are offered by the remote syslog server. The **server** parameter nominates the host to which the agent is bound for event recording.

```
server=hostname
```

ssl_keyfile

The name of the GSKit key database file that contains the CA certificate. The logging agent uses the CA certificate to establish a secure connection with the remote syslog server over SSL.

The path of this file is relative to the config file. You do not need to manually specify a path.

If you do not configure this value, the logging agent uses clear text that is not encrypted to communicate with the remote syslog server.

ssl_label

The name of the certificate that the logging agent presents to the remote syslog server to establish a secure connection.

If you do not configure this field, the agent uses the default certificate from the key database.

ssl_stashfile

The name of the GSKit stash file that contains the password for the ssl-keyfile database. This field is mandatory if you specify a value for the **ssl-keyfile** field.

The path of this file isp relative to the config file. You do not need to manually specify a path.

ssl_protocols

A colon separated list of SSL protocols to be enabled. Valid protocols include: **sslv3**, **tlsv10**, **tlsv11**, and **tlsv12**.

Note: This entry will be ignored if the NSA suite-b SSL compliance support has been enabled.

severity
An integer in the range 0 to 7 inclusive as defined in RFC 5424, The Syslog Protocol.

facility
An integer in the range 0 to 23 inclusive as defined in RFC 5424, The Syslog Protocol.

Sending events to a remote syslog server

You can configure Security Verify Access to send event records to a remote syslog server.

Before you begin

Before you begin this task, review the information in “Configuring remote syslog agents” on page 25.

Procedure

1. Edit the appropriate server configuration file. Each server provides its own stanza entry values in its configuration file.
2. Locate the stanza that contains the `logcfg` entries.
3. Specify that the category is to send event records to a remote server by using the format *`category`*`:rsyslog`.

   For example, a category that audits authorization events (`audit`):

   ```
   logcfg=audit:rsyslog
   ```
4. Specify the frequency for flushing log file buffers:

```
flush_interval={0|number_seconds}
```

The `flush_interval` parameter limits the time a process waits to fill a consolidation buffer.

By default, the flush interval value is `20` seconds. You cannot use a flush interval of `0` seconds. If you specify a value of `0`, the agent flushes the buffer every 600 seconds.

5. Specify the maximum number of events to queue:

```
queue_size={0|number_events}
```

By default, the queue size is `0`. A zero queue size means that the agent does not limit the growth of the unprocessed event queue. The requesting thread is blocked until space is available in the queue if:

- The maximum value for *number_events* is specified.
- The maximum value for *number_events* is reached.
- A new event is ready to be placed on the queue.

6. Specify the event queue high water mark:

```
hi_water={0|1|number}
```

By default, the event queue high water mark value is a *number* that represents two-thirds of the maximum configured queue size.

If the maximum queue size is `0`, the high water mark is set to a default of `100`. The transaction rates and the values of these options determine the maximum amount of memory that the agent uses for logging events to file.

If the event queue high water mark is set to 1, WebSEAL relays every queued event to the log agent as soon as possible. This setting is not optimal.

7. Specify the time to wait whenever a send to a remote service fails and an error occurs:

```
error_retry=seconds
```

By default, the **`error_retry`** timeout is 2 seconds.

8. Specify the cache file location:

```
path=fully_qualified_path
```

The default cache file name is `./log_id.cache`. For example: `rsyslog.cache`

Note: The directory portion of this path must exist. If the log file does not exist, the agent creates the file.

9. Specify the time between attempts to rebind (sign on):

```
rebind_retry=number_seconds
```

By default, the **`rebind_retry`** timeout value is `300` seconds.

10. Specify the host name of the remote syslog server:

```
server=hostname
```

11. Specify the remote server port number:

```
port=rsyslog_port
```

The default port number is 514 for clear text communication and 6514 for SSL communication.

12. Specify the application name that the syslog agent includes in the messages sent to the remote server:

```
log_id=name
```

13. Specify the maximum length of an event that the agent transmits to the remote syslog server. If the event text is longer than this configured value, the agent truncates the message to the maximum event length. If the maximum event length is `0`, the agent does not truncate the event text.

    ```
    max_event_len=length
    ```

    Note: If you are using clear text communication to transmit the event, set the **`max_event_len`** parameter to a value less than the maximum transmission unit (MTU). Use a value less than the MTU for the network path to the server to avoid fragmentation of the event.
14. Optional: If you require SSL communication with the remote server, you must specify the SSL keyfile:

    ```
    ssl_keyfile=key_file
    ```

15. Optional: If you are using SSL communication, you can use **`ssl_label`** to specify the certificate name:

    Note: If you do not configure a value for this field, the agent uses the default certificate from the key database.

    ```
    ssl_label=my_label
    ```

16. Optional: If you require SSL communication with the remote server, you must specify the SSL stash file:

    ```
    ssl_stashfile=stash_file
    ```

Example

This example sends event records to the remote `timelord` server:

```
[aznapi-configuration]
logcfg = audit:rsyslog error_retry=2,path=rsyslog.cache,
rebind_retry=600,server=timelord,port=514,log_id=webseal-instance
```

Disabling resource access events

You can use protected object policies (POPs) to selectively disable auditing of access to particular resources.

Procedure

- Disable generating audit records.

  If a POP with the `auditthttp` extended attribute set to no is attached to a resource, access to that resource does not generate an HTTP access audit record. For example, if access to the `/images` subdirectory is not of sufficient interest to merit an audit record, you can disable audit records by using the following commands:

  ```
  pdadmin sec_master> pop create nohttpaudit
  pdadmin sec_master> pop modify nohttpaudit set attribute auditthttp no
  pdadmin sec_master> pop attached /WebSEAL/server/images nohttpaudit
  ```

  After you attach the `nohttpaudit` POP to the `/images` subdirectory, access to files under this directory no longer generates an audit event.
- Enable generating audit records.

  If you have a specific resource that must be audited, you can enable auditing of that resource. To enable auditing, attach a second POP *without* the `auditthttp` attribute. For example, the

special.jpg file in the /images subdirectory must be audited. You can enable audit records for the file with the following commands:

```
pdadmin sec_master> pop create restorehttpaudit
pdadmin sec_master> pop attached /WebSEAL/server/images/special.jpg \
  restorehttpaudit
```

Process flow for logcfg logging

The following example process flow assumes the [aznapi-configuration] stanza of a WebSEAL configuration file.

Use the syntax of the logcfg entry to specify a log file. The log file is opened at WebSEAL initialization. If no log file is opened during initialization, regardless of other configuration settings, no events are logged. Unless a log file is specified, all event data is lost.

```
[aznapi-configuration]
logcfg = http.agent:file path=abc.log,log_id=agent
```

You can use the log_id identifier to facilitate the recording of events from different categories to the same file. You can construct more log agents. The log agents can gather different event data. These agents use log_id to direct the data to the log file that was opened by the initial log agent. The first logcfg entry must be used to define the log agent. If the log agent is defined after the first log_id, no events for that category are logged.

In the following example, events from the http.agent category are directed to the abc.log file. The log agent has the log_id=httplogs identifier. Events from http.ref and http.clf audit categories are also logged to this file because the logcfg entry uses the same identifier log_id=httplogs:

```
[aznapi-configuration]
logcfg = http.agent:file path=abc.log,log_id=httplogs
logcfg = http.ref:file log_id=httplogs
logcfg = http.clf:file log_id=httplogs
```

Auditing using logaudit

WebSEAL and Plug-in for Web Servers continue to support audit logging that uses the logaudit entries and its related entries in the [aznapi-configuration] stanza. This approach uses the following stanza entries:

```
[aznapi-configuration]
logaudit
auditlog
auditcfg
logsize
logflush
```

This approach is comparable to the logcfg entry with a file agent.

For example, to capture authentication events, you can set the configuration file entries as follows:

```
[aznapi-configuration]
logaudit = yes
auditcfg = authn
auditlog = /var/pdweb/log/audit.log
logsize = 2000000
logflush = 20
```

If you are still using the logaudit approach, consider using the logcfg approach. The logcfg approach provides more configuration options, such as buffer size and event queues, and the ability to use the console, pipe, and remote log agents.

WebSEAL HTTP logging

This chapter describes WebSEAL HTTP logging.

HTTP log files

WebSEAL maintains the following HTTP log files that record HTTP activity:

- `request.log`
- `agent.log`
- `referer.log`

Stanza entries for configuring traditional HTTP logging are in the `[logging]` stanza of the WebSEAL configuration file.

Table 5 on page 31 illustrates the relationship among the HTTP logs and the configuration file entries:

Table 5. Relationship between HTTP logs and the stanza entries

File name	Log file entry	Enablement entry
request.log	`requests-file`	`requests`
referer.log	`referers-file`	`referers`
agent.log	`agents-file`	`agents`

Enabling HTTP logging

By default, HTTP logging is enabled in the WebSEAL configuration file. For example:

```
[logging]
requests = yes
referers = yes
agents = yes
```

You can enable or disable each log independently from the others. If any stanza entry is set to no, logging is disabled for that file.

Configuring HTTP logging in the `[logging]` stanza implements the standard event logging mechanism that is described in "Audit event logging" on page 11.

The following configurations are created when the WebSEAL HTTP logging stanza entries are enabled. These configurations accept the values of the `requests-file`, `referers-file`, `agents-file`, `flush-time`, and `max-size` stanza entries from the WebSEAL configuration file `[logging]` stanza:

request.log

```
logcfg = http.clf:file path=requests-file,flush=flush-time,
rollover=max-size,max_rollover_files=max-files,
log=clf,buffer_size=8192,queue_size=48
```

referer.log

```
logcfg = http.ref:file path=referers-file,flush=flush-time,
rollover=max-size,max_rollover_files=max-files,
log=ref,buffer_size=8192,queue_size=48
```

agent.log (common log format)

```
logcfg = http.agent:file path=agents-file,flush=flush-time,
rollover=max-size,max_rollover_files=max-files,
log=agent,buffer_size=8192,queue_size=48
```

See “Process flow for logcfg logging” on page 30 for special considerations and conditions when you use both traditional HTTP logging (`[logging]` stanza) and the event logging mechanism (`[aznapi-configuration]` stanza).

Specifying the timestamp

You can choose to have timestamps in each HTTP log file that is recorded in Greenwich Mean Time (GMT). This GMT choice overrides the local time zone. By default, the local time zone is used.

To use GMT timestamps, set the value of the `gmt-time` entry to `yes` as shown in the following entry:

```
gmt-time = yes
```

Specifying rollover thresholds

The `max-size` stanza entry specifies the maximum size to which each of the HTTP log files can grow and has the following default value in bytes:

```
[logging]
max-size = 2000000
```

When a log file reaches its rollover threshold:

- The existing file is backed up to a file of the same name. The file name is appended with the current date and timestamp.
- A new log file is started.

The various possible `max-size` values are interpreted as follows:

- If the `max-size` value is less than zero (< 0), a new log file is created:
  - With each invocation of the logging process.
  - Every 24 hours from that instance.
- If the `max-size` value is equal to zero (= 0), no rollover is completed and the log file grows indefinitely. If a log file exists, new data is appended to it.
- If the `max-size` value is greater than zero (> 0), a rollover is completed when a log file reaches the configured threshold value. If a log file exists at startup, new data is appended to it.

Specifying the frequency for flushing buffers

Log files are written to buffered data streams. If you are monitoring the log files in real time, alter the frequency with which the server flushes the log file buffers.

By default, log files are flushed every 20 seconds as shown in the following example:

```
[logging]
flush-time = 20
```

If you specify a negative value, a flush is forced after each record is written.

Distinguishing virtual hosts

When you use virtual hosts, you can use the `request-log-format` entry in the `[logging]` stanza to distinguish between requests to different virtual hosts.

Use the **%v** directive at the start of the `request-log-format` configuration item to include the header at the front of each line in the request log.

When you use the **%R** directive entry in the `request-log-format` configuration item, the log contains the absolute URI.

Customizing the HTTP request log

You can customize the content of the `request.log` file by adding a configuration entry in the `[logging]` stanza. The syntax is as follows:

```
request-log-format=directives
```

The following directives can be used to customize the log format:

Table 6. Directives for customizing the format of the request.log file

Directive	Description
%a	Remote IP address
%A	Local IP address
%b	Bytes in the response excluding HTTP headers in CLF format: '-' instead of 0 when no bytes are returned
%B	Bytes in the response excluding HTTP headers
%c	The HTTP response status received from the junctioned server.
%{Attribute}C	Attribute from the Security Verify Access credential named 'Attribute'
%d	Transaction identifier, or session sequence number
%{cookie-name}e	Contents of the cookie 'cookie-name' in the request
%{cookie-name}E	Contents of the cookie 'cookie-name' in the response
%F	Time that it takes to serve the request in microseconds
%h	Remote host
%H	Request protocol
%{header-name}i	Contents of the Header 'header-name' in the request
%j	The name of the junction that services the request
%J	The length of time, in microseconds, that the junction server spent processing the request. This includes the time that it took to send the request to the server, the length of time that it took the server to process the request, and the length of time that it took to read and process the response header.
%l	Remote logname
%m	Request method (that is, GET, POST, HEAD)
%M	The time, in Common Log Format, at which the request was received with millisecond precision.
%{header-name}o	Contents of the Header 'header-name' in the response
%p	Port over which the request was received
%q	The query string (prefixed with '?' or empty)
%Q	Raw query strings that must be decoded manually.
%r	First line of the request

Table 6. Directives for customizing the format of the request.log file (continued)	
Directive	**Description**
%R	First line of the request including HTTP://HOSTNAME
%s	Response status
%S	The hostname of the junctioned server which serviced this request.
%t	Time and date in CLF format
%{format}t	The time and date in the specified format
%T	Time that it takes to serve the request in seconds
%u	Remote user
%U	The URL requested
%v	Canonical ServerName of the server that serves the request
%z	The decoded path string.
%Z	The raw path string.

The following configuration entry shows an example of customizing the `request.log` file:

```
request-log-format = %h %l %u %t "%r" %s %b
```

Customized HTTP logs also support the new line (\n), carriage return (\r), and tab (\t) special characters. Any character that is either not part of a directive or not a special character is written out in the log entry. You can direct the system to ignore the % and \ characters by prefixing them with the backslash (\) character. For example:

```
request-log-format = \%{header}i\t->\t%{header}i
```

renders the following output:

```
%{header}i    ->    header
```

Process flow for [logging] and logcfg logging

You can configure WebSEAL auditing you use both the `[logging]` stanza and the `[aznapi-configuration]` stanza.

When you use both configuration settings, WebSEAL processes the `[aznapi-configuration]` stanza before the `[logging]` stanza.

For example, assuming the following entries in the WebSEAL configuration file:

```
[logging]
requests = yes
requests-file =request.log

[aznapi-configuration]
logcfg = stats.pdweb.authn:file path=stats.log,log_id=stats
logcfg = http.agent:file path=abc.log,log_id=httplogs
logcfg = http.ref:file log_id=httplogs
```

WebSEAL processes these entries in the following manner:

1. The `[aznapi-configuration]` stanza is read.
2. The `stats.log` file with `log_id=stats` is opened. All `stats.pdweb.authn` events are logged to this file.
3. The `abc.log` file with `log_id=httplogs` is opened. All `http.agent` events are logged to this file.

4. Because the next log agent uses `log_id=httplogs`, all `http.ref` events are logged to the previously opened `abc.log` file.
5. The `[logging]` stanza is read.
6. HTTP request logging is enabled. All `http.clf` events are logged to the `request.log` file that uses the default `log_id=clf`. See the following example for an explanation of this default identifier.

HTTP logging using the `[logging]` stanza operates by generating its own default log agent entries. Each HTTP log file has a default value for the `log_id` parameter.

Log file	log_id
request.log	`log_id=clf`
referer.log	`log_id=ref`
agent.log	`log_id=agent`

If a `logcfg` entry in the `[aznapi-configuration]` stanza contains the same `log_id` as one used in the `[logging]` stanza, the HTTP log file is not created. Audit events with the same `log_id` are directed to 1 log file only. That 1 log file is always the first one opened.

In the following example, the `abc.log` file with `log_id=clf` is opened first. Because the HTTP requests logging defined in the `[logging]` stanza uses a default `log_id=clf`, the `requests.log` file is never created and all `http.clf` (`requests`) events are directed to `abc.log` file.

```
[logging]
requests = yes
requests-file = request.log

[aznapi-configuration]
logcfg = http.agent:file path=abc.log,log_id=clf
logcfg = http.ref:file log_id=clf
```

HTTP logging can be configured in the `[logging]` and `[aznapi-configuration]` stanzas. Therefore, it is possible to have duplicate entries for HTTP events in a log file when both mechanisms are enabled.

In the following example, `http.clf` audit events are recorded twice in the `abc.log` file:

- From the event logging configuration.
- From the enabled request logging, which uses `log_id=clf` by default. The `requests.log` is not created because the `abc.log` file with `log_id=clf` is opened first.

```
[logging]
requests = yes
requests-file =request.log

[aznapi-configuration]
logcfg = http.agent:file path=abc.log,log_id=clf
logcfg = http.ref:file log_id=clf
logcfg = http.clf:file log_id=clf
```

Sample request.log file

The content of the `request.log` file is set by the `request-log-format` configuration item. The following table shows all the possible initial `request-log-format` combinations that are based on the existing `absolute-uri-in-request-log` and `host-header-in-request-log` configuration items:

Table 7. Example output of the request.log file			
absolute-uri-in-request-log	**host-header-in-request-log**	**request-log-format**	**Example output**
No	No	%h %l %u %t "%r" %s %b	10.251.173.1 - sec_master [04/Jan/2009:11:13:07 +1000] "GET /pics/iv30.gif HTTP/1.1" 200 46498
No	Yes	%v %h %l %u %t "%r" %s %b	tamtestbed 10.251.173.1 - sec_master [04/Jan/2009:11:10:04 +1000] "GET /pics/iv30.gif HTTP/1.1" 200 46498
Yes	No	%h %l %u %t "%R" %s %b	10.251.173.1 - sec_master [04/Jan/2009:11:14:51 +1000] "GET HTTP://tamtestbed/pics/iv30.gif HTTP/1.1" 200 46498
Yes	Yes	%v %h %l %u %t "%R" %s %b	tamtestbed 10.251.173.1 - sec_master [04/Jan/2009:11:16:40 +1000] "GET HTTP://tamtestbed/pics/iv30.gif HTTP/1.1" 200 46498

Sample agent.log file

The `agent.log` file records the contents of the `User_Agent:` header in the HTTP request.

This log reveals information about the client browser, such as architecture or version number, for each request. The following example shows a sample version of the `agent.log` file:

```
Mozilla/4.01 [en] (WinNT; U)
Mozilla/4.01 [en] (WinNT; U)
Mozilla/4.01 [en] (WinNT; U)
Mozilla/4.01 [en] (WinNT; U)
```

Sample referer.log

The `referer.log` records the `Referer:` header of the HTTP request. For each request, the log records the document that contained the link to the requested document.

The log uses the following format:

```
referer -> object
```

This information is useful for tracking external links to documents in your web space. The log reveals that the source indicated by *referer* contains a link to a page (*object*). With this log, you can track stale links and to find out who is creating links to your documents.

The following example shows a sample version of a `referer` log file:

```
http://manuel/maybam/index.html -> /pics/tivoli_logo.gif
http://manuel/maybam/pddl/index.html -> /pics/tivoli_logo.gif
http://manuel/maybam/ -> /pddl/index.html
http://manuel/maybam/ -> /pddl/index.html
http://manuel/maybam/pddl/index.html -> /pics/tivoli_logo.gif
http://manuel/maybam/ -> /pddl/index.html
```

Working with local statistics

This chapter provides information about working with the Security Verify Access modules that can monitor and collect statistical information.

Using stats commands for statistics

Use the **server tasks stats** command that is provided as by the **pdadmin** utility to manage statistics components. You can use the **stats** command to complete the following operations:

stats on
Enable statistics for a specific component.

stats off
Disable statistics for a specific component or for all components.

stats show
List enabled components.

stats get
Display current statistics values for a specific component or for all components.

stats reset
Reset statistics values for a specific component or for all components.

stats list
List all statistics components.

See "server task stats" on page 202 for more information about the command.

Enabling statistics

You can enable statistics reporting with the **stats on** command or with stanza entries in the configuration file for the specific server.

For details about using stanza entries to enable statistics, see "Using stanza entries for statistics" on page 41.

To enable the gathering of statistics with the **stats on** command, set the statistics report frequency, event count, and destination for the component. For more information about the **stats on** command, see "server task stats" on page 202.

Note:

- By default, the WebSEAL pdweb.threads, pdweb.doccache, and pdweb.jmt components are always enabled and cannot be disabled.
- Using **stats on** and changing the runtime Policy server trace settings affects only the current run of the Policy server. If the Policy server is stopped and then started later, the default trace settings take effect. To persist trace settings across multiple runs of the Policy server, modify the `/etc/pdmgrd_routing` file.

When you enable statistics, you can specify one log file for the statistics report. If you specify two equivalent commands that differ only on the destination, the second invocation deactivates the first log file and activates the second log file. The following example illustrates this limitation:

```
#pdadmin> server task default-webseald-abc.ibm.com stats on pdweb.http 20 \
  file path=A.log
```

```
#pdadmin> server task default-webseald-abc.ibm.com stats on pdweb.http 20 \
  file path=B.log
```

The first command enables the `pdweb.http` component and sends statistics reports to the `A.log` file. The second command attempts to activate a second log file, `B.log`. However, this action actually deactivates the `A.log` file while it also activates the `B.log` file.

Enabling basic statistics

To enable basic statistics gathering, use the **stats on** command and specify only the *component* option. Because the *interval* option is not specified, you can obtain statistics information only for this component with the **stats get** command. Because the *destination* option is not specified, the information is sent to the standard log file for that component.

The following example enables the gathering of statistics for the pdweb.http component of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats on pdweb.http
```

Enabling statistics with frequency and count

To enable the gathering of statistics at a designated frequency and event count, use the **stats on** command and specify the following options:

- *component*
- *interval*
- *count*

The *interval* and *count* options:

- Cause the buffer to accumulate a specific number of entries that represent a statistics report.
- Flush the buffer after a specific number of seconds elapse.

Because the *destination* option is not specified, the information is sent to the standard log file for that component.

The following example enables the gathering of statistics for the pdweb.http component of a WebSEAL instance. In this example, the buffer accumulates 100 entries and sends statistics reports every 20 seconds:

```
#pdadmin> server task default-webseald-abc.ibm.com stats on pdweb.http 20 100
```

Enabling statistics with frequency and destination

To enable gathering of statistics at a designated frequency and write the statistics to a specific file, use the **stats on** command and specify the following options:

- *component*
- *interval*
- *destination*

The *interval* option, without a *count* option, indefinitely sends statistics reports after a specific number of seconds elapses. The *destination* option specifies the exact file where the statistics are written. When you specify a file that is different for the file log agent for the component, you can specify more configuration options.

The following example enables the gathering of statistics for the pdweb.http component of a WebSEAL instance where:

- A statistics report is sent to the `jmt-stats.log` file every 20 seconds.
- A new file is created each time that the buffer is flushed.

```
#pdadmin> server task default-webseald-abc.ibm.com stats on pdweb.http 20 \
  file path=jmt-stats.log,rollover_size=-1
```

The growth of the log file is controlled by the `rollover_size` configuration option. For complete details about configuring event logging, see the Troubleshooting topics in the Knowledge Center.

Disabling statistics

You can disable statistics reporting with the **stats off** command for a specific component or for all components. By default, the pdweb.threads, pdweb.doccache, and pdweb.jmt components are always enabled and cannot be disabled.

Disabling statistics for all components

To disable the gathering of statistics for all components, use the **stats off** command without options.

The following example disables statistics for all components of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats off
```

Disabling statistics for a single component

To disable the gathering of statistics for a single component, use the **stats off** command with the *component* option.

The following example disables statistics for the pdweb.sescache component of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats off pdweb.sescache
```

Listing enabled components

You can use the **stats show** command to:

- List all enabled statistics components.
- Determine whether a specific component is enabled.

Listing all enabled components

To display a list of all components, use the **stats show** command without options.

The following example displays a list of the enabled component of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats show

pdweb.authn
pdweb.doccache
pdweb.jmt
pdweb.sescache
pdweb.threads
```

Because the pdweb.threads, pdweb.doccache, and pdweb.jmt components are always enabled, the output for a WebSEAL instance always contains these entries.

Determining whether a component is enabled

To determine whether a component is enabled, use the **stats show** command with the *component* option.

If the component is enabled, the output lists that component. If the component is not enabled, no output is displayed.

Displaying statistics

You can display the current statistics for all enabled components or for a single component with the **stats get** command.

Displaying statistics for all components

To display statistics for all components, use the **stats get** command without options. For each enabled component, the name of the component is displayed followed by its statistics. For details about the

specifics of the statistics for each component, see the information for that specific component in one of the following sections:

- "Security Verify Access components and activity types" on page 42
- "WebSEAL components and activity types" on page 43

The following example displays the current statistics for all enabled components of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats get

pd.ras.stats.monitor
  ...
pd.log.EventPool.queue
  ...
pd.log.file.clf
  ...
pd.log.file.ref
  ...
pd.log.file.agent
  ...
pdweb.authn
  ...
pdweb.authz
  ...
pdweb.http
  ...
pdweb.https
  ...
pdweb.threads
  ...
pdweb.sescache
  ...
pdweb.doccache
  ...
pdweb.jct.1
  ...
pdweb.jct.2
...
```

Displaying statistics for a single component

To display statistics for a single component, use the **stats get** command with the *component* option.

The following example displays the current statistics for the pdweb.threads component of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats get pdweb.threads

active : 4
total : 50
'default' active : 4
'default' total : 50
```

Resetting statistics

You can reset the current statistics for all enabled components or for a single component with the **stats reset** command.

To reset statistics for all components, use the **stats reset** command without options.

To reset statistics for a single component, use the **stats reset** command with the *component* option.

Listing components

You can list all components that are available to gather and report statistics with the **stats list** command.

To determine which queues are implemented on a server, use the **stats list** command. The following example lists all available components of a WebSEAL instance:

```
#pdadmin> server task default-webseald-abc.ibm.com stats list

pd.ras.stats.monitor
pd.log.EventPool.queue
pd.log.file.clf
pd.log.file.ref
pd.log.file.agent
pdweb.authn
pdweb.authz
pdweb.http
pdweb.https
pdweb.threads
pdweb.jmt
pdweb.sescache
pdweb.doccache
pdweb.jct.1
```

Using stanza entries for statistics

The configuration file for each server contains the following stanza entries that can be set to:

- Enable the statistics interface.
- Specify the destination for statistics reports.
- `stats`
- `logcfg`

The following segment of a configuration file shows the structure of the `stats` and `logcfg` stanza entries:

```
[aznapi-configuration]
stats = component [interval [count]]
logcfg = stats.component:destination
```

For information about the *interval* and *count* options, see "server task stats" on page 202. For complete details about configuring event logging, see the Troubleshooting topics in the Knowledge Center.

Enabling statistics for a single component

In a server configuration file, you can enable gathering of statistics by using the `stats` and `logcfg` entries. These entries are in the `[aznapi-configuration]` stanza.

In the following example:

- The `stats` stanza entry enables gathering of statistics for the `pdweb.jmt` component. The frequency is 20 seconds.
- The `logcfg` stanza entry specifies the destination for the statistics report as the `jmt.log` file. The entry contains more configuration information for the `rollover_size` and `flush` configuration settings:

```
[aznapi-configuration]
stats = pdweb.jmt 20
logcfg = stats.pdweb.jmt:file path=jmt.log,rollover_size=-1,flush=20
```

For detailed information about configuration files, see the Administering topics in the Knowledge Center.

Enabling statistics for multiple components

Unlike the **stats on** command, you enable gathering of statistics for multiple components by using multiple `stats` and `logcfg` entries in the `[aznapi-configuration]` stanza. The stanza is in the server configuration file.

In the following example, statistics gathering is enabled for the following WebSEAL components:

pdweb.authn
For the pdweb.authn component:

- The frequency is set to 40 seconds.
- The destination for the statistics report is the an.log file.

The component has more configuration information for the rollover_size and flush configuration settings.

pdweb.jct.1
For the pdweb.jct.1 component:

- The frequency is set to 50 seconds,
- The destination for the statistics report is the jct.log file.

The component has more configuration information for the rollover_size and flush configuration settings.

pdweb.jmt
For the pdweb.jmt component:

- The frequency is set to 20 seconds.
- The destination for the statistics report is the jmtA.log and the jmtB.log files.

The component has more configuration information for the rollover_size and flush configuration settings.

```
[aznapi-configuration]
stats = pdweb.jmt 20
stats = pdweb.authn 40
stats = pdweb.jct.1 50
logcfg = stats.pdweb.jmt:file path=jmtA.log,rollover_size=-1,flush=20
logcfg = stats.pdweb.jmt:file path=jmtB.log,rollover_size=-1,flush=20
logcfg = stats.pdweb.authn:file path=an.log,rollover_size=-1,flush=20
logcfg = stats.pdweb.jct.1:file path=jct.log,rollover_size=-1,flush=20
```

For detailed information about configuration files, see the Administering topics in the Knowledge Center.

Security Verify Access components and activity types

The following statistics components are available to Security Verify Access servers:

pd.log.EventPool.queue component

The pd.log.EventPool.queue component is the main event propagation queue. Use the statistics interface to monitor:

- The queuing profiles that are configured for the main propagation queue.
- Each file agent.
- Remote agent.
- Pipe log agent.

Each queue that is created as an instance of the EventQueue object registers itself with the statistics subsystem with its category name. The category name is constructed from the logging agent type and the pd.log string.

The following example shows the output from a **stats get** command for the pd.log.EventPool.queue component:

```
#pdadmin> server task ivacld-instance stats get \
  pd.log.EventPool.queue

dispatcher wakes on timeout (20) : 3617
dispatcher wakes by notify : 0
   notifies above highwater (100) : 0
   notifies below highwater : 0
   spurious notifies : 0
```

```
total events processed : 24
average number of events handled per activation : 1
greatest number of events handled per activation : 7
blocks in queue requests : 0
```

In the previous output:

- The flush frequency for the queue is 20, the value that is denoted in the parentheses after `timeout`.
- The high water setting for the queue is 100, the value that is denoted in the parentheses after `highwater`.

The settings that are defined for the various queue configuration options must attempt to balance:

- The maximum amount of memory that is consumed between queue activations, and
- The rate at which a particular log agent can consume events.

Set the queue high water mark such that the number of events that are processed during a queue activation fills a processing time slice. This setting avoids unnecessary thread context-switching. However, setting these options to large values is not productive. The reason is that event log processing must be done at some point and cannot be deferred indefinitely. Consuming large amounts of memory has its own drawbacks.

pd.log.file.agent component

```
dispatcher wakes on timeout (20) : 299
dispatcher wakes by notify : 0
    notifies above highwater (33) : 0
    notifies below highwater : 0
    spurious notifies : 0
total events processed : 146
average number of events handled per activation : 0
greatest number of events handled per activation : 1
blocks in queue requests : 0
```

pd.log.file.clf component

```
dispatcher wakes on timeout (20) : 299
dispatcher wakes by notify : 0
    notifies above highwater (33) : 0
    notifies below highwater : 0
    spurious notifies : 0
total events processed : 147
average number of events handled per activation : 0
greatest number of events handled per activation : 1
blocks in queue requests : 0
```

pd.log.file.ref component

```
dispatcher wakes on timeout (20) : 300
dispatcher wakes by notify : 0
    notifies above highwater (33) : 0
    notifies below highwater : 0
    spurious notifies : 0
total events processed : 148
average number of events handled per activation : 0
greatest number of events handled per activation : 1
blocks in queue requests : 0
```

pd.ras.stats.monitor component

```
5 components reporting statistics
5 reports generated
```

WebSEAL components and activity types

The following statistics components are available to WebSEAL instances:

pdweb.authn component

The pdweb.authn statistics component gathers information about WebSEAL authentication. The following list describes the types of available information:

pass
The total number of successful authentications.

fail
The total number of failed authentications.

pwd exp
The total number of authentication attempts that were made with an expired password.

max
The maximum time for a single authentication process.

avg
The average time for a single authentication process.

total
The total time for all authentication processing.

The following example shows the output from a **stats get** command for the pdweb.authn component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.authn

pass     : 2
fail     : 1
pwd exp  : 0
max      : 0.178
avg      : 0.029
total    : 0.382
```

pdweb.authz component

The pdweb.authz statistics component gathers information about WebSEAL authorization. The following list describes the types of available information:

pass
The total number of successful authorization requests. That is, the total number of resources that were successfully accessed.

fail
The total number of failed authorization requests.

The following example shows the output from a **stats get** command for the pdweb.authz component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.authz

pass     : 2
fail     : 1
```

pdweb.doccache component

The pdweb.doccache statistics component gathers information about WebSEAL document-caching activity. This component reports statistics for all MIME types that aer enabled in the `[content-cache]` stanza of the WebSEAL configuration file. This component is always enabled by default and cannot be disabled.

The following list describes the types of global information available for all MIME types:

General Errors
The number of errors reported by the pdweb.doccache component when there are memory allocation failures, initialization failures, or invalid MIME type header values.

Uncachable
The number of instances when there is no cache that is defined for the MIME type of the document to be cached.

Pending Deletes
The number of entries that are marked for deletion, but these entries are still in use.

Pending Size
The number of bytes that are used by entries that are marked for deletion, but these entries are still in use.

Misses
The number of times a URL is looked up in the document cache and is not found. A found cached document eliminates the need to access the real document again.

Cache MIME type
The MIME type of documents that is stored in this cache. The following list describes the cache MIME types:

Max size
The maximum combined byte size of all documents in the cache.

Max entry size
The maximum byte size for any single cached document. If the document size exceeds this internally calculated value, it is not cached.

Size
The total byte count for all documents currently located in the cache.

Count
The current number of entries in the cache.

Hits
The number of successful lookups. (Documents that are successfully found in the cache.)

Stale hits
The number of successful lookups that found an entry that was too old and was purged instead.

Create waits
The number of times subsequent requests for a document are blocked (made to wait) while the document content is initially being cached.

Cache no room
The number of times a document that is valid for caching cannot fit into the cache. The reason is that there are too many entries that are being created at the same time.

Additions
The number of successful new entries in the cache.

Aborts
The number of times the creation of a new cache entry is canceled. The reason might be a header that indicates the entry must not be cached.

Deletes
The number of cache entries that were deleted because the entry is stale (expired) or because the creation was canceled.

Updates
The number of entries that had expiry times updated.

Too big error
The number of attempts to cache documents that exceed the maximum entry size (and therefore are not cached).

MT errors
The number of times more than one thread tries to create the same entry in the cache. (MT=Multi-Threading)

The following example shows the output from a **`stats get`** command for the pdweb.doccache component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.doccache

General Errors : 0
Uncachable     : 0
```

```
Pending Deletes: 0
Pending Size   : 0
Misses         : 0
Cache MIME type  : text/html
  Max size       : 2048000
  Max entry size : 128000
  Size           : 0
  Count          : 0
  Hits           : 0
  Stale hits     : 0
  Create waits   : 0
  Cache no room  : 0
  Additions      : 0
  Aborts         : 0
  Deletes        : 0
  Updates        : 0
  Too big errors : 0
  MT errors      : 0
```

pdweb.http component

The pdweb.http statistics component gathers information about WebSEAL HTTP communication. The following list describes the types of available information:

reqs
The total number of HTTP requests received.

max-worker
The maximum time that is used by a single worker thread to process an HTTP request.

total-worker
The total time that is used by all worker threads that process HTTP requests.

max-webseal
The maximum time that is used to process a single HTTP request - measured inside the worker thread, after the request headers are read, and eliminating connection setup overhead.

total-webseal
The total time that is used to process all HTTP requests - measured inside the worker threads, after the request headers are read, and eliminating connection setup overhead.

The following example shows the output from a **`stats get`** command for the pdweb.http component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.http
reqs             : 0
max-worker       : 0.000
total-worker     : 0.000
max-webseal      : 0.000
total-webseal    : 0.000
```

pdweb.http2stats component

The pdweb.http2stats statistics component gathers information about WebSEAL HTTP/2 communication. The following list describes the types of available information:

browser total connections
The total number of HTTP/2 requests received.

browser current connections
The number of active HTTP/2 connections.

junction total connections
The total number of requests sent to HTTP/2 junctions.

junction current connections
The number of active connections to HTTP/2 junctions.

browser total streams
The total number of HTTP/2 streams created.

browser current streams
Number of active HTTP/2 streams.

junction total streams
Total number of streams sent over HTTP/2 junctions.

junction current streams
Current active streams over HTTP/2 junctions.

browser idle timeouts
Number of HTTP/2 client connections closed due to idle timeout.

browser full timeouts
Number of HTTP/2 client connections closed due to session timeout.

browser exceeded max connections
Number of HTTP/2 client connections closed due to exceeding max connections.

browser stream read timeouts
Number of HTTP/2 client connections closed waiting on response.

junction stream read timeouts
Number of HTTP/2 junction connections closed waiting on response.

The following example shows the output from a **stats get** command for the pdweb.http2stats component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.http2stats

browser total connections : 0
browser current connections : 0
junction total connections : 0
junction current connections : 0
browser total streams : 0
browser current streams : 0
junction total streams : 0
junction current streams : 0
browser idle timeouts : 0
browser full timeouts : 0
browser exceeded max connections : 0
browser stream read timeouts : 0
junction stream read timeouts : 0
```

pdweb.https component

The pdweb.https statistics component gathers information about WebSEAL HTTPS communication. The following list describes the types of available information:

reqs
The total number of HTTPS requests received.

max-worker
The maximum time that is used by a single worker thread to process an HTTPS request.

total-worker
The total time that is used by all worker threads that process HTTPS requests.

max-webseal
The maximum time that is used to process a single HTTPS request - measured inside the worker thread, after the request headers are read, and eliminating connection setup overhead.

total-webseal
The total time that is used to process all HTTPS requests - measured inside the worker threads, after the request headers are read, and eliminating connection setup overhead.

The following example shows the output from a **stats get** command for the pdweb.https component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.https

reqs            : 0
max-worker      : 0.000
total-worker    : 0.000
max-webseal     : 0.000
total-webseal   : 0.000
```

pdweb.jct.# component

The pdweb.jct.# statistics component gathers information about configured junctions. The following list describes the types of available information:

[/]
The actual junction name (listed as the number in the command)

reqs
The total number of requests that are routed across this junction

max
The maximum time that is consumed by a single request across this junction

total
The total time that is consumed by requests across this junction

The following example shows the output from a **stats get** command for the pdweb.jct.1 component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.jct.1

[/]
reqs    : 0
max     : 0.000
total   : 0.000
```

pdweb.jmt component

The pdweb.jmt statistics component gathers information about the WebSEAL junction mapping table. This component is always enabled by default and cannot be disabled. The following list describes the types of available information:

hits
The total number of requests that required URL mapping with the junction mapping table.

The following example shows the output from a **stats get** command for the pdweb.jmt component:

```
#pdadmin> server task default-webseald-instance stats get pdweb.jmt

hits  : 5
```

pdweb.sescache component

The pdweb.sescache component gathers statistics about the WebSEAL session cache. This component gathers the following activity information:

hit
The number of requests where a cache entry for a user was referenced successfully. That is, the number of requests that resulted in a session cache hit.

miss
The number of requests that missed a session cache hit.

add
The number of cache entries that was added to the session cache.

del
The number of cache entries that was deleted from the session cache.

inactive
The number of times where a cache entry hit the inactivity timeout.

lifetime
The number of times where a cache entry hit the lifetime timeout.

LRU expired
The number of times that a "least recently used" cache entry was deleted from the session cache to make room for a new cache entry.

The following example shows the output from a **stats get** command for the pdweb.sescache component:

```
pdadmin sec_master>  server task default-webseald-instance stats get pdweb.sescache
hit      : 225
miss     : 75
add      : 375
del      : 150
inactive : 60
lifetime : 15
LRU expired : 75
```

In the previous release, the pdweb.sescache component contained activity that was associated with callback certificates and user session mappings. These statistics are now managed by the following components:

pdweb.certcallbackcache
This cache stores the SSL IDs of sessions that require certificate validation when a user is stepping up. The reported information has the same categories as pdweb.sescache. These activities are internal.

pdweb.usersessidcache
This cache stores a mapping of users to their sessions. The reported information has the same categories as pdweb.sescache. These activities are internal.

Therefore, the first time that you gather statistics for the pdweb.sescache component and compare it to your last report, the figures might appear to be wrong. To set a new baseline, add the statistics from the following components and then compare them to your previous baseline (last pdweb.sescache report):

- pdweb.sescache
- pdweb.certcallbackcache
- pdweb.usersessidcache

The output against the pdweb.sescache component must be your new baseline.

pdweb.threads component

The pdweb.threads statistics component gathers information about WebSEAL worker thread activity. Its report is the overall thread usage statistics that include not just request traffic, but all the worker threads for the WebSEAL process.

WebSEAL, version 6.0, and later can be configured to use multiple interfaces. Each separately configured interface can use a separate worker thread pool. The thread pool has the same name as the specified interface.

Alternatively, all configured interfaces can share worker thread pool. The default WebSEAL interface configuration uses the **default** name to differentiate between that interface and the corresponding thread pool, from other separately configured interfaces. The default WebSEAL interface configuration is defined under the [server] stanza. A separately configured WebSEAL interface (defined under the [interfaces] stanza) uses the specified name.

The pdweb.threads component is always enabled by default and cannot be disabled. The following list describes the types of available information:

active
The total number of active worker threads of all WebSEAL interfaces that are handling requests.

total
The total number of worker threads that are configured for all WebSEAL interfaces.

'default' active
The total number of active worker threads in the default interface thread pool that are handling requests. If you do not configure one or more more WebSEAL interfaces, the value of **default active** matches the value of **active**.

'default' total
The total number of configured worker threads for the default interface thread pool. If you do not configure one or more more WebSEAL interfaces, the value of **default total** matches the value of **total**.

'*other_interface*' active
The total number of active worker threads in the thread pool that is handling requests for an additional configured interface. *other_interface* is the name that is assigned to the interface.

'*other_interface*' total
The total number of worker threads in the thread pool that is used by an additional interface named *other_interface*.

The following example shows the output from a **`stats get`** command for the `pdweb.threads` component. The example assumes that no additional WebSEAL interface is configured:

```
#pdadmin> server task default-webseald-instance stats get pdweb.threads
active   : 0
total    : 50
'default' active : 0
'default' total : 50
```

Monitoring

Sending statistics to `Statsd`

WebSEAL provides a series of built-in software modules that, when enabled, can monitor specific server activity and collect information about those activities.

This statistical information is periodically sent to a remote statsd server over UDP.

The information that is gathered by WebSEAL statistics provides a relative view of the activity being recorded. If statistics are captured at regular intervals over a period of time, you can generate a graphical view of the relative relationship of the server activities.

Configuration

In order to enable statistics gathering the settings must be added to the WebSEAL configuration file in the `[statistics]` stanza. See the configuration options that are detailed here [statistics] stanza.

Example

An example configuration to enable statistics gathering is provided below:

```
[statistics]

server=statsd.ibm.com
port=8125
frequency=30
component= pweb.https
component = pdweb.junctions
```

Components

This topic lists the types of components for which statistics will be sent to the `statsd` server.

`pdweb.authn`

The `pdweb.authn` statistics component gathers information related to WebSEAL authentication. The following table describes the statistical information available:

Name	Description	Type
pdweb.authn.pass	The total number of successful authentications.	Counter
pdweb.authn.fail	The total number of failed authentications.	Counter
pdweb.authn.time	The time that it took to process an authentication operation.	Timer

pdweb.http

The pdweb.http statistics component gathers information about WebSEAL HTTP communication. The following table describes the statistical information available:

Name	Description	Type
pdweb.http.reqs	The total number of HTTP requests received.	Counter
pdweb.http.worker.time	The time that is used by a single worker thread to process a HTTP request.	Timer
pdweb.http.process.time	The time that is used to process a single HTTP request -measured inside the worker thread, after the request headers are read, and eliminating connection setup overhead.	Timer

pdweb.https

The pdweb.https statistics component gathers information about WebSEAL HTTPS communication. The following table describes the statistical information available:

Name	Description	Type
pdweb.https.reqs	The total number of HTTPS requests received.	Counter
pdweb.https.worker.time	The time that is used by a single worker thread to process a HTTPS request.	Timer
pdweb.https.process.time	The time that is used to process a single HTTPS request -measured inside the worker thread, after the request headers are read, and eliminating connection setup overhead.	Timer

pdweb.http2

The pdweb.http2 statistics component gathers information about WEBSEALHTTP/2 communication. The following table describes the statistical information available:

Name	Description	Type
pdweb.http2.browser_total_connections	The total number of HTTP/2 requests received.	Counter
pdweb.http2.browser_current_connections	The number of active HTTP/2 connections.	Gauge
pdweb.http2.application_total_connections	The total number of requests sent to HTTP/2 resource servers.	Counter
pdweb.http2.application_current_connections	The number of active HTTP/2 connections to resource servers.	Gauge
pdweb.http2.browser_total_streams	The total number of HTTP/2 streams created.	Counter
pdweb.http2.browser_current_streams	Number of active HTTP/2 streams.	Gauge
pdweb.http2.application_total_streams	Total number of streams sent to HTTP/2 resource servers.	Counter
pdweb.http2.application_current_streams	Current active streams over HTTP/2 resource servers.	Gauge
pdweb.http2.browser_idle_timeouts	Number of HTTP/2 client connections closed due to idle timeout.	Counter
pdweb.http2.browser_full_timeouts	Number of HTTP/2 client connections closed due to session timeout.	Counter
pdweb.http2.browser_exceeded_max_connections	Number of HTTP/2 client connections closed due to exceeding max connections.	Counter
pdweb.http2.browser_stream_read_timeouts	Number of HTTP/2 client connections closed waiting on a response.	Counter
pdweb.http2.application_stream_read_timeouts	Number of HTTP/2 resource server connections closed waiting on a response.	Counter

pdweb.jct

The pdweb.jct statistics component gathers information about configured junctions. The following table describes the statistical information available:

Name	Description	Type
pdweb.jct.<jct-id>.reqs	The total number of requests that are routed to this junction.	Counter
pdweb.jct.<jct-id>.time	The time that is consumed by a single request to this junction.	Timer

The <jct-id> component of the statistic name will match the name of the hosting junction,where the ':' character in the name is replaced with '_'. For standard junctions this will correspond to the configured 'path', and for virtual host junctions this will correspond to the configured 'virtual host'.

pdweb.redis

The pdweb.redis statistics component gathers information related to WebSEAL communication with Redis servers for remote session storage. The following table describes the statistical information available:

Name	Description	Type
pdweb.redis.<collection-name>.time	The time that is consumed by a single request to this collection of replicated Redis servers.	Timer
pdweb.redis.<collection-name>.reqs	The total number of requests which have been set to this collection of replicated Redis servers.	Counter

The <collection-name> component of the statistic name refers to the configured name of the collection of replicated Redis servers for which the statistic applies.

pdweb.sescache

The pdweb.sescache statistics component gathers information related to the WebSEAL session cache activity. The following table describes the statistical information available:

Name	Description	Type
pdweb.sescache.hit	The number of requests that resulted in a session cache hit -that is, the user had a session cache entry and it was successfully referenced.	Counter
pdweb.sescache.miss	The number of requests that missed a session cache hit.	Counter
pdweb.sescache.add	The number of entries that have been added to the session cache.	Counter
pdweb.sescache.del	The number of entries that have been deleted from the cache.	Counter
pdweb.sescache.inactive	The number of entries removed from the cache because the inactivity timeout value had expired.	Counter
pdweb.sescache.lifetime	The number of entries removed from the cache because the lifetime timeout value had expired.	Counter
pdweb.sescache.lru_expired	The number of times a "Least Recently Used" cache entry is expired or removed to make room for a new entry.	Counter

pdweb.threads

The pdweb.threads statistics component gathers information about WebSEAL worker thread activity. It reports the overall thread usage statistics that include not just request traffic, but all of the worker threads for the WebSEAL process. The following table describes the statistical information available:

Name	Description	Type
pdweb.threads.active	The total number of active worker threads that are handling requests.	Gauge

Name	Description	Type
pdweb.threads.total	The total number of worker threads that are configured.	Gauge

pdweb.websocket

The pdweb.websocket statistics component gathers information related to WebSEALWebSocket communication. The following table describes the statistical information available:

Name	Description	Type
pdweb.websocket.requests	Total WebSocket proxy requests received.	Counter
pdweb.websocket.rejected	Total WebSocket proxy requests rejected.	Counter
pdweb.websocket.timeouts	The number of timeouts that have occurred when reading or writing through a proxied WebSocket connection.	Counter
pdweb.websocket.active	The current number of WebSocket connections that are proxied.	Gauge
pdweb.websocket.client_bytes	The number of bytes read from the client side.	Counter
pdweb.websocket.application_bytes	The number of bytes read from the resource server.	Counter

Example output

Example 'statsd' output is shown below:

```
pdweb.authn.pass:2|c
pdweb.authn.time:392.5|ms|@0.5
pdweb.https.reqs:8|c
pdweb.https.worker.time:587.625|ms|@0.125
pdweb.https.process.time:99|ms|@0.125
pdweb.jct./.reqs:5|c
pdweb.jct./.time:0.4|ms|@0.2
pdweb.https.reqs:8|c
pdweb.https.worker.time:492|ms|@0.125
pdweb.https.process.time:1.5|ms|@0.125
pdweb.jct./.reqs:2|c
pdweb.jct./.time:1.5|ms|@0.5
```

Note: The protocol supports sending multiple metrics in a single packet by separating the metrics with a newline (\n) character. When you are using this method, it is important that the packet size does not exceed the Maximum Transmission Unit (MTU) of any single machine in the network traversal path. For this reason, an MTU size of 512 will be assumed, and a single packet will never exceed this length.

Chapter 5. Audit events

XML output of native audit events

When you use native Security Verify Access auditing, audit events are captured in the audit trail in a standard format with the Extensible Markup Language (XML) elements. XML is only an intermediary step to delivering a presentation view of the data. The XML file is in ASCII format and can be read directly or passed to other external parsing engines for further analysis.

DTD intermediate format

As an audit administrator, you are expected to select and extract events according to your own criteria. This activity might include reformatting each event by applying an appropriate Document Type Definition (DTD) or schema for the analysis tool that you are using. The DTD is an intermediate format that provides a description of the data that can be captured.

Data blocks and output elements

An entire audit trail does not represent a single XML document. Each audit event within the file is written as an isolated XML data block. Each data block conforms to the rules of standard XML syntax.

Sample authorization event

For example, the following data block is an audit record for getting user authorization credentials:

```
<event rev="1.2">
  <date>2005-11-14-16:25:08.341+00:00I-----</date>
  <outcome status="0">0</outcome>
  <originator blade="pdmgrd">
    <component rev="1.2">azn</component>
    <action>0</action>
    <location>phaedrus</location>
  </originator>
  <accessor name="">
    <principal auth="IV_LDAP_V3.0">sec_master</principal>
  </accessor>
  <target resource="3">
    <object>IV_LDAP_V3.0:sec_master</object>
  </target>
  <data>azn_id_get_creds</data>
</event>
```

Sample resource access event

For example, the following data block is an audit record for an HTTP request:

```
<event rev="1.2">
  <date>2005-10-02-22:01:36.187-04:00I-----</date>
  <outcome status="953091111" reason="unauthorized">1</outcome>
  <originator blade="webseald" instance="default">
    <component rev="1.2">http</component>
    <event_id>109</event_id>
    <action>1</action>
    <location>cmd.wma.ibm.com</location>
  </originator>
  <accessor name="unauthenticated">
    <principal auth="IV_UNAUTH_V3.0" domain="Default">Unauthenticated</principal>
    <user_location>9.54.83.206</user_location>
    <user_location_type>IPV4</user_location_type>
  </accessor>
  <target resource="5">
    <object>/</object>
    <object_nameinapp>HTTP://cmd.wma.ibm.com:80/</object_nameinapp>
  </target>
  <resource_access>
```

```
    <action>httpRequest</action>
    <httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>
    <httpmethod>GET</httpmethod>
    <httpresponse>200</httpresponse>
  </resource_access>
  <data>
    GET HTTP://cmd.wma.ibm.com:80/ HTTP/1.0
    1970
    Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
  </data>
</event>
```

Sample successful authentication events

For example, the following data block is an audit record for a successful authentication:

```
<event rev="1.2">
  <date>2005-10-02-21:59:31.980-04:00I-----</date>
  <outcome status="0">0</outcome>
  <originator blade="webseald" instance="default">
    <component rev="1.4">authn</component>
    <event_id>101</event_id>
    <action>0</action>
    <location>cmd.wma.ibm.com</location>
  </originator>
  <accessor name="">
    <principal auth="IV_LDAP_V3.0" domain="Default">testuser268</principal>
    <name_in_rgy>cn=testuser268,dc=ibm,dc=com</name_in_rgy>
    <session_id>56a701a4-33b1-11da-a8d3-00096bc369d2</session_id>
    <user_location>9.54.83.206</user_location>
    <user_location_type>IPV4</user_location_type>
  </accessor>
  <target resource="7">
    <object></object>
  </target>
  <authntype>formsPassword</authntype>
  <data></data>
</event>
```

Sample failed authentication events

For example, the following data block is an audit record for a failed authentication:

```
<event rev="1.2">
  <date>2005-10-02-21:59:31.977-04:00I-----</date>
  <outcome status="320938184" reason="authenticationFailure">1</outcome>
  <originator blade="webseald" instance="default">
    <component rev="1.4">authn</component>
    <event_id>101</event_id>
    <action>0</action>
    <location>cmd.wma.ibm.com</location>
  </originator>
  <accessor name="">
    <principal auth="" domain="">testuser335</principal>
    <user_location>9.54.83.206</user_location>
    <user_location_type>IPV4</user_location_type>
  </accessor>
  <target resource="7">
    <object></object>
  </target>
  <authntype>formsPassword</authntype>
  <data>
    Password Failure: testuser335
  </data>
</event>
```

Sample authentication terminate event

For example, the following data block is an audit record for the termination of an authentication:

```
<event rev="1.2">
  <date>2005-10-04-11:45:27.487-04:00I-----</date>
  <outcome status="0">0</outcome>
  <originator blade="webseald" instance="default">
    <component rev="1.4">authn</component>
```

```
    <event_id>103</event_id>
    <action>103</action>
    <location>cmd.wma.ibm.com</location>
  </originator>
  <accessor name="">
    <principal auth="IV_LDAP_V3.0" domain="Default">testuser1</principal>
    <name_in_rgy>cn=testuser1,dc=ibm,dc=com</name_in_rgy>
    <session_id>e005b3ae-34ed-11da-a016-00096bc369d2</session_id>
    <user_location>9.65.85.162</user_location>
    <user_location_type>IPV4</user_location_type>
  </accessor>
  <target resource="7">
    <object></object>
  </target>
  <authntype>formsPassword</authntype>
  <terminateinfo>
    <terminatereason>userLoggedOut</terminatereason>
  </terminateinfo>
  <data></data>
</event>
```

XML output elements

Table 8 on page 57 describes the XML output elements that are possible by using the default Security Verify Access DTD elements. If you create your own DTD, each element must represent the events that you selected and extracted according to your own criteria.

Table 8. Names and descriptions for XML output elements

Output element name	Description

`<event>`
`...`
`</event>`
	Auditing event. Each auditing event captures the result of an action. A principal attempts an action on a target object.
	The `event` element can include the following elements:
	• `date`
	• `outcome`
	• `originator`
	• `accessor`
	• `target`
	• `resource_access` (for resource access events)
	• `authntype` (for authentication events)
	• `terminationinfo` (for authentication terminate events)
	• `data`
	Because Security Verify Access auditing uses a standard record format, not all elements are relevant to each event that is recorded. Fields that are not relevant for a particular event might contain a default value.
	The `event` element can include the following attribute:
	• `rev`
	Example:
	`<event rev="1.2">`
	`  <date>2003-11-14-16:25:08.341+00:00I-----</date>`
	`  <outcome status="0">0</outcome>`
	`  ...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<date>`
`...`
`</date>`
	Current date and timestamp. The date element has the following format:
	yyyy-mm-dd-hh:*mm*:*ss*.*xxx*-*xx*:*xx*I-----
	Where:
	yyyy-mm-dd
	Relates to the year (*yyyy*), the month (*mm*), and the day (*dd*).
	hh*:*mm*:*ss
	Relates to hours (*hh*), minutes (*mm*), and seconds (*ss*).
	***xxx-xx*:*xx*I**
	Refers to the time zone.
	Example:
	`<event rev="1.2">`
	`<date>2005-11-14-16:25:08.341+00-----</date>`
	`...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<outcome>`
`...`
`</outcome>`
	Outcome of the event. The `outcome` element can be one of the following values:
	0
	Success
	1
	Failure
	2
	Pending
	3
	Unknown
	The following information is captured in a common format header of the audit record:
	• The outcome.
	• The action.
	• The credentials for the principal.
	• The target object.
	This element can include the following attributes:
	• `status`
	• `reason`
	Example of a failed event:
	`<outcome status="320938184" reason="authenticationFailure">`
	`1`
	`</outcome>`
	For information about the contents of the `status` attribute, use the **`errtext`** command. The command provides the error message that is associated with the status code (320938184) of a failed event. If the error is not identified by the **`errtext`** command, the error did not originate in Security Verify Access. See your third-party documentation for more status code definitions.
	For information about the contents of the `reason` attribute, see "Outcome output for failures" on page 84.
	Example of a successful event:
	`<event rev="1.2">`
	`...`
	`<outcome status="0">0</outcome>`
	`...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<originator>`
`...`
`</originator>`
	Server that originated the event being logged. The `originator` element can include the following elements:
	• `component`
	• `event_id`
	• `action`
	• `location`
	The `originator` element can include the following attributes:
	• `blade`
	• `instance`
	The `blade` attributes represents the server that originated the event. For example, `pdmgrd` is the Security Verify Access policy server, `webseald` is the Security Verify Access WebSEAL server. The `instance` attribute applies to WebSEAL and represents the name of the instance.
	Example:
	`<event rev="1.2">`
	`...`
	`<originator blade="webseald">`
	`<component rev="1.4">authn</component>`
	`<event_id>101</event_id>`
	`<action>0</action>`
	`<location>cmd.wma.ibm.com</location>`
	`</originator>`
	`...`
	`</event>`

`<component>`
`...`
`</component>`
	Audit events, categorized by the server functionality that generates them. Some functionality is common across Security Verify Access servers while other functionality is server-specific.
	The `component` element can be one of the following values:
	`authz or azn`
	Captures authorization events.
	`authn`
	Captures authentication events.
	`mgmt`
	Captures management events.
	`http`
	Captures WebSEAL HTTP events. See the Configuring topics in the Knowledge Center for more information about this value.
	The `component` element can contain the `rev` attribute.
	Example:
	`<originator blade="webseald">`
	`<component rev="1.4">authn</component>`
	`<event_id>101</event_id>`
	`<action>0</action>`
	`<location>cmd.wma.ibm.com</location>`
	`</originator>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<event_id>`
`...`
`</event_id>`
	The category of the event ID. The `event_id` element can be one of the following values:
	101
	Login
	102
	Password change
	103
	Logout
	104
	Authenticate
	105
	Step-up
	106
	Re-authentication
	107
	Credentials refresh
	108
	Authorization check
	109
	Resource access
	110
	Get credentials
	111
	Modify credentials/combine credentials
	112
	Get credentials from pac
	113
	Get pac
	114
	Get entitlements
	115
	Runtime start
	116
	Runtime stop
	117
	Runtime audit start
	118
	Runtime audit stop
	119
	Runtime audit level change
	120
	Runtime statistic
	121
	Runtime heartbeat up
	122
	Runtime heartbeat down
	123
	Runtime lost contact
	124
	Runtime contact restored

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<action>`
`...`
`</action>`
	Audit record action code, which can be for one of the following groups of events:
	Authentication or authorization events
	Audit records for authentication or authorization events contain one of the following event action codes:
	0
	Authentication or authorization events
	1
	Change password events
	2
	WebSEAL events
	Management events
	Audit records for management events contain an action code that identifies the **pdadmin** utility. For example, the `<action>13702</action>` action code relates to the `POP_MODIFY` action for the **pop modify** command. See "Action codes for management commands" on page 78, which relates the action code reference number for each command.
	A common format header of the audit record captures information about:
	• The action.
	• The credentials of the principal.
	• The target object.
	• The outcome.
	Example:
	`<originator blade="webseald">`
	`<component rev="1.4">authn</component>`
	`<event_id>101</event_id>`
	`<action>0</action>`
	`<location>cmd.wma.ibm.com</location>`
	`</originator>`

`<location>`
`...`
`</location>`
	The host name (location) of the machine. If there is no host name specified, a notation of "location not specified" is substituted in the `location` element.
	Example:
	`<originator blade="webseald">`
	`<component rev="1.4">authn</component>`
	`<event_id>101</event_id>`
	`<action>0</action>`
	`<location>cmd.wma.ibm.com</location>`
	`</originator>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<accessor>`
`...`
`</accessor>`
	The name of the user that caused the event. If there is no user name specified, a notation of "`name="user not specified""` or "`name=""`" is substituted in the `accessor` element.
	The `accessor` element can include the following elements:
	• `principal`
	• `name_in_rgy` (for authenticated users)
	• `session_id` (for authenticated users)
	• `principal`
	• `user_location`
	• `user_location_type`
	The `accessor` element includes the `name` attribute.
	The following example shown the `accessor` element for an unauthenticated user:
	`<event rev="1.2>`
	`  ...`
	`  <accessor name="unauthenticated">`
	`  <principal auth="IV_UNAUTH_V3.0" domain="Default">`
	`  testuser2`
	`  </principal>`
	`  <user_location>9.65.85.162</user_location>`
	`  <user_location_type>IPV4</user_location_type>`
	`  </accessor>`
	`  ...`
	`</event>`
	The following example shown the `accessor` element for an authenticated user:
	`<event rev="1.2>`
	`  ...`
	`  <accessor name="">`
	`  <principal auth="IV_LDAP_V3.0" domain="Default">`
	`  testuser2`
	`  </principal>`
	`  <name_in_rgy>`
	`  cn=testuser1,dc=ibm,dc=com`
	`  </name_in_rgy>`
	`  <session_id>`
	`  e005ba3-34ed-11da-a016-00096bc369d`
	`  </session_id>`
	`  <user_location>9.65.85.162</user_location>`
	`  <user_location_type>IPV4</user_location_type>`
	`  </accessor>`
	`  ...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<principal>`
`...`
`</principal>`
	User authorization credentials. Generally each event captures the result of an action that a user (principal) attempts on a target object. If there is no user name specified, a notation of "`auth="invalid"`" is substituted in the `principal` element.
	The `principal` element can contain the following attributes:
	• `auth`
	• `domain`
	To determine the actual authentication method, use the data in the `authntype` element.
	A common format header of the audit record captures information about:
	• The credentials of the principal.
	• The action.
	• The target object.
	• The outcome.
	Example:
	`<accessor name="">`
	`<principal auth="IV_LDAP_V3.0" domain="Default">`
	`testuser2`
	`</principal>`
	`<name_in_rgy>`
	`cn=testuser1,dc=ibm,dc=com`
	`</name_in_rgy>`
	`<session_id>`
	`e005ba3-34ed-11da-a016-00096bc369d`
	`</session_id>`
	`<user_location>9.65.85.162</user_location>`
	`<user_location_type>IPV4</user_location_type>`
	`</accessor>`

`<name_in_rgy>`
`...`
`</name_in_rgy>`
	The name in the registry for the user.
	Example:
	`<accessor name="">`
	`<principal auth="IV_LDAP_V3.0" domain="Default">`
	`testuser2`
	`</principal>`
	`<name_in_rgy>`
	`cn=testuser1,dc=ibm,dc=com`
	`</name_in_rgy>`
	`<session_id>`
	`e005ba3-34ed-11da-a016-00096bc369d`
	`</session_id>`
	`<user_location>9.65.85.162</user_location>`
	`<user_location_type>IPV4</user_location_type>`
	`</accessor>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<session_id>`
`...`
`</session_id>`
	The session ID that is associated with this session. This ID can be used to trace a series of events back to the authentication data that was initially provided by the user. For example, the data in the `session_id` element could be used to determine when a user logged in and when a user logged out.
	Example:
	`<accessor name="">`
	`  <principal auth="IV_LDAP_V3.0" domain="Default">`
	`  testuser2`
	`  </principal>`
	`  <name_in_rgy>`
	`  cn=testuser1,dc=ibm,dc=com`
	`  </name_in_rgy>`
	`  <session_id>`
	`  e005ba3-34ed-11da-a016-00096bc369d`
	`  </session_id>`
	`  <user_location>9.65.85.162</user_location>`
	`  <user_location_type>IPV4</user_location_type>`
	`</accessor>`

`<user_location>`
`...`
`</user_location>`
	The IP address in IPv4 or IPv6 format.
	Example:
	`<accessor name="">`
	`  <principal auth="IV_LDAP_V3.0" domain="Default">`
	`  testuser2`
	`  </principal>`
	`  <name_in_rgy>`
	`  cn=testuser1,dc=ibm,dc=com`
	`  </name_in_rgy>`
	`  <session_id>`
	`  e005ba3-34ed-11da-a016-00096bc369d`
	`  </session_id>`
	`  <user_location>9.65.85.162</user_location>`
	`  <user_location_type>IPV4</user_location_type>`
	`</accessor>`

`<user_location_type>`
`...`
`</user_location_type>`
	The format of the data in the `user_location` element. Values are:
	• IPV4
	• IPV6
	Example:
	`<accessor name="">`
	`  <principal auth="IV_LDAP_V3.0" domain="Default">`
	`  testuser2`
	`  </principal>`
	`  <name_in_rgy>`
	`  cn=testuser1,dc=ibm,dc=com`
	`  </name_in_rgy>`
	`  <session_id>`
	`  e005ba3-34ed-11da-a016-00096bc369d`
	`  </session_id>`
	`  <user_location>9.65.85.162</user_location>`
	`  <user_location_type>IPV4</user_location_type>`
	`</accessor>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<target>`
`...`
`</target>`
	Target information. The `target` element can include the following elements:
	• `object`
	• `object_nameinapp`
	• `process`
	• `azn`
	• `url`
	The `target` element includes the `resource` attribute, which represents a broad categorization of the target object: The `resource` attribute can be one of the following values:
	0
	AUTHORIZATION
	1
	PROCESS
	2
	TCB
	3
	CREDENTIAL
	5
	GENERAL
	6
	APPLICATION
	7
	AUTHENTICATION
	Examples:
	`<target resource="7">`
	`<object></object>`
	`</target>`
	`<target resource="3">`
	`<object>IV_LDAP_V3.0:sec_master</object>`
	`</target>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<object>`
`...`
`</object>`
	Target object. Authorization audit records can be captured when a target object in the policy database (protected object space) has a POP attached to it. The POP must enable audit functionality. For example:
	`<object>/Management</object>`
	A common format header of the audit record captures information about:
	• The target object.
	• The action.
	• The user credentials.
	• The outcome.
	Example:
	`<target resource="3">`
	`  <object>IV_LDAP_V3.0:sec_master</object>`
	`</target>`

`<url>`
`…`
`</url>`
	The URL which was accessed to cause the authentication event.
	Example:
	`<target resource="3">`
	`  <url>https://www.ibm.com/security-verify-access</url>`
	`</target>`

`<azn>`
`...`
`</azn>`
	Authorization service information. The authorization service:
	• Checks the access permissions on the target requested object.
	• Compares these access permissions with the capabilities of the requesting user.
	The `azn` element can include the following elements:
	• `perm`
	• `result`
	• `qualifier`
	`<target resource="3">`
	`  ...`
	`  <azn>`
	`  <perm>64</perm>`
	`  <result>0</result>`
	`  <qualifier>0</qualifier>`
	`  </azn>`
	`  ...`
	`</target>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<perm>`
`...`
`</perm>`
	Set of controls (permissions) that specifies the conditions necessary to complete certain operations on that resource. The permission can be specified in this element by using either the binary number such as `<perm>64</perm>` or the letters for the specified action permissions such as `<perm>Tr</perm>`.
	Example:
	`<target resource="3">`
	`  ...`
	`  <azn>`
	`  **<perm>64</perm>**`
	`  <result>0</result>`
	`  <qualifier>0</qualifier>`
	`  </azn>`
	`  ...`
	`</target>`

`<result>`
`...`
`</result>`
	Results of the authorization service check.
	Example:
	`<target resource="3">`
	`  ...`
	`  <azn>`
	`  <perm>64</perm>`
	`  **<result>0</result>**`
	`  <qualifier>0</qualifier>`
	`  </azn>`
	`  ...`
	`</target>`

`<qualifier>`
`...`
`</qualifier>`
	Qualifier information.
	Example:
	`<target resource="3">`
	`  ...`
	`  <azn>`
	`  <perm>64</perm>`
	`  <result>0</result>`
	`  **<qualifier>0</qualifier>**`
	`  </azn>`
	`  ...`
	`</target>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<process>`
`...`
`</process>`
	Type of process. The `process` element can include the following elements:
	• `pid` (process ID)
	• `uid` (user ID)
	• `eid` (effective user ID)
	• `gid` (group ID)
	• `egid` (effective group ID)
	The `process` element includes the `architecture` attribute, which is one of the following values:
	0
	For AIX, Linux, and Solaris operating systems.
	1
	For Windows operating systems.
	Example:
	`<process architecture="0">`
	`...`
	`<pid></pid>`
	`</process>`

`<pid></pid>`
`<eid></eid>`
`<uid></uid>`
`<gid></gid>`
`<egid></egid>`
	The identifier of the process, which is contained in one of the following elements:
	pid
	Process ID
	eid
	Effective user ID
	uid
	User ID
	gid
	Group ID
	egid
	Effective group ID
	Example:
	`<process architecture="0">`
	`...`
	`<pid>3899</pid>`
	`</process>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<policy>`
`...`
`</policy>`
	The security policy information. The `policy` element can include the following elements:
	• `name`
	• `type`
	• `descr`
	Example of name element for policy element:
	`<policy>`
	`  <name>real-traders-only</name>`
	`  <type>rule</type>`
	`</policy>`

`<name>`
`...`
`</name>`
	Name of the policy attribute that you want to audit. The name matches the name that you specified in a list of attributes in the `[aznapi-configuration]` stanza of the appropriate configuration file. For example:
	`[aznapi-configuration]`
	`audit-attribute = real-traders-only`
	Example:
	`<policy>`
	`  <name>real-traders-only</name>`
	`  <type>rule</type>`
	`</policy>`

`<type>`
`...`
`</type>`
	Type of security policy being audited. The `type` element can contain the following values:
	• `ACL`
	• `POP`
	• `rule`
	Example:
	`<policy>`
	`  <name>traders-pop</name>`
	`  <type>POP</type>`
	`</policy>`

`<descr>`
`...`
`</descr>`
	Description of the security policy. This element is empty if no description was created for the policy.
	Example:
	`<policy><name>traders-acl</name>`
	`  <type>ACL</type>`
	`  <descr>traders that have ACL security policies</descr>`
	`</policy>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<attribute>`
`...`
`</attribute>`
	The container for the characteristics of the access decision information (ADI) attribute to audit. An attribute can establish accountability by providing information to help identify potentially inappropriate access of assets. You can grant or deny access based on rules applied to attributes.
	The `attribute` element can include the following elements:
	• `name`
	• `source`
	• `type`
	• `value`
	Example:
	`<attribute>`
	`  <name>tagvalue_su-admin</name>`
	`  <source>cred</source>`
	`  <type>string</type>`
	`  <value>test_customer_service_rep_1</value>`
	`</attribute>`

`<name>`
`...`
`</name>`
	Name of the ADI to audit. This ADI can be for auditing either a user credential if for the `authn` component or an app_context if for an `azn` component.
	The name of the authorization attribute matches the name that you specified in a list of attributes in the `[aznapi-configuration]` stanza of the appropriate configuration file. For example:
	`[aznapi-configuration]`
	`audit-attribute = AZN_CRED_AUTH_METHOD`
	Example of name element for the `attribute` element:
	`<attribute>`
	`  <name>AZN_CRED_AUTH_METHOD</name>`
	`  <source>credADI</source>`
	`  <type>string</type>`
	`  <value>su-forms</value>`
	`</attribute>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<source>`
`...`
`</source>`
	The source event. The source element can contain one of the following values:
	cred
	Applies to any Security Verify Access component.
	app
	Applies only to an authorization (azn) component.
	credADI
	Applies only to the authorization (azn) component when evaluating a Boolean rule.
	appADI
	Applies only to the authorization (azn) component when evaluating a Boolean rule.
	engineADI
	Applies only to the authorization (azn) component when evaluating a Boolean rule.
	dynADI
	Applies only to the authorization (azn) component when evaluating a Boolean rule.
	If the ADI attribute is multi-valued, a separate attribute element is written for each value.
	Example:
	`<attribute>`
	`  <name>AZN_CRED_AUTH_METHOD</name>`
	`  <source>credADI</source>`
	`  <type>string</type>`
	`  <value>su-forms</value>`
	`</attribute>`

`<type>`
`...`
`</type>`
	Type of security policy that is being audited. The type element can contain one of the following values:
	• `string`
	• `ulong`
	• `pobj`
	If `<type>pobj</type>`, the value is the name of the protected object.
	Example:
	`<attribute>`
	`  <name>AZN_CRED_AUTH_METHOD</name>`
	`  <source>credADI</source>`
	`  <type>string</type>`
	`  <value>su-forms</value>`
	`</attribute>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<value>`
`...`
`</value>`
	Value for the `aznAPI` attribute. If the ADI attribute is multi-valued, then a separate attribute element is written for each value.
	Example:
	`<attribute>`
	`  <name>AZN_CRED_AUTH_METHOD</name>`
	`  <source>credADI</source>`
	`  <type>string</type>`
	`  **<value>su-forms</value>**`
	`</attribute>`

`<resource_access>`
`...`
`</resource_access>`
	Example:
	`<event rev="1.2">`
	`  ...`
	`  **<resource_access>**`
	`  <action>httpRequest</action>`
	`  <httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>`
	`  <httpmethod>GET</httpmethod>`
	`  <httpresponse>200</httpresponse>`
	`  **</resource_access>**`
	`  ...`
	`</event>`

`<action>`
`...`
`</action>`
	Example:
	`<event rev="1.2">`
	`  ...`
	`  <resource_access>`
	`  **<action>httpRequest</action>**`
	`  <httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>`
	`  <httpmethod>GET</httpmethod>`
	`  <httpresponse>200</httpresponse>`
	`  </resource_access>  ...`
	`</event>`

`<httpurl>`
`...`
`</httpurl>`
	Example:
	`<event rev="1.2">`
	`  ...`
	`  <resource_access>`
	`  <action>httpRequest</action>`
	`  **<httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>**`
	`  <httpmethod>GET</httpmethod>`
	`  <httpresponse>200</httpresponse>`
	`  </resource_access>  ...`
	`</event>`

`<httpmethod>`
`...`
`</httpmethod>`
	Example:
	`<event rev="1.2">`
	`  ...`
	`  <resource_access>`
	`  <action>httpRequest</action>`
	`  <httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>`
	`  **<httpmethod>GET</httpmethod>**`
	`  <httpresponse>200</httpresponse>`
	`  </resource_access>  ...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<httpresponse>`
`...`
`</httpresponse>`
	Example:
	`<event rev="1.2">`
	`  ...`
	`  <resource_access>`
	`  <action>httpRequest</action>`
	`  <httpurl>HTTP://cmd.wma.ibm.com:80/</httpurl>`
	`  <httpmethod>GET</httpmethod>`
	`  **<httpresponse>200</httpresponse>**`
	`  </resource_access>  ...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<authntype>`
`...`
`</authntype>`
	The type of authentication that the user completed. The following strings are authentication types that are associated with WebSEAL and Plug-in for Web Servers:
	itamFailoverCookie
	Failover cookie
	itamCDSSO
	WebSEAL or Plug-in for Web Servers authentication using cross domain single-sign on (CDSSO)
	itamECSSO
	WebSEAL or Plug-in for Web Servers authentication using e-Community single-sign on (ECSSO)
	certificate
	SSL certificate authentication
	twoFactor
	WebSEAL or Plug-in for Web Servers using token authentication
	formsPassword
	Password authentication using an HTML form
	basicAuthRFC2617
	Password authentication using HTTP Basic Authentication (BA)
	passwordOther
	Password authentication using an undetermined mechanism
	itamHTTPHeader
	WebSEAL or Plug-in for Web Servers using HTTP header authentication
	itamIPAddress
	WebSEAL or Plug-in for Web Servers using IP address-based authentication
	kerberos
	WebSEAL or Plug-in for Web Servers using SPNEGO authentication
	itamEAI
	WebSEAL or Plug-in for Web Servers using external authentication interface (EAI) authentication
	itamIVCreds
	Plug-in for Web Servers authentication using the IV_CREDS header
	itamIVUser
	Plug-in for Web Servers authentication using the IV_USER header
	tokenLTPA
	Plug-in for Web Servers authentication using a lightweight third-party authentication (LTPA) token
	ntlm
	Plug-in for Web Servers using NTLM authentication
	itamWebServerAuthentication
	Plug-in for Web Servers authentication that is provided by the hosting Web server
	Example:
	`<event rev="1.2">`
	`...`
	`<authntype>formsPassword</authntype>`
	`...`
	`</event>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<terminateinfo>`
`...`
`</terminateinfo>`
	Contains information about why a session ended. The `terminateinfo` element contains the `terminatereason` element.
	Example:
	`<event rev="1.2">`
	`  ...`
	`  <terminateinfo>`
	`  <terminatereason>userLoggedOut</terminatereason>`
	`  </terminateinfo>`
	`  ...`
	`</event>`

`<terminatereason>`
`...`
`</terminatereason>`
	The reason why the session ended. The following values are possible:
	idleTimeout
	The session timed out because the user was inactive.
	sessionExpired
	The session timed out because the user was logged in for too long.
	sessionDisplaced
	The session ended because another user with the same user ID logged in.
	sessionTerminatedByAdmin
	The session ended because an administrator logged out the user.
	userLoggedOut
	The session ended because the user logged out.
	reathLockOut
	The session ended because the user did not authenticate again.
	Example:
	`<terminateinfo>`
	`  <terminatereason>userLoggedOut</terminatereason>`
	`</terminateinfo>`

Table 8. Names and descriptions for XML output elements (continued)

Output element name	Description

`<data>`
`...`
`</data>`
	Event-specific data. The `data` element can contain the `audit` element.
	
	Additional event-specific information is recorded in a free format data area at the end of the event record. For failed authentication attempts, "Data output for errors" on page 84 provides details about the data information that is returned.
	
	Note: Decoding the meaning of certain data values in the record might require an advanced knowledge of the Security Verify Access code and architecture.
	
	Command arguments are listed in the `data` element of the event record in their internal format. For example:
	
	`<data>azn_id_get_creds</data>`
	
	Commands that do not result in an effective state change (**`list`** and **`show`**) are never captured.
	
	Examples:
	
	• <event>
	...
	<data>
	POST /pkmspasswd.form HTTP/1.1 0 Mozilla/4.0
	(compatible; MSIE 6.0; Windows NT 5.0)
	https://c03comcrit2.somecompany.com/pkmspasswd
	</data>
	</event>
	
	• **<data>**
	"2019"
	"1002"
	"pop1"
	"0"
	""
	</data>

`<audit/>`	Beginning and ending of an audit event. The `audit` element can include the `event` attribute, which can be one of the following values: • `Start` • `Stop` Example: <event rev="1.2"> ... <data> **<audit event="Start"/>** </data> </event> ... <event rev="1.2"> ... <data> **<audit event="Stop"/>** </data> </event>

Action codes for management commands

The action code identifies one of the **pdadmin** management commands. The tables in this section relate the action code reference number for each management command. For example, the action code 13702 relates to the POP_MODIFY action command. In other words, the **pdadmin pop modify** command.

Command arguments are listed in the data section of the event record in their internal format. Commands that do not result in an effective change of state of the database (such as the **list** and **show** commands) are never captured.

Table 9 on page 78 maps the action codes to the management commands.

Table 9. Mapping of action codes to management commands

Action code	Management command
13000	ACL_LIST
13001	ACL_GET
13002	ACL_SET_LEGACY
13003	ACL_DELETE
13005	ACL_FIND
13006	ACTION_LIST
13007	ACTION_SET
13008	ACTION_DELETE
13009	ACTION_GROUPLIST
13010	ACTION_GROUPCREATE
13011	ACTION_GROUPDELETE
13012	ACTION_LISTGROUP
13013	ACTION_CREATEGROUP
13014	ACTION_DELETEGROUP
13020	ACL_CREATE
13021	ACL_SET
13100	OBJ_GET
13101	OBJ_ACL_SET (deprecated)
13102	OBJ_GET_OBJ
13103	OBJSPC_CREATE
13104	OBJSPC_DELETE
13105	OBJSPC_LIST
13106	OBJ_CREATE
13107	OBJ_DELETE
13110	OBJ_MOD_SET_NAME
13111	OBJ_MOD_SET_DESC
13112	OBJ_MOD_SET_TYPE
13113	OBJ_MOD_SET_ISLF

Table 9. Mapping of action codes to management commands (continued)

Action code	Management command
13114	OBJ_MOD_SET_ISPOL
13115	OBJ_MOD_SET_ATTR
13116	OBJ_MOD_DEL_ATTR
13117	OBJ_MOD_DEL_ATTRVAL
13118	OBJ_SHOW_ATTR
13119	OBJ_LIST_ATTR
13120	ACL_ATTACH
13121	ACL_DETACH
13123	ACL_MOD_SET_ATTR
13124	ACL_MOD_DEL_ATTR
13125	ACL_MOD_DEL_ATTRVAL
13126	ACL_SHOW_ATTR
13127	ACL_LIST_ATTR
13128	POP_MOD_SET_ATTR
13129	POP_MOD_DEL_ATTR
13130	POP_MOD_DEL_ATTRVAL
13131	POP_SHOW_ATTR
13132	POP_LIST_ATTR
13133	OBJ_SHOW_ATTRS
13134	ACL_SHOW_ATTRS
13135	POP_SHOW_ATTRS
13136	OBJ_SHOW_V417
13137	OBJ_LIST
13138	OBJ_LISTANDSHOW_V417
13139	OBJ_EXISTS (deprecated)
13140	OBJ_ACCESS_CHECK
13141	OBJ_SHOW
13142	OBJ_LISTANDSHOW
13150	ACL_CREATE_ATTR (deprecated, see 13134)
13200	SERVER_GET
13201	SERVER_RESTORE
13202	SERVER_DELETE (deprecated)
13203	SERVER_LIST
13204	SERVER_PERFORMTASK

Table 9. Mapping of action codes to management commands (continued)

Action code	Management command
13205	SERVER_GETTASKLIST
13206	SERVER_REPLICATE
13207	SERVER_ACTION
13208	SERVER_STATUS_GET
13209	SERVER_ENABLE (deprecated)
13210	SERVER_DISABLE (deprecated)
13400	ADMIN_SHOWCONF
13401	USER_CREATE
13402	USER_IMPORT
13403	USER_MODDESC
13404	USER_MODPWD
13405	USER_MODAUTHMECH
13406	USER_MODACCVALID
13407	USER_MODPWDVALID
13408	USER_DELETE
13409	USER_SHOWGROUPS
13410	USER_SHOW
13411	USER_SHOWDN
13412	USER_LIST
13413	USER_LISTDN
13414	GROUP_CREATE
13415	GROUP_IMPORT
13416	GROUP_MODDESC
13417	GROUP_MODADD
13418	GROUP_MODREMOVE
13419	GROUP_DELETE
13420	GROUP_SHOW
13421	GROUP_SHOWDN
13422	GROUP_LIST
13423	GROUP_LISTDN
13424	GROUP_SHOWMEMB
13425	USER_MODGSOUSER
13426	USER_SET (deprecated)
13427	GROUP_SET (deprecated)

Table 9. Mapping of action codes to management commands (continued)

Action code	Management command
13428	GROUP_MODADD2
13500	GSO_RESOURCE_CREATE
13501	GSO_RESOURCE_DELETE
13502	GSO_RESOURCE_LIST
13503	GSO_RESOURCE_SHOW
13504	GSO_RESOURCE_CRED_CREATE
13505	GSO_RESOURCE_CRED_DELETE
13506	GSO_RESOURCE_CRED_MODIFY
13507	GSO_RESOURCE_CRED_LIST
13508	GSO_RESOURCE_CRED_SHOW
13509	GSO_RESOURCE_GROUP_CREATE
13510	GSO_RESOURCE_GROUP_DELETE
13511	GSO_RESOURCE_GROUP_ADD
13512	GSO_RESOURCE_GROUP_REMOVE
13513	GSO_RESOURCE_GROUP_LIST
13514	GSO_RESOURCE_GROUP_SHOW
13600	POLICY_SET_MAX_LOGIN_FAILURES
13601	POLICY_GET_MAX_LOGIN_FAILURES
13602	POLICY_SET_DISABLE_TIME_INTERVAL
13603	POLICY_GET_DISABLE_TIME_INTERVAL
13604	POLICY_SET_MAX_ACCOUNT_AGE
13605	POLICY_GET_MAX_ACCOUNT_AGE
13606	POLICY_SET_ACCOUNT_EXPIRY_DATE
13607	POLICY_GET_ACCOUNT_EXPIRY_DATE
13608	POLICY_SET_MAX_INACTIVITY_TIME
13609	POLICY_GET_MAX_INACTIVITY_TIME
13610	POLICY_GET_ACCOUNT_CREATION_DATE
13611	POLICY_GET_LAST_LOGIN_ATTEMPT_DATE
13612	POLICY_SET_MAX_PASSWORD_AGE
13613	POLICY_GET_MAX_PASSWORD_AGE
13614	POLICY_SET_MIN_PASSWORD_AGE
13615	POLICY_GET_MIN_PASSWORD_AGE
13616	POLICY_SET_MAX_PASSWORD_REPEATED_CHARS
13617	POLICY_GET_MAX_PASSWORD_REPEATED_CHARS

Table 9. Mapping of action codes to management commands (continued)

Action code	Management command
13618	POLICY_SET_MIN_PASSWORD_ALPHAS
13619	POLICY_GET_MIN_PASSWORD_ALPHAS
13620	POLICY_SET_MIN_PASSWORD_NON_ALPHAS
13621	POLICY_GET_MIN_PASSWORD_NON_ALPHAS
13622	POLICY_SET_MIN_PASSWORD_DIFFERENT_CHARS
13623	POLICY_GET_MIN_PASSWORD_DIFFERENT_CHARS
13624	POLICY_SET_PASSWORD_SPACES
13625	POLICY_GET_PASSWORD_SPACES
13626	POLICY_SET_MIN_PASSWORD_LENGTH
13627	POLICY_GET_MIN_PASSWORD_LENGTH
13628	POLICY_SET_MIN_PASSWORD_REUSE_TIME
13629	POLICY_GET_MIN_PASSWORD_REUSE_TIME
13630	POLICY_GET_PASSWORD_FAILURES
13631	POLICY_GET_LAST_PASSWORD_CHANGE_DATE
13632	POLICY_SET_NUMBER_WARN_DAYS
13633	POLICY_GET_NUMBER_WARN_DAYS
13634	POLICY_SET_PASSWORD_REUSE_NUM
13635	POLICY_GET_PASSWORD_REUSE_NUM
13636	POLICY_SET_TOD_ACCESS
13637	POLICY_GET_TOD_ACCESS
13638	POLICY_GET_ALL_POLICY
13639	POLICY_SET_MAX_CONCURRENT_WEB_SESSIONS
13640	POLICY_GET_MAX_CONCURRENT_WEB_SESSIONS
13700	POP_CREATE
13701	POP_DELETE
13702	POP_MODIFY
13703	POP_SHOW
13704	POP_LIST
13705	POP_ATTACH
13706	POP_DETACH
13707	POP_FIND
13800	CFG_CONFIG
13801	CFG_UNCONFIG
13802	CFG_RENEWCERT

Table 9. Mapping of action codes to management commands (continued)

Action code	Management command
13803	CFG_SETPORT
13804	CFG_SETLISTENING
13805	CFG_SETKEYRINGPWD
13806	CFG_SETSSLTIMEOUT
13807	CFG_SETAPPLCERT
13808	CFG_ADDREPLICA
13809	CFG_CHGREPLICA
13810	CFG_RMVREPLICA
13811	CFG_GETVALUE
13812	CFG_SETVALUE
13813	CFG_RMVVALUE
13814	CFG_SETSVRPWD
13900	DOMAIN_CREATE
13901	DOMAIN_DELETE
13902	DOMAIN_MODIFY_DESC
13903	DOMAIN_SHOW
13904	DOMAIN_LIST
13950	AUTHZRULE_CREATE
13951	AUTHZRULE_DELETE
13952	AUTHZRULE_MODIFYTEXT
13953	AUTHZRULE_MODIFYREASON
13954	AUTHZRULE_MODIFYDESC
13955	AUTHZRULE_SHOW
13956	AUTHZRULE_LIST
13957	AUTHZRULE_ATTACH
13958	AUTHZRULE_DETACH
13959	AUTHZRULE_FIND
13960	AUTHZRULE_MOD_SET_ATTR
13961	AUTHZRULE_MOD_DEL_ATTR
13962	AUTHZRULE_MOD_DEL_ATTRVAL
13963	AUTHZRULE_SHOW_ATTRS
13964	AUTHZRULE_SHOW_ATTR
13965	AUTHZRULE_LIST_ATTR

Authentication failures

The reason for authentication failure is included in two different locations in the authentication audit event:

- The `data` element
- The `outcome` element

Primarily, the `data` element is for compatibility with the earlier version of audit events. Later versions of audit events use the `outcome` element.

Data output for errors

Table 10 on page 84 lists the authentication error codes and the `data` output element structures that are returned when an authentication attempt fails.

Table 10. Authentication errors

Error type	Error code (in hex)	Error code (in decimal)	Generated XML

Password failure	132120c8	320938184	`<data>`
			`Password failure: user`
			`</data>`

Account lock-out	13212132	320938290	`<data>`
			`Account lock-out: user`
			`</data>`

General failure	All others	All others	`<data>`
			`<username>user</username>`
			`</data>`

Outcome output for failures

The `outcome` element provides more detailed information about the authentication failure. The following snippet of an audit event shows the `outcome` element:

```
<outcome status="320938184" reason="authenticationFailure">
```

The following list explains the meaning for the `reason` attribute of the `outcome` element:

accountDisabled
The account is disabled.

accountDisabledRetryViolation
The account was disabled because of a violation of the max-login-failures policy. The account was permanently disabled.

accountExpired
The account is expired or disabled.

accountLockedOutMaxLoginFail
The login failed because the account is temporarily disabled due to the max-login-failures policy.

authenticationFailure
General authentication failure, including incorrect password.

certificateFailure
Incorrect SSL certificate.

invalidUserName
Incorrect user name.

nextToken
Token authentication requires next token.

passwordExpired
The password expired and must be changed.

pinRequired
Token authentication requires a new PIN (personal identification number).

policyViolationMaxLotginsReached
Violation of the max-concurrent-web-session policy.

policyViolationTOD
Violation of the time-of-day policy.

userNameMismatch
Attempt at authentication or step-up authenticate failed because the user name that was provided did not match the previous user name.

Elements by event types

This section lists the elements that are available for each common audit event type.

For each event type, this documentation provides a description of the event and a listing of the available element. For each available element, the table provides the element name, whether it is always in the event output, and its abbreviated XPath statement.

The abbreviated XPath statement is represented in one of the following ways:

element
element_type.element

When the representation is *element*, the full XPath statement would be:

```
CommonBaseEvent/extendedDataElements[@name='element']/values
```

When the representation is *element_type.element*, the full XPath statement would be:

```
CommonBaseEvent/extendedDataElements[@name='element_type']/children
[@name='element']/values
```

For detailed information about these elements and element types, see "Reference information about elements and element types" on page 121.

Elements for AUDIT_AUTHN events

This event type identifies authentication events.

The following table lists the elements that can be displayed in the output of an AUDIT_AUTHN event and their abbreviated XPath statements.

Table 11. Elements used in AUDIT_AUTHN events

Element	Always in output	Abbreviated XPath
action	No	`action`
Action ID	No	`ActionID`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
authenProvider	No	`authenProvider`
Authn Policy URI	No	`AuthnPolicyURI`

Table 11. Elements used in AUDIT_AUTHN events (continued)

Element	Always in output	Abbreviated XPath
authnType	Yes	`authnType`
authnTypeVersion	No	`authnTypeVersion`
Authentication Method	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration.
Authentication Method ID	No	`authMethod.id`
Authentication Method Type	No	`authMethod.type`
Authentication Method Enabled	No	`authMethod.enabled`
Authentication Method Algorithm	No	`authMethod.algorithm`
Authentication Method Public Key	No	`authMethod.publicKey`
Authentication Method Key Handle	No	`authMethod.keyHandle`
Challenge	No	`Challenge`
Data Type	No	Parameters.Parameters0.dataType
Date created	No	`DateCreated`
Date modified	No	`DateModified`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_AUTHN'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
issuer	No	Parameters.Parameters0.issuer
name	No	Parameters.Parameters0.name
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
Parameter	No	This element is a container element and has no valid XPath. A valid XPath requires a `value` declaration
Parameters	No	This element is a container element and has no valid XPath. A valid XPath require `values` declaration.
partner	No	`partner`
progName	No	`progName`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`

Table 11. Elements used in AUDIT_AUTHN events (continued)

Element	Always in output	Abbreviated XPath
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
Requested URL	No	`RequestedURL`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
Signed Challenge	No	`SignedChallenge`
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
State	No	`State`
timestamp	Yes	`CommonBaseEvent/@creationTime`
Transaction ID	No	`TransactionID`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
sessionId	No	`userInfo.sessionId`
uniqueId	No	`userInfo.uniqueId`

Table 11. Elements used in AUDIT_AUTHN events (continued)		
Element	**Always in output**	**Abbreviated XPath**
uri	No	Parameters.Parameters0.uri
value	No	Parameters.Parameters0.value

Elements for AUDIT_AUTHN_CREDS_MODIFY events

This event type modifies credentials for a given user identity.

The following table lists the elements that can be displayed in the output of an AUDIT_AUTHN_CREDS_MODIFY event and their abbreviated XPath statements.

Table 12. Elements used in AUDIT_AUTHN_CREDS_MODIFY events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_AUTHN'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`

Table 12. Elements used in AUDIT_AUTHN_CREDS_MODIFY events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
sessionId	No	`userInfo.sessionId`
uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_AUTHN_MAPPING events

This event type records the mapping of principal and credentials where there are two user identities involved.

The following table lists the elements that can be displayed in the output of an AUDIT_AUTHN_MAPPING event and their abbreviated XPath statements.

Table 13. Elements used in AUDIT_AUTHN_MAPPING events		
Element	**Always in output**	**Abbreviated XPath**
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`

Table 13. Elements used in AUDIT_AUTHN_MAPPING events (continued)

Element	Always in output	Abbreviated XPath
extensionName	Yes	`"'AUDIT_AUTHN'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
mappedRealm	No	`mappedRealm`
mappedSecurityDomain	Yes	`mappedSecurityDomain`
mappedUserName	Yes	`mappedUserName`
originalRealm	No	`originalRealm`
originalSecurityDomain	Yes	`originalSecurityDomain`
originalUserName	Yes	`originalUserName`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`

Elements for AUDIT_AUTHN_TERMINATE events

This event type identifies authentication termination events.

The following table lists the elements that can be displayed in the output of an AUDIT_AUTHN_TERMINATE event and their abbreviated XPath statements.

Table 14. Elements used in AUDIT_AUTHN_TERMINATE events

Element	Always in output	Abbreviated XPath
action	No	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
authnType	Yes	`authnType`
authnTypeVersion	No	`authnTypeVersion`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_AUTHN'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
loginTime	Yes	`loginTime`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
progName	No	`progName`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`

Table 14. Elements used in AUDIT_AUTHN_TERMINATE events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
terminateReason	When `action` is `logout`	`terminateReason`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	When `action` is `logout`	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
sessionId	No	`userInfo.sessionId`
uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_AUTHZ events

This event type identifies authorization events.

The following table lists the elements that can be displayed in the output of an AUDIT_AUTHZ event and their abbreviated XPath statements.

Table 15. Elements used in AUDIT_AUTHZ events		
Element	**Always in output**	**Abbreviated XPath**
accessDecision	When `outcome.result` is `SUCCESSFUL`	`accessDecision`
accessDecisionReason	When `accessDecision` is `Denied`	`accessDecisionReason`
action	No	`action`
appName	No	`appName`

Table 15. Elements used in AUDIT_AUTHZ events (continued)

Element	Always in output	Abbreviated XPath
attributePermissionInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the attributePermissionInfoType element type.
attributePermissionInfo attributeNames	Yes	`attributePermissionInfo.attributeNames`
attributePermissionInfo checked	Yes	`attributePermissionInfo.checked`
attributePermissionInfo denied	No	`attributePermissionInfo.denied`
attributePermissionInfo granted	No	`attributePermissionInfo.granted`
attributes	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the attributeType element type.
attributes name	Yes	`attributes.name`
attributes source	No	`attributes.source`
attributes value	Yes	`attributes.value`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_AUTHZ'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
permissionInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the permissionInfoType element type.
permissionInfo checked	Yes	`permissionInfo.checked`
permissionInfo denied	No	`permissionInfo.denied`
permissionInfo granted	No	`permissionInfo.granted`
permissionInfo J2EERolesChecked	No	`permissionInfo.J2EERolesChecked`
permissionInfo J2EERolesGranted	No	`permissionInfo.J2EERolesGranted`
policyInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the policyInfoType element type.
policyInfo attributes	No	`policyInfo.attributes`

Table 15. Elements used in AUDIT_AUTHZ events (continued)

Element	Always in output	Abbreviated XPath
policyInfo branch	No	`policyInfo.branch`
policyInfo description	Yes	`policyInfo.description`
policyInfo name	Yes	`policyInfo.name`
policyInfo type	Yes	`policyInfo.type`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
resourceInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`

Table 15. Elements used in AUDIT_AUTHZ events (continued)		
Element	**Always in output**	**Abbreviated XPath**
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_COMPLIANCE events

This event type records whether a specified security policy was being complied with.

The following table lists the elements that can be displayed in the output of an AUDIT_COMPLIANCE event and their abbreviated XPath statements.

Table 16. Elements used in AUDIT_COMPLIANCE events		
Element	**Always in output**	**Abbreviated XPath**
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
complianceStatus	Yes	`complianceStatus`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_COMPLIANCE'"`
fixDescription	No	`fixDescription`
fixId	No	`fixId`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
message	No	`message`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`

Table 16. Elements used in AUDIT_COMPLIANCE events (continued)

Element	Always in output	Abbreviated XPath
policyDescription	No	`policyDescription`
policyName	No	`policyName`
recommendation	No	`recommendation`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
severity	No	`severity`
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
suppressed	No	`suppressed`
startTime	No	`startTime [type='dateTime']`
targetAccount	No	`targetAccount`
targetResource	No	`targetResource`
targetUser	No	`targetUser`
timestamp	Yes	`CommonBaseEvent/@creationTime`
violationClassification	No	`violationClassification`
violationDescription	No	`violationDescription`
violationName	When `complianceStatus` is `nonCompliant`	`violationName`

Elements for AUDIT_DATA_SYNC events

The event type provides information on data synchronization events.

The following table lists the elements that can be displayed in the output of an AUDIT_DATA_SYNC event and their abbreviated XPath statements.

Table 17. Elements used in AUDIT_DATA_SYNC events

Element	Always in output	Abbreviated XPath
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
extensionName	No	`endTime [type='dateTime']`
eventType	Yes	`"'AUDIT_DATA_SYNC'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
outcome registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
outcome serverLocation	Yes	`registryInfo.serverLocation`
outcome serverLocationType	Yes	`registryInfo.serverLocationType`
outcome serverPort	Yes	`registryInfo.serverPort`
outcome type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
resourceInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/@application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/@component`

Table 17. Elements used in AUDIT_DATA_SYNC events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/@componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/@componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/@executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/@instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/@locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/@processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/@subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_MGMT_CONFIG events

This event type identifies configuration and other management events for a server.

The following table lists the elements that can be displayed in the output of an AUDIT_MGMT_CONFIG event and their abbreviated XPath statements.

Table 18. Elements used in AUDIT_MGMT_CONFIG events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.

Table 18. Elements used in AUDIT_MGMT_CONFIG events (continued)

Element	Always in output	Abbreviated XPath
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_MGMT_CONFIG'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
mgmtInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the mgmtInfoType element type.
mgmtInfo command	No	`mgmtInfo.command`
mgmtInfo targetInfo	No	`mgmtInfo.targetInfo`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`

Table 18. Elements used in AUDIT_MGMT_CONFIG events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
type	Yes	`type`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_MGMT_POLICY events

This event type identifies the security policy management events, such as creation of access control lists.

The following table lists the elements that can be displayed in the output of an AUDIT_MGMT_POLICY event and their abbreviated XPath statements.

Table 19. Elements used in AUDIT_MGMT_POLICY events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_MGMT_POLICY'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
memberships	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the membershipInfoType element type.
memberships id	No	`memberships.id`

Table 19. Elements used in AUDIT_MGMT_POLICY events (continued)

Element	Always in output	Abbreviated XPath
memberships name	No	`memberships.name`
memberships type	Yes	`memberships.type`
mgmtInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the mgmtInfoType element type.
mgmtInfo command	No	`mgmtInfo.command`
mgmtInfo targetInfo	No	`mgmtInfo.targetInfo`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
policyInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the policyInfoType element type.
policyInfo attributes	No	`policyInfo.attributes`
policyInfo branch	No	`policyInfo.branch`
policyInfo description	Yes	`policyInfo.description`
policyInfo name	Yes	`policyInfo.name`
policyInfo type	Yes	`policyInfo.type`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
resourceInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.

Table 19. Elements used in AUDIT_MGMT_POLICY events (continued)

Element	Always in output	Abbreviated XPath
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/@application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/@component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/@componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/@componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/@executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/@instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/@locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/@processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/@subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_MGMT_PROVISIONING events

This event type identifies provisioning events, such as creating an account for a user on a specific machine.

The following table lists the elements that can be displayed in the output of an AUDIT_MGMT_PROVISIONING event and their abbreviated XPath statements.

Table 20. Elements used in AUDIT_MGMT_PROVISIONING events

Element	Always in output	Abbreviated XPath
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_MGMT_PROVISIONING'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
provisioningInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the provisioningInfoType element type.
provisioningInfo accountId	No	`provisioningInfo.accountId`
provisioningInfo resourceId	Yes	`provisioningInfo.resourceId`
provisioningInfo resourceType	Yes	`provisioningInfo.resourceType`
registryInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.

Table 20. Elements used in AUDIT_MGMT_PROVISIONING events (continued)

Element	Always in output	Abbreviated XPath
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
targetUserInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryObjectInfoType element type.
targetUserInfo attributes	No	`registryObjectInfo.attributes`
targetUserInfo description	No	`registryObjectInfo.description`
targetUserInfo name	Yes	`registryObjectInfo.name`
targetUserInfo registryName	No	`registryObjectInfo.registryName`
targetUserInfo type	Yes	`registryObjectInfo.type`
targetUserRegistryInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the targetUserRegistryInfoType element type.
targetUserRegistryInfo serverLocation	Yes	`registryInfo.serverLocation`
targetUserRegistryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
targetUserRegistryInfo serverPort	Yes	`registryInfo.serverPort`
targetUserRegistryInfo type	Yes	`registryInfo.type`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`

Table 20. Elements used in AUDIT_MGMT_PROVISIONING events (continued)		
Element	**Always in output**	**Abbreviated XPath**
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_MGMT_REGISTRY events

This event type identifies registry management events, such as creating users and groups, changing passwords by the administrator, and changing the properties for users and groups.

The following table lists the elements that can be displayed in the output of an AUDIT_MGMT_REGISTRY event and their abbreviated XPath statements.

Table 21. Elements used in AUDIT_MGMT_REGISTRY events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_MGMT_REGISTRY'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
mgmtInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the mgmtInfoType element type.
mgmtInfo command	No	`mgmtInfo.command`
mgmtInfo targetInfo	No	`mgmtInfo.targetInfo`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`

Table 21. Elements used in AUDIT_MGMT_REGISTRY events (continued)

Element	Always in output	Abbreviated XPath
registryObjectInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryObjectInfoType element type.
registryObjectInfo attributes	No	`registryObjectInfo.attributes`
registryObjectInfo description	No	`registryObjectInfo.description`
registryObjectInfo name	Yes	`registryObjectInfo.name`
registryObjectInfo registryName	No	`registryObjectInfo.registryName`
registryObjectInfo type	Yes	`registryObjectInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/@application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/@component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/@componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/@componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/@executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/@instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/@locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/@processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/@subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`

Table 21. Elements used in AUDIT_MGMT_REGISTRY events (continued)		
Element	**Always in output**	**Abbreviated XPath**
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_MGMT_RESOURCE events

This event type identifies resource management events.

The following table lists the elements that can be displayed in the output of an AUDIT_MGMT_RESOURCE event and their abbreviated XPath statements.

Table 22. Elements used in AUDIT_MGMT_RESOURCE events		
Element	**Always in output**	**Abbreviated XPath**
Action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_MGMT_RESOURCE'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
mgmtInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the mgmtInfoType element type.
mgmtInfo command	No	`mgmtInfo.command`
mgmtInfo targetInfo	No	`mgmtInfo.targetInfo`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`

Table 22. Elements used in AUDIT_MGMT_RESOURCE events (continued)

Element	Always in output	Abbreviated XPath
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryObjectInfoType element type.
registryInfo attributes	No	`registryObjectInfo.attributes`
registryInfo description	No	`registryObjectInfo.description`
registryInfo name	Yes	`registryObjectInfo.name`
registryInfo registryName	No	`registryObjectInfo.registryName`
registryInfo type	Yes	`registryObjectInfo.type`
resourceInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`

Table 22. Elements used in AUDIT_MGMT_RESOURCE events (continued)		
Element	**Always in output**	**Abbreviated XPath**
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_PASSWORD_CHANGE events

This event type identifies password changes initiated by the user.

The following table lists the elements that can be displayed in the output of an AUDIT_PASSWORD_CHANGE event and their abbreviated XPath statements.

Table 23. Elements used in AUDIT_PASSWORD_CHANGE events		
Element	**Always in output**	**Abbreviated XPath**
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_PASSWORD_CHANGE'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
provisioningInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the provisioningInfoType element type.
provisioningInfo accountId	No	`provisioningInfo.accountId`
provisioningInfo resourceId	Yes	`provisioningInfo.resourceId`
provisioningInfo resourceType	Yes	`provisioningInfo.resourceType`

Table 23. Elements used in AUDIT_PASSWORD_CHANGE events (continued)

Element	Always in output	Abbreviated XPath
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`

Table 23. Elements used in AUDIT_PASSWORD_CHANGE events (continued)		
Element	**Always in output**	**Abbreviated XPath**
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_RESOURCE_ACCESS events

This event type identifies all accesses to a resource, such as a file or HTTP request or response events outside of the AUDIT_AUTHZ events.

The following table lists the elements that can be displayed in the output of an AUDIT_RESOURCE_ACCESS event and their abbreviated XPath statements.

Table 24. Elements used in AUDIT_RESOURCE_ACCESS events		
Element	**Always in output**	**Abbreviated XPath**
accessDecision	No	`accessDecision`
accessDecisionReason	When `accessDecision` is `Denied`	`accessDecisionReason`
action	Yes	`action`
appName	No	`appName`
attributePermissionInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the attributePermissionInfoType element type.
attributePermissionInfo attributeNames	Yes	`attributePermissionInfo.attributeNames`
attributePermissionInfo checked	Yes	`attributePermissionInfo.checked`
attributePermissionInfo denied	No	`attributePermissionInfo.denied`
attributePermissionInfo granted	No	`attributePermissionInfo.granted`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_RESOURCE_ACCESS'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
httpURLInfo	When `action` is `HTTPRequest`	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the HTTPURLInfoType element type.
httpURLInfo method	No	`HTTPURLInfo.method`
httpURLInfo requestHeaders		`HTTPURLInfo.requestHeaders`
httpURLInfo responseCode		`HTTPURLInfo.responseCode`
httpURLInfo responseHeaders		`HTTPURLInfo.responseHeaders`
httpURLInfo url		`HTTPURLInfo.url`

Table 24. Elements used in AUDIT_RESOURCE_ACCESS events (continued)

Element	Always in output	Abbreviated XPath
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
permissionInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the permissionInfoType element type.
permissionInfo checked	Yes	`permissionInfo.checked`
permissionInfo denied	No	`permissionInfo.denied`
permissionInfo granted	No	`permissionInfo.granted`
permissionInfo J2EERolesChecked	No	`permissionInfo.J2EERolesChecked`
permissionInfo J2EERolesGranted	No	`permissionInfo.J2EERolesGranted`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from the sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
resourceInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`

Table 24. Elements used in AUDIT_RESOURCE_ACCESS events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_RUNTIME events

This event type identifies runtime events, such as starting, stopping, and capacity planning-related events for security servers. This event type is not meant for administrative operations performed by a system administrator. Such operations need to use the AUDIT_MGMT_* event types.

The following table lists the elements that can be displayed in the output of an AUDIT_RUNTIME event and their abbreviated XPath statements.

Table 25. Elements used in AUDIT_RUNTIME events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`

Table 25. Elements used in AUDIT_RUNTIME events (continued)

Element	Always in output	Abbreviated XPath
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_RUNTIME'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
perfInfo	When `action` is `statistic`	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the perfInfoType element type.
perfInfo aggregate	Yes	`perfInfo.aggregate`
perfInfo description	Yes	`perfInfo.description`
perfInfo name	Yes	`perfInfo.name`
perfInfo maxValue	No	`perfInfo.maxValue`
perfInfo minValue	No	`perfInfo.minValue`
perfInfo numDataPoints	Yes	`perfInfo.numDataPoints`
perfInfo unit	Yes	`perfInfo.unit`
perfInfo value	Yes	`perfInfo.value`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from the sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
resourceInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the resourceInfoType element type.
resourceInfo attributes	No	`resourceInfo.attributes`
resourceInfo nameInApp	Yes	`resourceInfo.nameInApp`
resourceInfo nameInPolicy	Yes	`resourceInfo.nameInPolicy`
resourceInfo type	Yes	`resourceInfo.type`
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.

Table 25. Elements used in AUDIT_RUNTIME events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_RUNTIME_KEY events

This event type identifies certificate expiration and expiration check events that occur during runtime.

The following table lists the elements that can be displayed in the output of an AUDIT_RUNTIME_KEY event and their abbreviated XPath statements.

Table 26. Elements used in AUDIT_RUNTIME_KEY events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`

Table 26. Elements used in AUDIT_RUNTIME_KEY events (continued)

Element	Always in output	Abbreviated XPath
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_RUNTIME_KEY'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
keyLabel	Yes	`keyLabel`
lifetime	No	`lifetime`
location	Yes	`location`
locationType	Yes	`locationType`
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`
reporterComponentId	When different from the sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to `#RECORD_ID`.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`

Table 26. Elements used in AUDIT_RUNTIME_KEY events (continued)		
Element	**Always in output**	**Abbreviated XPath**
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`

Elements for AUDIT_WORKFLOW events

This event type identifies workflow events.

The following table lists the elements that can be displayed in the output of an AUDIT_WORKFLOW event and their abbreviated XPath statements.

Table 27. Elements used in AUDIT_WORKFLOW events		
Element	**Always in output**	**Abbreviated XPath**
action	Yes	`action`
auditMsg	No	`auditMsg`
auditMsgElement	No	Neither this element, nor its children, should be defined in the shredder configuration file.
auditTrailId	No	`auditTrailId`
authenticators	No	This element is a container element and has no valid XPath. A valid XPath requires a values declaration.
authenticator	No	This element is a container element and has no valid XPath. A valid XPath requires a values declaration.
authenticator id	No	`authenticators.authenticator.id`
authenticator oauth grant	No	`authenticators.authenticator.oauthGrant`

Table 27. Elements used in AUDIT_WORKFLOW events (continued)

Element	Always in output	Abbreviated XPath
authenticator enabled	No	`authenticators.authenticator.enabled`
authenticator OS version	No	`authenticators.authenticator.osVersion`
authenticator device type	No	`authenticators.authenticator.deviceType`
authenticator device name	No	`authenticators.authenticator.deviceName`
authenticator methods	No	This element is a container element and has no valid XPath. A valid XPath requires a values declaration.
authenticator method	No	This element is a container element and has no valid XPath. A valid XPath requires a values declaration.
authentication method id	No	`authenticators.authenticator.authMethods.authMethod.id` or `authMethods.authMethod.id`
authentication method type	No	`authenticators.authenticator.authMethods.authMethod.type` or `authMethods.authMethod.type`
authentication method enabled	No	`authenticators.authenticator.authMethods.authMethod.enabled` or `authMethods.authMethod.enabled`
authentication method algorithm	No	`authenticators.authenticator.authMethods.authMethod.algorithm` or `authMethods.authMethod.algorithm`
authentication method public key	No	`authenticators.authenticator.authMethods.authMethod.publicKey` or `authMethods.authMethod.publicKey`
authentication method key handle	No	`authenticators.authenticator.authMethods.authMethod.keyHandle` or `authMethods.authMethod.keyHandle`
endTime	No	`endTime [type='dateTime']`
extensionName	Yes	`"'AUDIT_WORKFLOW'"`
globalInstanceId	Yes	Not applicable. This value is an internal number that is not related to `#GLOBAL_ID`.
outcome	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditOutcomeType element type.
outcome failureReason	No	`outcome.failureReason`
outcome majorStatus	No	`outcome.majorStatus`
outcome minorStatus	No	`outcome.minorStatus`
outcome result	Yes	`outcome.result`
registryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
registryInfo serverLocation	Yes	`registryInfo.serverLocation`
registryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
registryInfo serverPort	Yes	`registryInfo.serverPort`
registryInfo type	Yes	`registryInfo.type`

Table 27. Elements used in AUDIT_WORKFLOW events (continued)

Element	Always in output	Abbreviated XPath
reporterComponentId	When different from sourceComponentId	Neither this element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code. This container element uses the children of the auditComponentIdType element type.
sequenceNumber	Yes	Not applicable. This value is an internal number that is not related to #RECORD_ID.
sourceComponentId	Yes	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the auditComponentIdType element type.
sourceComponentId application	Yes	`CommonBaseEvent/SourceComponentId/ @application`
sourceComponentId component	Yes	`CommonBaseEvent/SourceComponentId/ @component`
sourceComponentId componentIdType	Yes	`CommonBaseEvent/SourceComponentId/ @componentIdType`
sourceComponentId componentType	Yes	`CommonBaseEvent/SourceComponentId/ @componentType`
sourceComponentId executionEnvironment	No	`CommonBaseEvent/SourceComponentId/ @executionEnvironment`
sourceComponentId instanceId	No	`CommonBaseEvent/SourceComponentId/ @instanceId`
sourceComponentId location	Yes	`CommonBaseEvent/SourceComponentId/@location`
sourceComponentId locationType	Yes	`CommonBaseEvent/SourceComponentId/ @locationType`
sourceComponentId processed	No	`CommonBaseEvent/SourceComponentId/ @processed`
sourceComponentId subComponent	Yes	`CommonBaseEvent/SourceComponentId/ @subComponent`
sourceComponentId threadId	No	`CommonBaseEvent/SourceComponentId/@threadId`
startTime	No	`startTime [type='dateTime']`
targetUserInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
targetUserInfo appUserName	Yes	`userInfo.appUserName`
targetUserInfo attributes	No	`userInfo.attributes`
targetUserInfo callerList	No	`userInfo.callerList`
targetUserInfo domain	No	`userInfo.domain`
targetUserInfo location	No	`userInfo.location`
targetUserInfo locationType	No	`userInfo.locationType`
targetUserInfo realm	No	`userInfo.realm`

Table 27. Elements used in AUDIT_WORKFLOW events (continued)

Element	Always in output	Abbreviated XPath
targetUserInfo registryUserName	Yes	`userInfo.registryUserName`
targetUserInfo sessionId	No	`userInfo.sessionId`
targetUserInfo uniqueId	No	`userInfo.uniqueId`
targetUserRegistryInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the registryInfoType element type.
targetUserRegistryInfo serverLocation	Yes	`registryInfo.serverLocation`
targetUserRegistryInfo serverLocationType	Yes	`registryInfo.serverLocationType`
targetUserRegistryInfo serverPort	Yes	`registryInfo.serverPort`
targetUserRegistryInfo type	Yes	`registryInfo.type`
timestamp	Yes	`CommonBaseEvent/@creationTime`
userInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the userInfoType element type.
userInfo appUserName	Yes	`userInfo.appUserName`
userInfo attributes	No	`userInfo.attributes`
userInfo callerList	No	`userInfo.callerList`
userInfo domain	No	`userInfo.domain`
userInfo location	No	`userInfo.location`
userInfo locationType	No	`userInfo.locationType`
userInfo realm	No	`userInfo.realm`
userInfo registryUserName	Yes	`userInfo.registryUserName`
userInfo sessionId	No	`userInfo.sessionId`
userInfo uniqueId	No	`userInfo.uniqueId`
userInputs	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the attributeType element type.
userInputs name	Yes	`attributeType.name`
userInputs source	No	`attributeType.source`
userInputs value	Yes	`attributeType.value`
workItemInfo	No	This element is a container element and has no valid XPath. A valid XPath requires a `values` declaration. This container element uses the children of the workItemInfoType element type.
workItemInfo id	Yes	`workItemInfoType.id`
workItemInfo type	Yes	`workItemInfoType.type`

Reference information about elements and element types

This section defines the various elements and element types that are available for the common audit event types.

For each element and element type that can be used in an audit event, this documentation provides a description, the values that can be displayed in the output, and the XPath statement that can be used when modifying the shredder configuration file.

For information on the elements and element types described in this section, refer to the Common Base Event specification at the following Web site: http://www.eclipse.org/tptp/platform/documents/index.php

accessDecision element

Reference information about the accessDecision element.

Description

Decision of the authorization call.

Values

String

The following strings are suggested values:

denied
Access was denied.

permitted
Access was permitted.

permittedWarning
Access was permitted in warning mode.

unknown
Cannot determine whether access is denied or not. Might be due to a non-access error (configuration problem or internal problem) or because more access decision information is needed.

XPath

```
CommonBaseEvent/extendedDataElements[@name='accessDecision']/values
```

accessDecisionReason element

Reference information about the accessDecisionReason element.

Description

Additional information about the access decision.

For example, when `accessDecision='denied'`, provides the reason for the denial.

Values

String

The following strings are suggested values:

authnLevelUnauthorized
The user is not authenticated at a sufficiently high level to access the resource.

authzRuleUnauthorized
The authorization rule policy denied access.

delegateUnauthorized
Delegate principal is unauthorized to perform delegation.

qopUnauthorized
The communication channel that is used to access the resource has an insufficient level of quality of protection.

reauthnUnauthorized
Access is denied until the user interactively reauthenticates.

timeOfDayUnauthorized
Access denied due to time of day policy.

unauthorized
Operation is not authorized. Use this value only if you cannot provide a more specific reason.

XPath

```
CommonBaseEvent/extendedDataElements[@name='accessDecisionReason']/values
```

action element

Reference information about the action element.

Description

The action that is performed.

Values

String

- For the AUDIT_AUTHN event type, the following strings are suggested values:

  authentication
  An authentication operation. Multiple authentications can occur as part of a single login.

  credsRefresh
  Refresh of a credential. For example, in the case of Kerberos.

  login
  A login operation.

  reauthentication
  Reauthentication operation.

  stepUp
  Step-up authentication.

  tokenIssue
  Used when the Trust Service issues a token on behalf of an identity.

  tokenReceipt
  Used when an incoming security token is validated by the Trust Service.

  switchUser
  A switch user operation.

- For the AUDIT_AUTHN_CREDS_MODIFY event type, the following strings are suggested values:

  credsCombine
  Caller is adding a user to a credential chain.

  credsModify
  Caller is creating a modified copy of existing user credentials.

  getCreds
  Caller is getting credentials based on user information.

getCredsFromPAC
Resolve credentials from transferable object (privilege attribute certificate [PAC]).

getEntitlements
Add to credentials by using an entitlements service.

getPAC
Convert credentials to a transferable object (privilege attribute certificate [PAC]).

- For the AUDIT_AUTHN_TERMINATE event type, the following strings are suggested values:

logout
A logout operation.

switchUserTerminate
Used when the switch user session is ended.

- For the AUDIT_DATA_SYNC event type, the following strings are suggested values:

reconcile
Reconcile accounts. For example, the Identity Manager server might send a request to the remote provisioning resource to synchronize account data into the Identity Manager repository.

unsolicitedNotification
Notify of operations. For example, the remote provisioning resource might send a notification to the Identity Manager server to notify changes on the account data.

- For the AUDIT_MGMT_CONFIG, AUDIT_MGMT_POLICY, AUDIT_MGMT_REGISTRY, and AUDIT_MGMT_RESOURCE event types, the following strings are suggested values:

associate
Associate entities. For example, the user who is associated with groups, group associated with users, and policy associated with objects.

challengeResponse
Change the challenge and response configurations.

changePolicyEnforcementAction
Change the policy enforcement action of the management object. The following list shows the allowable actions:

Correct
Suspend
Mark
Non-Compliant

checkAccess
An authorization decision was made.

create
Create a management object.

delegate
Delegate authorities the user has to another user for a specified amount of time.

delete
Delete a management object. For example, delete a file from the Trusted Computing Base.

disable
Disable an account for login activity.

disassociate
Disassociate entities. For example, disassociate a user from groups, disassociate a group from users, and disassociate a policy from objects.

enable
Enable an account for login activity.

markTrusted
Mark as trusted. For example, mark a file as trusted in the Trusted Computing Base.

markUntrusted
Mark as untrusted. For example, mark a file as untrusted in the Trusted Computing Base.

modify
Modify a management object.

passthru
Indicates that request is passed to another server.

passwordChange
Indicates a password change operation initiated by the administrator.

passwordPickup
Pick up password for account.

register
To register. For example, register a daemon with the kernel.

restore
To restore. For example, to restore a suspended user or account.

retire
To retire. For example, a federation is retired when it is no longer used. This information is archived for future reference.

retrieve
A credential was retrieved.

show
Show a management object.

suspend
To suspend. For example, suspend a partner in a federation.

transfer
Transfer a user between different organization containers.

validate
To validate. For example, verify a security token that represents a user.

- For the AUDIT_MGMT_PROVISIONING event type, the following strings are suggested values:

  add
  Provision a new account on the target resource identified by provisioningTargetInfo.

  adopt
  Adopt an orphan account identified by provisioningTargetInfo.

  changePassword
  Change password for an account identified by provisioningTargetInfo.

  delete
  Delete an account identified by provisioningTargetInfo.

  modify
  Modify an existing account identified by provisioningTargetInfo.

  passwordPickup
  Pick up password for an account identified by provisioningTargetInfo.

  restore
  Restore a suspended account identified by provisioningTargetInfo.

  suspend
  Suspend an existing account identified by provisioningTargetInfo.

- For the AUDIT_RESOURCE_ACCESS event type, the following strings are suggested values:

  fileExec
  A program execution occurred.

  fileTrace
  A file access occurred.

httpRequest
A request was made to access a resource by using HTTP.

- For the AUDIT_RUNTIME event type, the following strings are suggested values:

auditLevelChange
An audit or warning level change request is sent to the server.

auditStart
Auditing started for a server component.

auditStop
Auditing stopped for a server component.

contactRestored
Restored contact. For example, the server regained contact with the Security Verify Access user registry.

heartbeatDown
Heartbeat information that a server or API is down.

heartbeatUp
Heartbeat information that a server or API is up.

lostContact
Lost contact. For example, the server currently has no contact with the Security Verify Access user registry.

monitor
A process was adopted in to the set of monitored processes.

start
A server successfully started.

statistic
Statistical information for a server for capacity planning purposes.

stop
A server successfully stopped.

- For the AUDIT_RUNTIME_KEY event type, the following strings are suggested values:

keyRetire
The key is retired.

keyCRLInvalidated
The CRL in the key is not valid.

keyCertExpired
The certificate in the key expired.

keySetInvalid
The key is set as not valid.

keyCertExpirationCheck
The expiration of the certificate is checked.

- For the AUDIT_WORKFLOW event type, the following strings are suggested values:

assign
A work item is assigned and routed to a user.

complete
A work item is completed by the user.

defer
More time is given for the completion of the work item.

delegate
A work item is being delegated to another user.

escalate
A work item is being escalated as a result of timeout.

lock
A work item is being locked by a user. After a work item is locked, no other potential work item owner can perform the operation on the work item.

unlock
A work item is unlocked by a user.

XPath

```
CommonBaseEvent/extendedDataElements[@name='action']/values
```

appName element

Reference information about the appName element.

Description

Name of the application that is accessing the resource.

Values

String

For example, an Emacs program can be accessing a file resource.

XPath

```
CommonBaseEvent/extendedDataElements[@name='appName']/values
```

attributePermissionInfo element

Reference information about the attributePermissionInfo element.

Description

A container for the information about access permissions on the attributes of the target.

This container uses the children of attributePermissionInfoType:

- attributePermissionInfoType.attributeNames
- attributePermissionInfoType.checked
- attributePermissionInfoType.denied
- attributePermissionInfoType.granted

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

attributePermissionInfo.attributeNames element

Reference information about the attributePermissionInfo.attributeNames element.

Description

List of attributes in which permissions are being checked.

Values

String[]

XPath

The XPath accesses the first attributeNames element from an array of attributeNames elements.

```
CommonBaseEvent/extendedDataElements
[@name='attributePermissionInfo']/children[1]/children
[@name='attributeNames']/values[1]
```

attributePermissionInfo.checked element

Reference information about the attributePermissionInfo.checked element.

Description

Permission that are being checked during the authorization call.

Values

String[]

XPath

The XPath accesses the first checked element from an array of checked elements.

```
CommonBaseEvent/extendedDataElements
[@name='attributePermissionInfo']/children[1]/children
[@name='checked']/values[1]
```

attributePermissionInfo element

Reference information about the attributePermissionInfo.denied element.

Description

Permission that are denied.

Values

String[]

XPath

The XPath accesses the first denied element from an array of denied elements.

```
CommonBaseEvent/extendedDataElements
[@name='attributePermissionInfo']/children[1]/children
[@name='denied']/values[1]
```

attributePermissionInfo.granted element

Reference information about the attributePermissionInfo.granted element.

Description

Permission that are granted.

Values

String[]

XPath

The XPath accesses the first granted element from an array of granted elements.

```
CommonBaseEvent/extendedDataElements
[@name='attributePermissionInfo']/children[1]/children
[@name='granted']/values[1]
```

attributes element

Reference information about the attributes element.

Description

A container for the array of application-specific attributes for this event.

This element type represents an attribute that is associated with an entity, such as a user, application, or authorization rule.

This element uses the children of the attributeType element:

- attributes.name
- attributes.source
- attributes.value

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

attributes.name element

Reference information about the attributes.name element.

Description

Name of the attribute.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='attributes']/children[1]/children
[@name='name']/values[1]
```

attributes.source

Reference information about the attributes.source element.

Description

Source of the attribute.

Values

String

The following strings are suggested values:

application
Provided by the application.

authzRuleADI
Provided as an input for authorization rules.

user
Provided by the user.

XPath

```
CommonBaseEvent/extendedDataElements[@name='attributes']/children[1]/children
[@name='source']/values[1]
```

attributes.value

Reference information about the attributes.value element.

Description

Value of the attribute.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='attributes']/children[1]/children
[@name='value']/values[1]
```

auditMsg

Reference information about the auditMsg element.

Description

Message for this audit event.

Values

xsd:string

Any arbitrary string

Refer to the msg field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='auditMsg']/values
```

auditMsgElement

Reference information about the auditMsgElement element.

Description

Information associated with message.

This container uses the field of msgDataElement and its children. For additional details, refer to the Common Base Event specification.

Values

cbe:msgDataElement

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

auditTrailId

Reference information about the auditTrailId element.

Description

ID that allows audit events that belong to a given transaction to be correlated.

For example, this could be populated using the propagationToken in WebSphere® Application Server.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='auditTrailId']/values
```

authenProvider

Reference information about the authenProvider element.

Description

Provider of the authentication service.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='authenProvider']/values
```

authnType

Reference information about the authnType element.

Description

Provider of the authentication service.

Values

Any arbitrary string

The following strings are suggested values:

basicAuth
Browser authentication based on user ID and password.

challengeResponse
Challenge and response authentication.

digest
Digest-based authentication.

form
Form-based authentication.

identityAssertion
Authentication based on identity assertion.

kerberos
Authentication based on Kerberos credentials.

ldap_v3.0
Authentication using the LDAP protocol.

ltpa
Lightweight third-party authentication.

sslAuthn
SSL-based authentication.

tokenAccessManagerCred
Authentication based on Security Verify Access credentials.

tokenLiberty
Authentication based on a Liberty token.

tokenSAML
Authentication based on a SAML token.

tokenUserName
Authentication based on user name based token.

trustAssociation
Authentication based on trust association.

XPath

```
CommonBaseEvent/extendedDataElements[@name='authnType']/values
```

authnTypeVersion

Reference information about the authnTypeVersion element.

Description

Version of the authentication type.

Values

String form of the version number

XPath

```
CommonBaseEvent/extendedDataElements[@name='authnTypeVersion']/values
```

complianceStatus

Reference information about the complianceStatus element.

Description

Status of compliance.

Values

String

The following strings are suggested values:

compliant
The reconciled account on the provisioning resource complies with the specified security policy.

disallowed
The reconciled account is not allowed by a provisioning policy.

nonCompliant
The reconciled account on the provisioning resource does not comply with the specified security policy.

orphan
No owner can be found for the reconciled account.

XPath

```
CommonBaseEvent/extendedDataElements[@name='complianceStatus']/values
```

endTime

Reference information about the endTime element.

Description

End time of the operation.

Values

xsd:DateTime

Refer to the creationTime field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='endTime'][@type='dateTime']/values
```

extensionName

Reference information about the extensionName element.

Description

The event type.

This information relates to the following line in the `CARSShredder.conf` file:

```
cars_t_event, eventType, "'event_type'"
```

Values

String

The actual name of the event type, which is one of the following literal values:

- AUDIT_AUTHN_CREDS_MODIFY
- AUDIT_AUTHN_MAPPING
- AUDIT_AUTHN_TERMINATE
- AUDIT_AUTHN
- AUDIT_AUTHZ
- AUDIT_COMPLIANCE
- AUDIT_DATA_SYNC
- AUDIT_MGMT_CONFIG
- AUDIT_MGMT_POLICY
- AUDIT_MGMT_PROVISIONING
- AUDIT_MGMT_REGISTRY
- AUDIT_MGMT_RESOURCE
- AUDIT_PASSWORD_CHANGE
- AUDIT_RESOURCE_ACCESS
- AUDIT_RUNTIME
- AUDIT_RUNTIME_KEY
- AUDIT_WORKFLOW

XPath

event_type

For example, to specify the AUDIT_AUTHN event type, specify:

```
"'AUDIT_AUTHN'"
```

fixDescription

Reference information about the fixDescription element.

Description

Description of specific fix. For example, "Apply patch *xyz*".

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='fixDescription']/values
```

fixId

Reference information about the fixId element.

Description

Identifier of specific fix.

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='fixId']/values
```

globalInstanceId

Reference information about the globalInstanceId element.

Description

An internal identifier for an audit event as shown in the XML output.

This information is not related to the following line in the CARSShredder.conf file:

```
cars_t_event, event_id, #GLOBAL_ID
```

httpURLInfo element

Reference information about the httpURLInfo element.

Description

The container for information about the HTTP request.

This container uses the children of HTTPURLInfoType:

- HTTPURLInfoType.method
- HTTPURLInfoType.requestHeaders
- HTTPURLInfoType.responseCode
- HTTPURLInfoType.responseHeaders
- HTTPURLInfoType.url

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a values declaration.

HTTPURLInfo.method

Reference information about the HTTPURLInfo.method element.

Description

Method used.

Values

String

Methods allowed by the HTTP protocol (for example, POST or GET). The following strings are suggested values:

GET
Passed in information using the HTTP GET method.

POST
Passed in information using the HTTP POST method.

XPath

```
CommonBaseEvent/extendedDataElements[@name='HTTPURLInfo']/children
[@name='method']/values
```

HTTPURLInfo.requestHeaders

Reference information about the HTTPURLInfo.requestHeaders element.

Description

HTTP request headers given by the client.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='HTTPURLInfo']/children
[@name='requestHeaders']/values
```

HTTPURLInfo.responseCode

Reference information about the HTTPURLInfo.responseCode element.

Description

Response code returned by the server.

Values

Integer

XPath

```
CommonBaseEvent/extendedDataElements[@name='HTTPURLInfo']/children
[@name='responseCode']/values
```

HTTPURLInfo.responseHeaders

Reference information about the HTTPURLInfo.responseHeaders element.

Description

HTTP response headers returned by the server.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='HTTPURLInfo']/children
[@name='responseHeaders']/values
```

HTTPURLInfo.url element

Reference information about the HTTPURLInfo.url element.

Description

URL of the HTTP request.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='HTTPURLInfo']/children
[@name='url']/values
```

keyLabel

Reference information about the keyLabel element.

Description

Indicates the key or certificate label.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='keyLabel']/values
```

lifetime

Reference information about the lifetime element.

Description

Indicates when a certificate will expire.

Values

xsd:DateTime

Refer to the creationTime field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='lifetime']/values
```

location

Reference information about the location element.

Description

Physical location of the key database.

Values

xsd:string

Refer to the location field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='location']/values
```

locationType

Reference information about the locationType element.

Description

Type of location.

Values

xsd:Name

Refer to the locationType field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='locationType']/values
```

loginTime

Reference information about the loginTime element.

Description

The time that the login occurred.

Values

xsd:DateTime

Refer to the creationTime field in the Common Base Event specification.

XPath

```
CommonBaseEvent/@creationTime
```

mappedRealm

Reference information about the mappedRealm element.

Description

Indicate the realm after mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='mappedRealm']/values
```

mappedSecurityDomain

Reference information about the mappedSecurityDomain element.

Description

Indicate the security domain after mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='mappedSecurityDomain']/values
```

mappedUserName

Reference information about the mappedUserName element.

Description

Indicate the user name after mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='mappedUserName']/values
```

membershipInfo

Reference information about the membershipInfo element.

Description

The container for list of memberships to which the policy applies.

The element uses the children of the membershipInfo element:

- membershipInfoType.id
- membershipInfoType.name
- membershipInfoType.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

memberships.id element

Reference information about the memberships.id element.

Description

Unique identifier of the member.

Values

String

For example, distinguished name of a role.

XPath

The XPath statement assumes the first membership element from an array of membership elements.

```
CommonBaseEvent/extendedDataElements
[@name='memberships']/children[1]/children
[@name='id']/values
```

memberships.name element

Reference information about the memberships.name element.

Description

Name of the member.

Values

String

XPath

The XPath statement assumes the first membership element from an array of membership elements.

```
CommonBaseEvent/extendedDataElements
[@name='memberships']/children[1]/children
[@name='name']/values
```

memberships.type element

Reference information about the memberships.type element.

Description

Membership type.

Values

String

The following strings are suggested values:

all
Applies to all users.

orgContainer
Applies to users that belong in a given organization container.

other
Is not one of the other types.

role
Applies to users that belong in a given role.

XPath

The XPath statement assumes the first membership element from an array of membership elements.

```
CommonBaseEvent/extendedDataElements
[@name='memberships']/children[1]/children
[@name='type']/values
```

message

Reference information about the message element.

Description

Generated message that describes specifics about the violation. Can include dynamically inserted information. Example:

Invalid ACL for
c:\winnt\repair:
Account: BUILTIN\users

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='message']/values
```

mgmtInfo

Reference information about the mgmtInfo element.

Description

The container for information about this management operation.

This element type represents information that is common for events that are related to management operations, such as managing policies, resources, registry objects, and so forth.

This element uses the children of mgmtInfoType:

- mgmtInfoType.command
- mgmtInfoType.targetInfo

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

mgmtInfo.command

Reference information about the mgmtInfo.command element.

Description

The application-specific command being performed. The command is particularly useful for modify actions to pinpoint what is being modified.

Values

String

An application-specific string that represents the command. Examples:

- Key user modify:

  modifyPassword
  modifyAccountValid
  modifyPasswordValidKey

- Policy modify:

  modifyPolicyMaxLoginFailures
  modifyPolicyMaxAccountAge
  modifyPolicyMaxPasswordAge
  modifyPolicyTimeOfDayAccess

- ACL modify:

  modifyACLSetAttribute
  modifyACLDelAttribute

- POP modify:

  modifyPOPSetAttribute
  modifyPOPDelAttribute

- protectedObject modify:

  modifyObjectDelAttribute
  modifyObjectSetAttribute

XPath

```
CommonBaseEvent/extendedDataElements[@name='mgmtInfo']/children
[@name='command']/values
```

mgmtInfo.targetInfo

Reference information about the mgmtInfo.targetInfo element.

Description

Information about the target resource of this operation.

Values

targetInfoType

XPath

Refer to “targetInfoType” on page 171 for details.

originalRealm

Reference information about the originalRealm element.

Description

Indicate the realm before mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='originalRealm']/values
```

originalSecurityRealm

Reference information about the originalSecurityRealm element.

Description

Indicate the security domain before mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='originalSecurityRealm']/values
```

originalUserName

Reference information about the originalUserName element.

Description

Indicate the user name before mapping.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='originalUserName']/values
```

outcome

Reference information about the outcome element.

Description

A container for the outcome of the action for which the audit record is generated.

This element type identifies a component that is the source of the event or reports an event, and defines the outcome of the event being audited.

This element uses the children of auditOutcomeType:

- outcome.failureReason
- outcome.majorStatus
- outcome.minorStatus
- outcome.result

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

outcome.failureReason

Reference information about the outcome.failureReason element.

Description

Additional information about the outcome.

Values

Any arbitrary string.

The outcome element contains the failureReason element. The values for the failureReason elements are event-specific. The following strings are some of the suggested values:

accountDisabled
User's account has been disabled.

accountDisabledRetryViolation
Retry maximum has been violated for authentications that are not valid. The account has been disabled in the registry.

accountExpired
User account has expired.

accountLockedOutMaxLoginFail
User account has been temporarily locked out due to too many failed login attempts. Lock time interval has not elapsed.

accountLockedOutRetryViolation
Invalid authentication retry maximum has been violated. The account has been temporarily locked out.

accountMaxInactiveElapsed
Maximum inactive days has elapsed for the account.

accountUnlocked
User account was unlocked because lock time interval has elapsed.

authenticationFailure
Authentication failed. Use this value when you do not have a more specific value for this audit element.

certificateFailure
A client certificate could not be authenticated.

invalidUserName
The supplied user name does not exist in the registry.

invalidUserPassword
The password associated with the given user name is incorrect.

mappingFailure
The login data entered could not be mapped to an application-specific user.

nextToken
Next token required for authentication.

passwordChangeMaxIntervalElapsed
Maximum time interval since last password change has elapsed.

passwordChangeMinIntervalUnexpired
Minimum time interval required between password changes has not elapsed.

passwordContainOld
Password contains the old password or is contained in the old password.

passwordExpired
The user's password has expired and no further grace logins remain.

passwordFirstLastNumeric
Password contains a numeric first or last character.

passwordMaxCharOld
Password exceeds the allowed number of consecutive characters that are common with the previous password.

passwordMaxRepeated
Password exceeds the maximum allowed number of repeated characters.

passwordMinAlphabetic
Password does not contain the required minimum number of alphabetic characters.

passwordMinAlphabeticLower
Password does not contain the required minimum number of lowercase characters.

passwordMinAlphabeticUpper
Password does not contain the required minimum number of uppercase characters.

passwordMinAlphanumeric
Password does not contain the required minimum number of alphanumeric characters

passwordMinNumeric
Password does not contain the required minimum number of numeric characters.

passwordMinSpecial
Password does not contain the required minimum number of special characters.

passwordNumCharViolation
Password does not contain the required number of characters.

passwordOldReused
Password is a recently used old password.

passwordUserName
Password contains the user name or is contained in the user name.

pinRequired
A PIN must be assigned to enable account.

policyAllowedAccess
All login policy checks permitted access.

policyViolation
Login rejected due to policy violation.

policyViolationMaxLoginsReached
Login rejected because maximum number of concurrent logins reached.

policyViolationTOD
Authentication denied at this time of the day.

tokenExpired
The lifetime for the token has expired.

tokenNotSupported
The given token is not a supported type.

tokenNotInValidFormat
The given token was not in the expected format or was corrupted.

tokenNotValidYet
The token is not valid yet.

tokenSignatureValidationFailed
The signature for the token was not valid.

usernameMismatch
In the case of `reauthentication` or `stepUp` authentication, the given user name does not match the current user name.

When a suggested value is not available, use the string "Unknown Failure Reason".

XPath

```
CommonBaseEvent/extendedDataElements[@name='outcome']/children
[@name='failureReason']/values
```

outcome.majorStatus

Reference information about the outcome.majorStatus element.

Description

Major status code. Typically, majorStatus will be zero when result is SUCCESSFUL, and some nonzero value when it is not.

Values

Any integer

XPath

```
CommonBaseEvent/extendedDataElements[@name='outcome']/children
[@name='majorStatus']/values
```

outcome.minorStatus

Reference information about the outcome.minorStatus element.

Description

Minor status code. Typically, minorStatus will be zero when result is SUCCESSFUL, and some non-zero value when it is not.

Values

Any integer

XPath

```
CommonBaseEvent/extendedDataElements[@name='outcome']/children
[@name='minorStatus']/values
```

outcome.result

Reference information about the outcome.result element.

Description

Overall status of the event commonly used for filtering. Use UNSUCCESSFUL when an error condition arose which prevented normal processing, and SUCCESSFUL for normal processing.

Values

Same as the successDisposition field in the Situation types in the Common Base Event specification.

- SUCCESSFUL
- UNSUCCESSFUL

XPath

```
CommonBaseEvent/extendedDataElements[@name='outcome']/children
[@name='result']/values
```

partner

Reference information about the partner element.

Description

End time of the operation.

Values

xsd:DateTime

XPath

```
CommonBaseEvent/extendedDataElements[@name='partner']/values
```

perfInfo

Reference information about the perfInfo element.

Description

A container that represents performance and statistical data This information that can be helpful during capacity planning activities.

This element uses the children of perfInfoType:

- perfInfo.aggregate
- perfInfo.description
- perfInfo.name
- perfInfo.maxValue

- perfInfo.minValue
- perfInfo.numDataPoints
- perfInfo.unit
- perfInfo.value

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

perfInfo.aggregate

Reference information about the perfInfo.aggregate element.

Description

Operation for combining with other statistic events.

Values

String

The following strings are suggested values:

addition
When combining with another statistic that measures the same data, then the values of the data should be added together.

average
When combining with another statistic that measures the same data, then the values of the data should be averaged.

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='aggregate']/values
```

perfInfo.description

Reference information about the perfInfo.description element.

Description

Description of the statistic.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='description']/values
```

perfInfo.name element

Reference information about the perfInfo.name element.

Description

Name of the statistic.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='name']/values
```

perfInfo.maxValue

Reference information about the perfInfo.maxValue element.

Description

Maximum value among all data points.

Values

Long

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='maxValue']/values
```

perfInfo.minValue

Reference information about the perfInfo.minValue element.

Description

Minimum value among all data points.

Values

Long

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='minValue']/values
```

perfInfo.numDataPoints

Reference information about the perfInfo.numDataPoints element.

Description

Number of data points gathered.

Values

Integer

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='numDataPoints']/values
```

perfInfo.unit element

Reference information about the perfInfo.unit element.

Description

Unit of measurement for the value.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='unit']/values
```

perfInfo.value

Reference information about the perfInfo.value element.

Description

Value of the statistic.

Values

Long

XPath

```
CommonBaseEvent/extendedDataElements[@name='perfInfo']/children
[@name='value']/values
```

permissionInfo

Reference information about the permissionInfo element.

Description

A container represents information about access permissions.

This element uses the children of permissionInfoType:

- permissionInfoType.checked
- permissionInfoType.denied
- permissionInfoType.granted
- permissionInfoType.J2EERolesChecked

- permissionInfoType.J2EERolesGranted

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

permissionInfo.checked

Reference information about the permissionInfo.checked element.

Description

Permission that are being checked during the authorization call.

Values

String[]

Any arbitrary string allowed by the application can be provided as an element of the String[].

XPath

The XPath accesses the first checked element from an array of checked elements.

```
CommonBaseEvent/extendedDataElements[@name='permissionInfo']/children
[@name='checked']/values[1]
```

permissionInfo.denied

Reference information about the permissionInfo.denied element.

Description

Permissions that are denied out of the ones requested.

Values

String[]

Any arbitrary string allowed by the application can be provided as an element of the String[].

XPath

The XPath accesses the first denied element from an array of denied elements.

```
CommonBaseEvent/extendedDataElements[@name='permissionInfo']/children
[@name='denied']/values[1]
```

permissionInfo.granted

Reference information about the permissionInfo.granted element.

Description

Permissions that are granted.

Values

String[]

Any arbitrary string allowed by the application can be provided as an element of the String[].

XPath
The XPath accesses the first granted element from an array of granted elements.

```
CommonBaseEvent/extendedDataElements[@name='permissionInfo']/children
[@name='granted']/values[1]
```

permissionInfo.J2EERolesChecked

Reference information about the permissionInfo.J2EERolesChecked element.

Description

J2EE roles being checked.

Values

String[]

Any arbitrary string allowed by the application can be provided as an element of the String[].

XPath
The XPath accesses the first J2EERolesChecked element from an array of J2EERolesChecked elements.

```
CommonBaseEvent/extendedDataElements[@name='permissionInfo']/children
[@name='J2EERolesChecked']/values[1]
```

permissionInfo.J2EERolesGranted

Reference information about the permissionInfo.J2EERolesGranted element.

Description

J2EE roles granted.

Values

String[]

Any arbitrary string allowed by the application can be provided as an element of the String[].

XPath
The XPath accesses the first J2EERolesGranted element from an array of J2EERolesGranted elements.

```
CommonBaseEvent/extendedDataElements[@name='permissionInfo']/children
[@name='J2EERolesGranted']/values[1]
```

policyDescription

Reference information about the policyDescription element.

Description

Description of the policy that contains violation specification.

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyDescription']/values
```

policyInfo

Reference information about the policyInfo element.

Description

A container for information about the policy object, which can includes policies that are attached to the resource or policies that are the container of a resource.

This element type represents a policy associated with an authorization resource or policy management event.

The element uses the children of policyInfoType:

- policyInfo.attributes
- policyInfo.branch
- policyInfo.description
- policyInfo.name
- policyInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

policyInfo.attributes

Reference information about the policyInfo.attributes element.

Description

Attributes associated with a policy.

Values

attributeType[]

See “attributes element” on page 128 for details.

XPath

The XPath accesses the first source element from an array of attributes elements.

```
CommonBaseEvent/extendedDataElements
[@name='policyInfo']/children[5]/children
[@name='source']/values
```

Note: The index is 5, for the `attributes` element must come after the`branch`, `description`, `name`, and `type` elements:

policyInfo.branch

Reference information about the policyInfo.branch element.

Description

Name of the branch to which the policy applies.

Values

String

For example: The product lets you group the policy for similar machines under user-defined policy branches.

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyInfo']/children
[@name='branch']/values
```

policyInfo.description

Reference information about the policyInfo.description element.

Description

Description of the policy.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyInfo']/children
[@name='description']/values
```

policyInfo.name element

Reference information about the policyInfo.name element.

Description

Name of the policy.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyInfo']/children
[@name='name']/values
```

policyInfo.type element

Reference information about the policyInfo.type element.

Description

Type of the policy.

Values

String

The following strings are suggested values:

accountPolicy
Account policy:

- Account expiry date
- Maximum account age
- Time of day (TOD) access

acl
Access control list.

action
Represents a permission.

actionGroup
Represents a collection of permissions.

authzRule
Authorization rule.

federation
A collection of groups or organizations that participate in a trust relationship.

identityPolicy
Specifies how identities, or user IDs, are generated when provisioning one or more resources.

key
A cryptographic key, either symmetric or asymmetric.

loginPolicy
Policy that controls login behavior:

- Login failure count
- Login disable time interval

partner
A group or organization that is participating in a federation.

passwordPolicy
A set of rules in which all passwords for one or more services must conform.

policy
Generic policy value to be used for policies that are not defined in the other values.

pop
Protected object policy (POP) controls.

- Audit level
- Additional attributes
- Quality of protection (QoP)

provisioningPolicy
Used to associate one or multiple groups of users with one or multiple entitlements. The group of users can be identified by organization or organization role. The entitlement is a construct to define a set of permissions, or privileges, on a managed provisioning resource.

serviceSelectionPolicy
Used in situations where the instance of a provisioning resource, on which the provisioning of an account is to take place, is determined dynamically based on account owner's attributes.

spsModule
A Single Sign-On (SSO) Protocol Service module (for example, the Liberty module).

stsChain
A grouping of Security Token Service (STS) module instances.

stsModule
Security Token Service (STS) module (for example, SAML module).

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyInfo']/children
[@name='type']/values
```

policyName

Reference information about the policyName element.

Description

Name of policy. Example: "ITCS104AIX".

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='policyName']/values
```

progName

Reference information about the progName element.

Description

Name of the program that is involved in the event.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='progName']/values
```

provisioningInfo

Reference information about the provisioningInfo element.

Description

A container for the information about a provisioned resource that is the target of the operation.

This element uses the children of provisioningInfoType:

- provisioningInfoType.accountId
- provisioningInfoType.resourceId
- provisioningInfoType.resourceType

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

provisioningInfo.accountId

Reference information about the provisioningInfo.accountId element.

Description

Unique identifier of the target account.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='provisioningInfo']/children
[@name='accountId']/values
```

provisioningInfo.resourceId

Reference information about the provisioningInfo.resourceId element.

Description

Unique identifier of the target resource.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='provisioningInfo']/children
[@name='resourceId']/values
```

provisioningInfo.resourceType

Reference information about the provisioningInfo.resourceType element.

Description

Type of the target. For example, the type of the user, or the type of the provisioning resource.

Values

An arbitrary string.

See suggested values for “resourceInfo.type element” on page 163 audit element.

XPath

```
CommonBaseEvent/extendedDataElements[@name='provisioningInfo']/children
[@name='resourceType']/values
```

provisioningTargetInfo

Reference information about the provisioningTargetInfo element.

Description

A container for target provisioning account.

This element uses the children of provisioningInfoType:

- provisioningInfoType.accountId
- provisioningInfoType.resourceId
- provisioningInfoType.resourceType

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

recommendation

Reference information about the recommendation element.

Description

Provides information related to remedial actions to take to protect against the vulnerability.

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='recommendation']/values
```

registryInfo

Reference information about the registryInfo element.

Description

A container for information about the user registry that is involved in the operation.

This element uses the children of the registryInfoType element:

- registryInfo.serverLocation
- registryInfo.serverLocationType
- registryInfo.serverPort
- registryInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

registryInfo.serverLocation

Reference information about the registryInfo.serverLocation element.

Description

Location of the registry server.

Values

xsd:string

Refer to the location field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryInfo']/children
[@name='serverLocation']/values
```

registryInfo.serverLocationType

Reference information about the registryInfo.serverLocationType element.

Description

Type of server location.

Values

xsd:Name

Refer to the locationType field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryInfo']/children
[@name='serverLocationType']/values
```

registryInfo.serverPort

Reference information about the registryInfo.serverPort element.

Description

Port on which the registry server is listening.

Values

String

Port number

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryInfo']/children
[@name='serverPort']/values
```

registryInfo.type element

Reference information about the registryInfo.type element.

Description

Type of registry.

Values

String

The following strings are suggested values:

ActiveDir
Active Directory registry.

AIX
AIX user registry.

LDAP
LDAP registry.

Linux
Linux user registry.

Solaris
Solaris user registry.

Windows
Windows user registry.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryInfo']/children
[@name='type']/values
```

registryObjectInfo

Reference information about the registryObjectInfo element.

Description

A container for information about the registry object that is being managed.

This container uses the children of the registryObjectInfoType element:

- registryObjectInfo.attributes
- registryObjectInfo.description
- registryObjectInfo.name
- registryObjectInfo.registryName
- registryObjectInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

registryObjectInfo.attributes

Reference information about the registryObjectInfo.attributes element.

Description

Attributes associated with a registry object.

Values

attributeType[]

See “attributes element” on page 128 for details.

XPath

The XPath accesses the first name element from an array of attributes elements.

```
CommonBaseEvent/extendedDataElements
[@name='registryObjectInfo']/children[5]
[@name='name']/values
```

Note: The index is 5, for the `attributes` element must come after the`description`, `name`, `registryName`, and `type` elements:

registryObjectInfo.description

Reference information about the registryObjectInfo.description element.

Description

Description of the policy.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryObjectInfo']/children
[@name='description']/values
```

registryObjectInfo.name element

Reference information about the registryObjectInfo.name element.

Description

Application name for the registry object.

Values

String

Any string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryObjectInfo']/children
[@name='name']/values
```

registryObjectInfo.registryName

Reference information about the registryObjectInfo.registryName element.

Description

Registry name for the registry object.

Values

String

Any string allowed by the registry.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryObjectInfo']/children
[@name='registryName']/values
```

registryObjectInfo.type element

Reference information about the registryObjectInfo.type element.

Description

Type of the registry object.

Values

String

The following strings are suggested values:

domain
A registry object that represents a domain.

group
A registry object that represents a group.

gsoResource
A registry object that represents a global sign-on (GSO) resource.

orgContainer
Identifies the organization hierarchy for the user.

user
A registry object that represents a user.

XPath

```
CommonBaseEvent/extendedDataElements[@name='registryObjectInfo']/children
[@name='type']/values
```

reporterComponentId

Reference information about the reporterComponentId element.

Description

A container for the reporter of the audit record on behalf of the source component. This container element is used when the reporting component is different from the source component.

When displayed in output, this element uses the children of the auditComponentIdType element:

- application
- component
- componentIdType
- componentType
- executionEnvironment
- instanceId
- location
- locationType
- processed
- subcomponent
- threadId

XPath

This element, nor its children, should be defined in the shredder configuration file. These elements are generated by the code.

resourceInfo

Reference information about the resourceInfo element.

Description

The container for information about the resource that is being accessed or that to which the policy applies.

This element uses the children of the resourceInfoType element:

- resourceInfo.attributes
- resourceInfo.nameInApp
- resourceInfo.nameInPolicy
- resourceInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

resourceInfo.attributes

Reference information about the resourceInfo.attributes element.

Description

Array of attributes for the resource.

Values

attributeType []

Refer to “attributes element” on page 128 for details.

XPath

The XPath accesses the first name element from an array of attributes elements.

```
CommonBaseEvent/extendedDataElements
[@name='registryObjectInfo']/children[4]
[@name='name']/values
```

Note: The index is 4, for the `attributes` element must come after the`nameInApp`, `nameInPolicy`, and `type` elements:

```
CommonBaseEvent/extendedDataElements[@name='resourceInfo']/children
[@name='attributes']/values
```

resourceInfo.nameInApp

Reference information about the resourceInfo.nameInApp element.

Description

Name of the resource in the context of the application.

Values

Any arbitrary string

User "Not Available" when not available.

XPath

```
CommonBaseEvent/extendedDataElements[@name='resourceInfo']/children
[@name='nameInApp']/values
```

resourceInfo.nameInPolicy

Reference information about the resourceInfo.nameInPolicy element.

Description

Name of the resource when applying a policy to it. For example, Security Verify Access protected object name.

Values

Any arbitrary string

User "Not Available" when not available.

XPath

```
CommonBaseEvent/extendedDataElements[@name='resourceInfo']/children
[@name='nameInPolicy']/values
```

resourceInfo.type element

Reference information about the resourceInfo.type element.

Description

Type of the resource.

Values

String

The following strings are suggested values:

application
An application such as Security Verify Access server, Directory Server, Identity Manager server, or any executable process.

file
File system resource. For example, `/OSSEAL/policy-branch/File/filespec`.

group
Used to group users for Role Based Access Control.

identityPolicy
Identify policy specifies how user identities are generated when provisioning one or more resources.

junction
Describes a WebSEAL junction.

login
Policies that are related to login. For example, password expiry, account suspension due to failed login attempts, or account lockouts due to account inactivity.

management
Authorization of a management command. The specific management object type is contained in the resourceName.

messageQueue
A message queue.

netIncoming
Incoming network accesses are controlled by network resources: NetIncoming resource:`/OSSEAL/policy-branch/NetIncoming/protocol[/service[/host]]`

netOutgoing
Outgoing network accesses are controlled by the following network resource. NetOutgoing resource:`/OSSEAL/policy-branch/NetOutgoing/[/hostspec[/protocol[/service]]]`

orgContainer
The organization container defines the organization hierarchy for the managed resources.

passwordPolicy
Specifies a set of rules in which all passwords for one or more services must conform. For example, password strength and password aging.

policyUpdate
Indicates a policy update. For example, the product might receive a policy update (downloaded from the policy database).

protectedResource
A generic value for a protected resource. For example, Security Verify Access protected object or Security Verify Access protected object space.

provisioningAccount
Represents a user's identity on the target provisioning resource.

provisioningPolicy
Used to associate one or multiple groups of users with one or multiple entitlements. The group of users can be identified by organization or organization role. The entitlement is a construct to define a set of permissions, or privileges, on a managed provisioning resource.

provisioningResource
A resource for which Identity Provisioning is enabled.

serviceSelectionPolicy
Used in situations where the instance of a provisioning resource, on which the provisioning of an account is to take place, is determined dynamically based on account owner's attributes.

sudo
Describe commands that require more stringent access control than whether a particular program can be run. Sudo commands allow access control based not only on a command but also on the parameters passed to that command.

You can use Sudo commands to remove the requirements for a user to become the root user on a system in order to perform administrative tasks.

Sudo resources are identified in the Security Verify Access namespace in the following way: /OSSEAL/policy-branch/Sudo/*sudo-command*[/*sudo-orglass*]

surrogate
Surrogate resources. Operations that can change the user identity or group identity of a process are referred to as surrogate operations and are controlled by resources of type surrogate. Surrogate resource names follow the form: /OSSEAL/policy-branch/Surrogate/User/*user-name*.

tcb
Trusted Computing Base resources.

workflowTemplate
Defines the flow of a business workflow process.

url
An absolute URL identifying the resource accessed. Use the File resource type for `file://` URLs.

user
The user entity that application manages in the registry.

XPath

```
CommonBaseEvent/extendedDataElements[@name='resourceInfo']/children
[@name='type']/values
```

sequenceNumber

Reference information about the sequenceNumber element.

Description

An internal identifier for an audit event as shown in the XML output.

This information is not related to the following line in the CARSShredder.conf file:

```
cars_t_event, cars_seq_number, #RECORD_ID
```

severity

Reference information about the severity element.

Description

Identifies severity of the violation.

Values

String

The following strings are suggested values:

high
Violation of high severity.

low
Violation of low severity.

medium
Violation of medium severity.

XPath

```
CommonBaseEvent/extendedDataElements[@name='severity']/values
```

sourceComponentId

Reference information about the sourceComponentId element.

Description

A container for the information about what originated the audit record.

When displayed in output, this element uses the children of the auditComponentIdType element:

- sourceComponentId/@application
- sourceComponentId/@component
- sourceComponentId/@componentIdType
- sourceComponentId/@componentType
- sourceComponentId/@executionEnvironment
- sourceComponentId/@instanceId
- sourceComponentId/@location
- sourceComponentId/@locationType
- sourceComponentId/@processed
- sourceComponentId/@subComponent
- sourceComponentId/@threadId

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

sourceComponentId/@application

Reference information about the sourceComponentId/@application element.

Description

Refer to the Common Base Event specification.

Values

xsd:string

Refer to same field in the ComponentIdentification in the Common Base Event specification. For example: WebSEAL is an application within the component IBM Security Verify Access.

XPath

```
CommonBaseEvent/sourceComponentId/@application
```

sourceComponentId/@component

Reference information about the sourceComponentId/@component element.

Description

Product name, version, and fix pack level.

Values

String

For example, WebSEAL is an application within the component IBM Security Verify Access, version 10.0.0.

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@component
```

sourceComponentId/@componentIdType

Reference information about the sourceComponentId/@componentIdType element.

Description

Specifies the format and meaning of the component identified by this componentIdentification.

Values

xsd:string

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@componentIdType
```

sourceComponentId/@componentType

Reference information about the sourceComponentId/@componentType element.

Description

A well-defined name that is used to characterize all instances of a given kind of component.

Values

xsd:string

Refer to same field in the ComponentType in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@componentType
```

sourceComponentId/@executionEnvironment

Reference information about the sourceComponentId/@executionEnvironment element.

Description

The immediate environment that an application is running in.

Values

xsd:string

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@executionEnvironment
```

sourceComponentId/@instanceId

Reference information about the sourceComponentId/@instanceId element.

Description

Module instance information, for example, port number.

Values

String

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@instanceId
```

sourceComponentId/@location

Reference information about the sourceComponentId/@location element.

Description

Physical location of the reporting component.

Values

xsd:string

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@location
```

sourceComponentId/@locationType

Reference information about the sourceComponentId/@locationType element.

Description

Type of location.

Values

xsd:string

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@locationType
```

sourceComponentId/@processId

Reference information about the sourceComponentId/@processId element.

Description

Process ID.

Values

String

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@processId
```

sourceComponentId/@subComponent

Reference information about the sourceComponentId/@subComponent element.

Description

Module name.

Values

String

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@subComponent
```

sourceComponentId/@threadId

Reference information about the sourceComponentId/@threadId element.

Description

Thread ID.

Values

String

Refer to same field in the ComponentIdentification in the Common Base Event specification.

XPath

```
CommonBaseEvent/sourceComponentId/@threadId
```

startTime

Reference information about the startTime element.

Description

Start time of the operation.

Values

xsd:DateTime

Refer to the creationTime field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='startTime'][@type='dateTime']/values
```

suppressed

Reference information about the suppressed element.

Description

Identifies if the violation was suppressed.

Values

String

Use one of the following strings:

- yes
- no

XPath

```
CommonBaseEvent/extendedDataElements[@name='suppressed']/values
```

targetAccount

Reference information about the targetAccount element.

Description

Name of the user account.

Values

String

Any string allowed by targetResource.

XPath

```
CommonBaseEvent/extendedDataElements[@name='targetAccount']/values
```

targetInfoType

Reference information about the targetInfoType element.

Description

This element type represents information about the target of a management action, such as associating an access control list with a protected resource.

When displayed in output, this element uses the children of the targetInfoType element:

- targetInfoType.attributes
- targetInfoType.targetNames

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

targetInfo.attributes

Reference information about the targetInfo.attributes element.

Description

Array of attributes for the values for the target.

targetInfo.targetNames

Reference information about the targetInfo.targetNames element.

Description

Object this operation is targeted against.

String

String allowed for the target object name by the application.

Examples:

- For group associate, target is a list of users added to a group.
- For ACL associate, target is a resource name associated with an ACL.

- For ACL disassociate, target is a resource name disassociated with the ACL.

XPath

```
CommonBaseEvent/extendedDataElements[@name='mgmtInfo']/children
[@name='targetInfo']/children
[@name='targetNames']/values[1]
```

Note: This XPath assumes that the targetInfo is part of mgmtInfo.

targetResource

Reference information about the targetResource element.

Description

Name of the resource on which the account exists.

Values

String

Any string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='targetResource']/values
```

targetUser

Reference information about the targetUser element.

Description

Name of the user.

Values

String

Any string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='targetUser']/values
```

targetUserInfo (1)

Reference information about the targetUserInfo element when used with the AUDIT_WORKFLOW event type.

Description

A container for information about the target users when used with the AUDIT_WORKFLOW event type.

This element uses the children of userInfoType:

- userInfo.appUserName
- userInfo.attributes

- userInfo.callerList
- userInfo.domain
- userInfo.location
- userInfo.locationType
- userInfo.realm
- userInfo.registryUserName
- userInfo.sessionId
- userInfo.uniqueId

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

targetUserInfo (2)

Reference information about the targetUserInfo element when used with the AUDIT_MGMT_PROVISIONING event type.

Description

A container for information about the target users when used with the AUDIT_MGMT_PROVISIONING event type.

For AUDIT_MGMT_PROVISIONING events, `registryObjectInfo.type` must be `User`.

This element uses the children of registryObjectInfoType:

- registryObjectInfo.attributes
- registryObjectInfo.description
- registryObjectInfo.name
- registryObjectInfo.registryName
- registryObjectInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

targetUserRegistryInfo

Reference information about the targetUserRegistryInfo element.

Description

A container for information about the registry to which the target user belongs.

This element uses the children of the registryInfoType element:

- registryInfo.serverLocation
- registryInfo.serverLocationType
- registryInfo.serverPort
- registryInfo.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

terminateReason

Reference information about the terminateReason element.

Description

The reason for the termination.

Values

String

The following strings are suggested values:

idleTimeout
The session was terminated because it was inactive for too long.

sessionExpired
The session was terminated because its maximum lifetime was exceeded.

sessionDisplaced
The session was terminated because the session's user created a new session displacing this one.

sessionTerminatedByAdmin
The session was terminated by an administrative action.

userLoggedOut
The session was terminated at the user's request.

XPath

```
CommonBaseEvent/extendedDataElements[@name='terminateReason']/values
```

timestamp

Reference information about the timestamp element.

Description

End time of the operation.

Values

xsd:DateTime

If not specified, it is generated automatically. The timestamp is used in reports to determine when the audit event occurred. If the caller specifies the timestamp, it is the caller's responsibility to ensure that the timestamp provided is not spoofed.

Refer to the creationTime field in the Common Base Event specification.

XPath

```
CommonBaseEvent/@creationTime
```

type

Reference information about the type element.

Description

The type of command.

Values

String

The following strings suggested values:

config
Configuration object.

server
Object that represents an application server.

XPath

```
CommonBaseEvent/extendedDataElements[@name='type']/values
```

userInfo

Reference information about the userInfo element.

Description

The container for information about the user.

This element uses the children of userInfoType:

- userInfo.appUserName
- userInfo.attributes
- userInfo.callerList
- userInfo.domain
- userInfo.location
- userInfo.locationType
- userInfo.realm
- userInfo.registryUserName
- userInfo.sessionId
- userInfo.uniqueId

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

userInfo.appUserName

Reference information about the userInfo.appUserName element.

Description

User's name within a given application.

Values

String

Any arbitrary string allowed by the application. For example, a Security Verify Access user name.

The following strings are suggested values:

unauthenticated
An unauthenticated user

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='appUserName']/values
```

userInfo.attributes

Reference information about the userInfo.attributes element.

Description

Array of attributes in the user's credential.

Values

attributeType

Refer to “attributes element” on page 128 for details.

XPath

The XPath is the first name element from an array of attributes elements.

```
CommonBaseEvent/extendedDataElements
[@name='userInfo']/children[10]/children
[@name='name']/values
```

Note: The index is 10, for the `attributes` element must come after the `appUserName`, `callerList`, `domain`, `location`, `locationType`, `realm`, `registryUserName`, `sessionId`, and `uniqueId` elements

userInfo.callerList

Reference information about the userInfo.callerList element.

Description

A list of names representing the user's identities.

Values

String[]

Any arbitrary string allowed by the application can be used in the String[].

XPath

The XPath is the first callerList element from an array of callerList elements.

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='callerList']/values[1]
```

userInfo.domain

Reference information about the userInfo.domain element.

Description

Domain in which user belongs.

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='domain']/values
```

userInfo.location

Reference information about the userInfo.location element.

Description

Location of the user. Example: In the case of WebSEAL, where the user authenticated from.

Values

xsd:string

Refer to the location field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='location']/values
```

userInfo.locationType

Reference information about the userInfo.locationType element.

Description

Type of location.

Values

xsd:Name

Refer to the locationType field in the Common Base Event specification.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='locationType']/values
```

userInfo.realm

Reference information about the userInfo.realm element.

Description

The registry partition to which the user belongs.

Values

String

Any arbitrary string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='realm']/values
```

userInfo.registryUserName

Reference information about the userInfo.registryUserName element.

Description

The registry partition to which the user belongs.

Values

String

Any arbitrary string allowed by the application.

Use "Not Available" when not available.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='registryUserName']/values
```

userInfo.sessionId

Reference information about the userInfo.sessionId element.

Description

ID for the user's session.

Values

Any arbitrary string

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='sessionId']/values
```

userInfo.uniqueId

Reference information about the userInfo.uniqueId element.

Description

User's unique identifier.

Values

Integer UUID

A value of `-99999` means that a unique ID is not available.

For events generated by Security Verify Access, the unique ID is not available and is always set to `0`. When using the distributed session cache component of Security Verify Access, the unique ID is always set to `-99999`.

XPath

```
CommonBaseEvent/extendedDataElements[@name='userInfo']/children
[@name='uniqueId']/values
```

userInputs

Reference information about the userInputs element.

Description

A container for information about the user inputs that are related to the work item. The inputs are collected as a list of attributes. For example, for approval and reject, one attribute could be the comment.

This element uses the children of the attributeType element:

- attributeType.name
- attributeType.source
- attributeType.value

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

violationClassification

Reference information about the violationClassification element.

Description

Identifies the type of violation.

Values

String

The following strings suggested values:

account
Generic classification for policy violations related to an account, or attributes associated with an account, that does not fit in one of the specific account violation classifications.

accountDisallowed
Account was disallowed. Example: Guest accounts could be forbidden.

aclRestriction
The authorization settings on a protected resource violate the policy. Example: The ACL settings on the executables for a Web server might be improperly set.

antiVirus
The proper antivirus protection is not in place. Example: Version*x.y* of antivirus product ABC may be required, or the antivirus scan must be configured to run at least once per week.

audit
The audit settings on a system may not comply with the policy. Example: The policy may require that all failed authentication attempts be audited. If audit settings do not comply, a violation is logged.

netConfig
Network configuration settings are not set as required by the policy. Example: The -s option must be specified when using the netlsd daemon in AIX.

password
The password policy is not being adhered to. Example: All passwords must be 8 characters or longer.

prohibitedService
Certain services might be prohibited. Example: Policy may require that TFTP never be active on a system.

softwareVersion
Policy may require that specific versions of software be installed. Example: A down-level version of Microsoft IIS or a version that requires a patch might be installed on a production server.

sysConfig
System configuration settings are not set as required by the policy. Example: Certain system log files may be required to exist.

XPath

```
CommonBaseEvent/extendedDataElements[@name='violationClassification']/values
```

violationDescription

Reference information about the violationDescription element.

Description

Predefined description of the particular violation.

Values

String

Any string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='violationDescription']/values
```

violationName

Reference information about the violationName element.

Description

Name of specific policy violation. Example: "Win2K Guest Account Restriction".

Values

String

Any string allowed by the application.

XPath

```
CommonBaseEvent/extendedDataElements[@name='violationName']/values
```

workItemInfo

Reference information about the workItemInfo element.

Description

An element type that represents information about a work item used in events related to workflow operations.

This container uses the children of workItemInfoType:

- workItemInfoType.id
- workItemInfoType.type

XPath

No valid XPath for the shredder configuration file. A valid XPath requires a `values` declaration.

workItemInfoType.id element

Reference information about the workItemInfoType.id element.

Description

Unique identifier of the work item.

Values

String

XPath

```
CommonBaseEvent/extendedDataElements[@name='workItemInfoType']/children
[@name='id']/values
```

workItemInfoType.type element

Reference information about the workItemInfoType.type element.

Description

Type of the work item.

Values

String

The following strings are suggested values:

approval
This type of work item allows a user to either approve or reject a specific request.

requestForInfo
This type of work item allows a user to provide additional information for a specific request.

workOrder
This type of work item is used to request manual operations for the user. For example, a work order to manually create a specific account on a resource.

XPath

```
CommonBaseEvent/extendedDataElements[@name='workItemInfoType']/children
[@name='type']/values
```

Chapter 6. Routing files

Routing files are ASCII files that you can use to customize the logging events for C language-based servers, daemons, and other C-language programs and applications. You can use the contents of routing files to control aspects of event logging, such as:

- Whether to enable logging for specific event classes
- Where to direct the output for each event class
- How many log files to use for each event class
- How large each log file can be for each event class

Locations of routing files

The location of the routing files can be found in the appliance dashboard. In the appliance dashboard, navigate **Web** > **Runtime Component** > **Manage** > **Configuration Files** > **Tracing Configuration Files**.

Routing file entries

Each routing file contains entries that control the logging of events. Use the following format (entered on a single line without spaces) when you define entries in routing files:

component:*subcomponent*.*level*[[,*subcomponent*.*level*]...] :*destination*:*location*
[[;*destination*:*location*]...] [;GOESTO:{*other_severity* | *other_component*}]

Where:

***component*:*subcomponent* [[,*subcomponent*]...]**
Specifies the component, subcomponents, and reporting levels of events to log.

For the component portion, you can specify an asterisk (*) to log data for all components.

For the subcomponent portion, you can specify an asterisk (*) to log data for all subcomponents of the specified component.

destination
Specifies where to log the events. For each destination, you must specify a location. When you specify multiple destination-location pairs, separate each pair with a semicolon (;). The following destinations are valid:

DISCARD
Discards the events.

FILE
Writes the events as ASCII text in the current code page and locale to the specified location.

When you use this destination on the appliance, do not include any path information.. Optionally, you can follow the `FILE` destination by a period and two numbers that are separated by a period (for example, `FILE.10.100`).

The first value indicates the number of files to use. The second value indicates the number of events each file can contain. If you do not specify these values, there is only 1 log file that grows without limit.

The average size of an ASCII event is 200 bytes. Because the maximum size of a log file is 2 GB, the maximum number of events must be limited to approximately 10,000,000 events.

STDERR
Writes the events as ASCII text in the current code page and locale to the standard error device.

STDOUT
Writes the events as ASCII text in the current code page and locale to the standard output device.

TEXTFILE
Same a FILE.

UTF8FILE
Writes the events as UTF-8 text to the specified location.

When you use this destination, do not include any path information. Optionally, you can follow the UTF8FILE destination by a period and two numbers that are separated by a period (for example, UTF8FILE.10.100).

The first value indicates the number of files to use. The second value indicates the number of events each file can contain. If you do not specify these values, there is only 1 log file that grows without limit.

The average size of a UTF-8 event is 200 bytes. Because the maximum size of a log file is 2 GB, the maximum number of events must be limited to approximately 10,000,000 events.

Note: When the operating system does not use a UTF-8 code page, the conversion to UTF-8 can result in data loss. When data loss occurs, the log file contains a series of question mark (?) characters at the location where the data conversion was problematic.

XMLFILE
Writes events to the specified location in the XML log format.

When you use this destination, do not include any path information. Optionally, you can follow the XMLFILE destination by a period and two numbers that are separated by a period (for example, XMLFILE.10.100). The first value indicates the number of files to use. The second value indicates the number of events each file can contain.

If you do not specify these values, there is only 1 log file that grows without limit.

The maximum size of a log file is 2 GB.

XMLSTDERR
Writes events to the standard error device in the XML log format.

XMLSTDOUT
Writes events to the standard output device in the XML log format.

GOESTO:{*other_severity* | *other_component*}]
Specifies that events must additionally be routed to the same destination and location as events of the specified component.

location
Specifies the name and location of the log file. When the destination is TEXT, TEXTFILE, UTF8FILE or XMLFILE, you must specify a location. When the destination is DISCARD, STDERR, STDOUT, XMLSTDERR, or XMLSTDOUT, you must specify a hyphen (-).

When you specify a fully qualified file name, you can use the %ld character string to insert the process ID into the file name.

When the number of files is specified as part of the destination, a period and the file number are appended to the specified log file.

Note: On Windows operating systems, the file name must not end with a period. If the file name ends with a period, when the file number is appended, the file name contains two consecutive periods. File names with two consecutive periods are not valid.

On AIX, Linux, and Solaris operating systems, the file name must be followed by:

- File permissions.
- The user who owns the file.
- The group that owns the file.

Use the following format:

```
location:permissions:owner:group
```

Chapter 7. Configuration stanzas

This appendix describes the guidelines for changing the following files:

- Configuration files.
- The location of the configuration files.
- The contents of the configuration files.

These files are used for auditing and statistic gathering purposes.

Guidelines for changing configuration files

These guidelines are provided to help you update the Security Verify Access configuration files. The guidelines are divided into the following categories:

General guidelines

Use the following general guidelines when you change the configuration settings:

- There is no order dependency or location dependency for stanzas in any configuration file.
- Stanza entries are marked as required or optional. When an entry is required, the entry must contain a valid key and value.
- Do not change the names of the keys in the configuration files. Changing the name of the key might cause unpredictable results for the servers.
- Stanza entries and key names are case-sensitive. For example, `usessl` and `UseSSL` are treated as different entries.
- Spaces are not allowed for names of keys.
- For the key value pair format of *key* = *value*, the spaces that surround the equal sign (=) are not required.
- Non-printable characters (such as tabs, carriage returns, and line feeds) that occur at the end of a stanza entry are ignored. Non-printable characters are ASCII characters with a decimal value less than 32.

Default values

Use the following guidelines when you change default configuration settings:

- Many values are created or modified only by using configuration programs. Do not manually edit these stanzas or values.
- Some values are added automatically during configuration. These values are needed for the initialization of the server after the configuration.
- The default values for a stanza entry might be different, depending on the server configuration. Some key value pairs are not applicable to certain servers and are omitted from the default configuration file for this server.

Strings

Some values accept a string value. When you manually edit the configuration file, use the following guidelines to change configuration settings that require a string:

- String values are expected to be characters that are part of the local code set.
- Additional or different restrictions on the set of allowable string characters might be imposed. For example, many strings are restricted to ASCII characters. Consult each stanza entry description for any restrictions.

- Double quotation marks are sometimes, but not always, required when you use spaces or more than one word for values. See the descriptions or examples for each stanza entry when in doubt.
- The minimum and maximum lengths of user registry-related string values, if there are limits, are imposed by the underlying registry. For example, for Active Directory the maximum length is 256 alphanumeric characters.

Defined strings

Some values accept a string value, but the value must be a set of defined strings. When you manually edit the configuration file, make sure that the string value you type matches one of the valid defined strings values.

For example, the `[aznapi-configuration]` stanza section contains the following entry:

```
mode = {local|remote}
```

The value for `mode` is expected to be `local` or `remote`. Any other value is invalid and results in an error.

File names

Some values are file names. For each stanza entry that expects a file name as a value, the description of the stanza entry specifies which of the following constructs are valid:

Filename
No directory path included.

Relative filename
A directory path is allowed but not mandatory.

These files typically are expected to be located relative to the location of a standard Security Verify Access directory. The stanza entry for each relative path name lists the root directory to which the file name is relative.

Fully qualified absolute path
An absolute directory path is required.

Some stanza entries allow more than one of the file name choices.

The set of characters that is permitted in a file name can be determined by the file system and by the local code set. For Windows operating systems, file names cannot have a backward slash (\), a colon (:), a question mark (?), or double quotation marks (").

Integers

Many stanza entries expect the value for the entry to be expressed as an integer. When you define an entry with an integer, consider the following guidelines:

- Stanza entries that take an integer value expect integer values within a valid range. The range is described in terms of a *minimum* value and a *maximum* value.

  For example, in the `[ivmgrd]` stanza, the `max-notifier-thread` stanza entry has a minimum value of 1 second and a maximum value of 128 threads.
- For some entries, the integer value must be positive, and the minimum value is 1. For other entries, a minimum integer value of 0 is allowed.

  Use caution when you set an integer value to 0. For example, an integer value of 0 might disable the function that is controlled by that stanza entry. For example, in the `[ivacld]` stanza, the entry `tcp-req-port = 0` disables the port number. Or, an integer value of 0 might indicate that the number is unlimited. For example, in the `[ldap]` stanza, the entry `max-search-size = 0` means that there is no limit to the maximum search size.
- For some entries that require integer values, Security Verify Access does not impose an upper limit for the maximum number allowed. For example, there is typically no maximum for timeout-related values, such as `timeout = number` in the `[ldap]` stanza.

For this type of entry, the maximum number is limited only by the size of memory that is allocated for an integer data type. This number can vary, based on the type of operating system. For systems that allocate 4 bytes for an integer, this value is 2147483647.

However, as the administrator, use a number that represents the value that is most logical for the value you are trying to set.

Boolean values

Many stanza entries represent a Boolean value. Security Verify Access recognizes the Boolean values `yes` and `no`.

Some of the entries in the configuration files are read by other servers and utilities. For example, many entries in the `[ldap]` stanza are read by the LDAP client. Some of these other programs recognize more Boolean characters:

- `yes` or `true`
- `no` or `false`

Anything other than `yes` | `true`, including a blank value, is interpreted as `no` | `false`.

The recognized Boolean entries are listed for each stanza entry. See the individual descriptions to determine when `true` or `false` are also recognized.

Configuration file reference

The operation of the Security Verify Access server is controlled by using configuration files. Each configuration file contains sections that are called *stanzas*.

Server configuration files are ASCII text-based and contain stanza entries. Configuration files are processed only when the servers start.

Location of configuration files

This section provides information about the server-specific location of the configuration files.

Security Verify Access runtime

If you installed Security Verify Access in the default directories, the configuration files for the runtime are found in the appliance dashboard.

From the appliance dashboard, navaigate to **Web** > **Runtime Component** > **Manage** > **Configuration Files**.

Contents of configuration files

This section provides information about the stanzas and stanza entries in the available configuration files. The configuration files are used for auditing and statistic gathering purposes.

Security Verify Access configuration files

Within the configuration files for the Security Verify Access servers, you can define auditing and statistics characteristics. All C-based servers have the `[aznapi-configuration]` stanza, and WebSEAL has an additional `[logging]` stanza.

Configuration file stanza reference

Within configuration files, stanza labels are shown within brackets, such as `[`*`stanza-name`*`]`. For example, the `[ssl]` stanza in the `ivmgrd.conf` configuration file defines the Secure Sockets Layer (SSL) configuration settings for the policy server. The `[ldap]` stanza defines the configuration settings that are required by the policy server to communicate with an LDAP-based user registry.

Each stanza in a Security Verify Access configuration file contains one or more key value pairs, which contain information that is expressed as a paired set of parameters. Each stanza entry is a key-value pair in the following format:

```
key = value
```

You must not change the names of the keys in the configuration files. Changing the name of the key might cause unpredictable results in the servers. The spaces that surround the equal sign (=) are not required.

The initial installation of Security Verify Access establishes many of the default values. Some values are static and never change; other values can be modified to customize server functionality and performance.

The following stanza descriptions provide a list of the valid stanza entries. Each stanza entry consists of key value pairs. Each stanza entry includes a description of its default behavior, when applicable.

[aznapi-configuration] stanza

The stanza entries for native Security Verify Access auditing and statistics gathering are in the `[aznapi-configuration]` stanza of the server-specific configuration file. The `[aznapi-configuration]` stanza contains more entries than the ones that are listed. For a complete list of entries that can be used in the server-specific configuration files, see the administration guide for that server or plug-in.

logcfg

Syntax

```
logcfg = category:[log-agent][[parameter[=value]] ...]
```

Description

Enables logging and auditing for the application. Category, destination, and other parameters are used to capture Security Verify Access auditing and logging events.

Each server provides its own event logging setting in its corresponding configuration file.

Options

category*:*log-agent
The category of the auditing event and the destination. *log-agent* is one of the following agents:

- `stdout`
- `stderr`
- `file path=`
- `pipe`
- `remote`

parameter=value
Allowable parameters. The parameters vary, depending on the category, the destination of events, and the type of auditing you want to perform.

See “Audit event logging” on page 11 for information about the log agents and the configuration parameters. Each log agent supports different parameters.

Usage

Optional

Default value

Remove the pound signs (#) at the beginning of the configuration file lines to enable authentication or authorization auditing (or both) for the application.

Example

```
logcfg = audit.azn:file path=audit.log,flush_interval=20,log_id=audit_log
```

[logging] stanza

The `[logging]` stanza contains the configuration details for logging HTTP audit events for WebSEAL servers. WebSEAL can be configured to maintain the following HTTP activities:

- agents
- referers
- requesters

The `[logging]` stanza is in the WebSEAL `webseald.conf` configuration file. Assume that the configuration file contains auditing entries in both the `[aznapi-configuration]` stanza and the `[logging]` stanza. Then, the logging details in the `[aznapi-configuration]` stanza take precedence over repeated details in the `[logging]` stanza.

For details about WebSEAL event processing, see “Process flow for logcfg logging” on page 30.
For information about the `[aznapi-configuration]` stanza entries in the WebSEAL webseald.conf configuration file, see the Stanza Reference topics in the IBM® Knowledge Center.

absolute-uri-in-request-log

Syntax

```
absolute-uri-in-request-log = {yes|no}
```

Description

Logs the absolute URI in the HTTP audit records. Adds protocol and host to the path.

Options

yes
Log the absolute URI.

no
Do not log the absolute URI.

Usage

This stanza entry is required.

Default value

no

Example

```
absolute-uri-in-request-log = no
```

agents

Syntax

```
agents = {yes|no}
```

Description

Enables or disables the agents log. This log records the contents of the `User_Agent:` header of each HTTP request.

Options

yes
The value `yes` enables logging for the agents.

no
The value no disables logging for the agents.

Usage

This stanza entry is required.

Default value

`yes`

Example

```
agents = yes
```

agents-file

Syntax

```
agents-file = file_name
```

Description

Fully qualified path to the agents log file.

Options

file_name
Name of the agents log file.

Usage

This stanza entry is required.

Default value

The default location is `agent.log`, located under the WebSEAL installation directory.

Example

Example on AIX, Linux, and Solaris:

```
agents-file = agent.log
```

config-data-log

Syntax

```
config-data-log = fully_qualified_path
```

Description

Fully qualified path to the configuration data log file.

Options

fully_qualified_path
Fully qualified path to the configuration data log file.

Usage

This stanza entry is required.

Default value

The default location is `log/config_data.log`, located under the WebSEAL installation directory.

Example

Example on AIX, Linux, and Solaris:

```
config-data-log = /var/pdweb/log/config_data.log
```

flush-time

Syntax

```
flush-time = number_of_seconds
```

Description

Integer value that indicates the frequency, in seconds, to force a flush of log buffers.

Options

number_of_seconds
Integer value that indicates the frequency, in seconds, to force a flush of log buffers. The minimum value is 1 second. The maximum value is 600 seconds.

Usage

This stanza entry is optional.

Default value

20

Example

```
flush-time = 20
```

gmt-time

Syntax

```
gmt-time = {yes|no}
```

Description

Enables or disables logging requests in Greenwich Mean Time (GMT) instead of the local time zone.

Options

yes
A value of yes means to use GMT.

no
A value of no means to use the local time zone.

Usage

This stanza entry is required.

Default value

no

Example

```
gmt-time = no
```

host-header-in-request-log (deprecated)

Syntax

```
host-header-in-request-log = {yes|no}
```

Description

Log the Host header at the front of each line in the request log and the combined log.

Options

yes
Log the Host header.

no
Do not log the Host header.

Usage

This stanza entry is required.

Default value

no

Example

```
host-header-in-request-log = no
```

max-size

Syntax

```
max-size = number_of_bytes
```

Description

Integer value that indicates the size limit of the log files. This value applies to the request, referrer, and agent logs. The size limit is also known as the rollover threshold. When the log file reaches this threshold, the original log file is renamed, and a new log file with the original name is created.

Options

number_of_bytes

When the value is zero (0), no rollover log file is created.

When the value is a negative integer, the logs are rolled over daily, regardless of the size.

When the value is a positive integer, the value indicates the maximum size, in bytes, of the log file before the rollover occurs. The allowable range is from 1 byte to 2 MB.

Usage

This stanza entry is required.

Default value

2000000

Example

```
max-size = 2000000
```

referers

Syntax

```
referers = {yes|no}
```

Description

Enables or disables the referers log. This log records the `Referer:` header of each HTTP request.

Options

yes
The value `yes` enables referers logging.

no
The value no disables referers logging.

Usage

This stanza entry is required.

Default value

`yes`

Example

```
referers = yes
```

referers-file

Syntax

```
referers-file = file_name
```

Description

Name of the referers log file.

Options

file_name
Name of the referers log file.

Usage

This stanza entry is required.

Default value

The default location is `referer.log`, located under the WebSEAL installation directory.

Example

Example on AIX, Linux, and Solaris:

```
referers-file = referer.log
```

requests

Syntax

```
requests = {yes|no}
```

Description

Enables or disables the requests log. This log records standard logging of HTTP requests.

Options

yes
The value `yes` enables requests logging.

no
The value no disables requests logging.

Usage

This stanza entry is required.

Default value

`yes`

Example

```
requests = yes
```

requests-file

Syntax

```
requests-file = file_name
```

Description

Name of the request log file.

Options

file_name
Name of the request log file.

Usage

This stanza entry is required.

Default value

The default location is `request.log`, located under the WebSEAL installation directory.

Example

Example on AIX, Linux, and Solaris:

```
requests-file = request.log
```

server-log

Syntax

```
server-log = file_name
```

Description

Name of the server error log file.

Options

file_name
Name of the server error log file.

Usage

This stanza entry is required.

Default value

The default location is `webseald.log`, located under the WebSEAL installation directory.

Example

Example on AIX, Linux, and Solaris:

```
server-log = msg__webseald.log
```

[pdaudit-filter] stanza

The stanza entries for native Security Verify Access auditing are in the `[pdaudit-filter]` stanza of the server-specific `pdaudit.conf` configuration file.

logcfg

Syntax

```
logcfg = category:[log-agent][[parameter[=value]] ...]
```

Description

Enables logging and auditing for the application. Category, destination, and other parameters are used to capture Security Verify Access auditing and logging events.

Each server provides its own event log setting in its corresponding configuration file.

Options

category*:*log-agent
The category of the auditing event and the destination. *log-agent* is one of the following agents:

- `stdout`
- `stderr`
- `file path=`
- `pipe`

- `remote`

parameter=value
Allowable parameters. The parameters vary, depending on the category, the destination of events, and the type of auditing that you want to complete.

See “Audit event logging” on page 11 for information about the log agents and the configuration parameters. Each log agent supports different parameters.

Usage

Optional

Default value

Remove the number signs (#) at the beginning of the configuration file lines to enable authentication or authorization auditing (or both) for the application.

Example

```
logcfg = audit.azn:file path=audit.log,flush_interval=20,log_id=audit_log
```

Chapter 8. Commands and utilities

This section provides reference information about the commands and utilities that are used for auditing, statistics gathering, and for viewing and changing entries in configuration files.

Reading syntax statements

The reference documentation uses the following special characters to define syntax:

[ ]
Identifies optional options. Options that are not enclosed in brackets are required.

...
Indicates that you can specify multiple values for the previous option.

|
Indicates mutually exclusive information. You can use the option to the left of the separator or the option to the right of the separator. You cannot use both options in a single use of the command.

{ }
Delimits a set of mutually exclusive options when one of the options is required. If the options are optional, they are enclosed in brackets ([ ]).

****
Indicates that the command line wraps to the next line. It is a continuation character.

The options for each command or utility are listed alphabetically in the Options section or in the Parameters section. When the order of the options or parameters must be used in a specific order, this order is shown in the syntax statements.

Commands

Table 28 on page 199 lists the **pdadmin** commands that can be used during auditing and gathering of statistics activities.

Table 28. Auditing and statistics commands

Command	Description
"login" on page 199	Establishes authentication credentials that are used during communication with the Security Verify Access policy server.
"server list" on page 202	Lists all registered Security Verify Access servers.
"server task stats" on page 202	Enables the gathering of statistical information for an installed Security Verify Access server or server instance.

login

Establishes authentication credentials that are used for communication with the Security Verify Access policy server. These credentials are used to determine access privileges for the user to policy server data. Most commands cannot be performed unless an explicit login is done.

This command does not require a login or authentication to use.

Syntax

login –a *admin_id* [–p *password*] [–d *domain*]

login –a *admin_id* [–p *password*] [–m]

login –l

Description

Credentials are used to determine user access privileges to policy server data. Except the **context, errtext**, **exit**, **help**, **login**, **logout**, and **quit** commands, and the local configuration commands, a user ID, and a password are needed for authentication.

Credentials are not accumulated or stacked. A **login** command completely replaces any existing credentials.

In interactive mode, the **pdadmin** prompt changes, depending on how the user logs in:

- Not interactive mode. This command starts the **pdadmin** utility. In interactive mode, the **login** commands are entered from the pdadmin> prompt.

```
c:\> pdadmin
pdadmin>
```

- A user local login that is performed for local configuration. No authentication is required.

```
pdadmin> login -l
pdadmin local>
```

- An administrator login that is performed to the local domain. In some cases, the local domain might be the management domain, which is named Default. Authentication is required.

```
pdadmin> login -a sec_master -p secmstrpw
pdadmin sec_master>
```

- A user login that is performed to the local domain. Authentication is required.

```
pdadmin> login -a dlucas -p lucaspw
pdadmin dlucas>
```

- A user login that is performed to another domain other than their local domain. Authentication is required.

```
pdadmin> login -a dlucas -p lucaspw -d domain_a
pdadmin dlucas@domain_a>
```

- A user login that is performed to the management domain. Authentication is required.

```
pdadmin> login -a dlucas -p lucaspw -m
pdadmin dlucas@Default>
```

Options

–a *admin_id*
Specifies an administrator ID.

–d *domain*
Specifies the Security Verify Access secure domain for the login. The *admin_id* user must exist in this domain.

–m
Specifies that the login operation must be directed to the management domain. The *admin_id* user must exist in this domain.

Note: Only one of the following domain options can be specified: –d *domain* or –m. If neither option is specified, the target domain is the local domain that is configured for the system. The *admin_id* user must exist in the target domain, whether it is explicitly specified.

–p *password*
Specifies the password for the *admin_id* user. If this option is not specified, the user is prompted for the password. The password cannot be specified if the *admin_id* is not specified.

-l
Specifies a local login operation. When modifications are made to local configuration files by using the **config** commands, a local login is required before you can run commands. The user can run the **context show** command to view more authentication information.

Return codes

0
The command completed successfully.

1
The command failed. When a command fails, the **pdadmin** command provides a description of the error and an error status code in hexadecimal format (for example, `0x14c012f2`). See "Error messages" in the IBM Knowledge Center. This reference provides a list of the Security Verify Access error messages by decimal or hexadecimal codes.

Examples

- The following example logs the `sec_master` user in to the management domain and then displays the authentication context for the user:

```
pdadmin> login -a sec_master -p pa55w0rd -m

pdadmin sec_master> context show

User: sec_master
Domain: Default
The user is logged in to the management domain.
```

- The following example logs in a user to the `domain1` domain and then displays the authentication context for the user:

```
pdadmin> login -a domain1_admin -p d0main1pwd -d domain1

pdadmin domain1_admin@domain1> context show

User: domain1_admin
Domain: domain1
The user is not logged in to the management domain
```

- The following example interactively logs in the user to their local domain that is configured for the system. The domain name is `testdomain`. The example then displays the authentication context of the user:

```
pdadmin> login
Enter User ID: testdomain_admin
Enter password: adminpwd

pdadmin testdomain_admin> context show

User: testdomain_admin
Domain: testdomain
The user is not logged in to the management domain
```

- The following example of a local login demonstrates how the prompt changes, depending on the type of interactive login:

```
c:\> pdadmin login -l
```

Provides this prompt:

```
pdadmin local>
```

server list

Lists all registered Security Verify Access servers.

Requires authentication (administrator ID and password) to use this command.

Syntax

server list

Description

Lists all registered Security Verify Access servers. The name of the server for all server commands must be entered in the exact format as it is displayed in the output of this command. The **server list** command does not have such a requirement.

Options

None.

Return codes

0
The command completed successfully.

1
The command failed. When a command fails, the **pdadmin** command provides a description of the error and an error status code in hexadecimal format (for example, `0x14c012f2`). See "Error messages" in the IBM Knowledge Center. This reference provides a list of the Security Verify Access error messages by decimal or hexadecimal codes.

Example

The following example lists registered servers:

```
pdadmin> server list
```

The output is as follows:

```
ivmgrd-master
ivacld-server1
ivacld-server2
```

where `ivmgrd-master` represents the Policy server; `ivacld-server2` and `ivacld-server1` represent Authorization server instances.

server task stats

Manages the gathering and reporting of statistics for Security Verify Access servers and server instances.

Requires authentication (administrator ID and password) to use this command.

Syntax

server task *server_name–host_name* stats get [*component*]

server task *server_name–host_name* stats list

server task *server_name–host_name* stats off [*component*]

server task *server_name–host_name* stats on *component* [*interval* [*count*]] [*destination*]

server task *server_name–host_name* stats reset [*component*]

server task *server_name–host_name* stats show [*component*]

Description

The **server task stats** command manages the gathering and reporting of statistics for Security Verify Access servers and server instances. You can use the **stats** commands with configuration setting that are defined by the stanza entries in the server configuration file to manage statistics.

Statistics gathering is enabled through:

- The **stats on** command.
- The defined configuration settings.

Then, you can use the **stats on** commands to modify the behavior for gathering and reporting statistics.

For example, statistics are enabled to create five statistics reports with each report generated each day. You can use the **stats on** command to change the frequency to every 12 hours. For this example, assume that the following command started statistics gathering:

```
pdadmin sec_master> server task PDWebPI-linuxweb.wasp.ibm.com stats on \
pdwebpi.stats 86400 5 file path=stats.log
```

To modify the interval to 12 hours and create 10 reports, issue the following command:

```
pdadmin sec_master> server task PDWebPI-linuxweb.wasp.ibm.com stats on \
pdwebpi.stats 43200 10
```

Although the destination is not specified, the statistics infrastructure assumes any preexisting value. Entering the previous command does disable statistics from being written to the previously defined log file. However, if you specified a different destination, statistics reports would be written to the new destination only. You cannot use the **stats on** command to write statistics reports to more than one destination.

For more information about gathering statistics, see the Auditing topics in the Knowledge Center.

Options

component
Specifies the component about which to gather or report statistics.

count
Specifies the number of reports to send to a log file. When you use the *count* option, you must specify the *interval* option. If you specify the *interval* option without the *count* option, the duration of reporting is indefinite.

After the count value is reached, reporting to a log file stops. Although statistics are no longer sent to a log file, the statistic component is still enabled. You can obtain reports from memory by using the **stats get** command.

destination
Specifies where the gathered statistics are written, where *destination* can be one of the following options:

file path=*file_name*
Specifies the fully qualified name of the log file.

log_agent
Specifies a directory where statistics information is gathered. For more information about logging events, see the Troubleshooting topics in the Knowledge Center.

get
Displays the current report for a specific component or for all enabled components. If you specify the *component* option, displays the current report for that component; otherwise, displays the current report for all enabled components.

interval
Specifies the interval in seconds when statistics are sent from memory to a log file. When this option is specified, statistics are sent, by default, to the server-specific log file designated by the `logcfg` entry in the server configuration file. You can specify another location by using the *destination* option. If an interval is not specified, statistics are not sent to a log file, but remain in memory.

Although statistics are not sent to a log file, the statistic component is still enabled. You can obtain reports from memory by using the **stats get** command.

list
Lists all components that are available to gather and report statistics.

off
Disables gathering of statistics for a specific component or for all components. If you specify the *component* option, disables gathering of statistics for that component; otherwise, disables gathering of statistics for all components.

on
Enables gathering of statistics for a specific component. When you enable gathering of statistics, you can also set the reporting frequency, count, and log file.

reset
Resets gathering of statistics for a specific component or for all enabled components. If you specify the *component* option, resets gathering of statistics for that component; otherwise, resets gathering of statistics for all components.

server_name–host_name
Specifies the name of the server or server instance. You must specify the server name in the exact format as it is shown in the output of the **server list** command.

For example, if the configured name of a single WebSEAL server on host `example.dallas.ibm.com` is `default`, the *server_name* would be `default-webseald` and the *host_name* would be `example.dallas.ibm.com`. For this example, the name of the server would be `default-webseald-example.dallas.ibm.com`.

If multiple server instances are configured on the same computer, for example:

- The host is `example.dallas.ibm.com`.
- The configured name of the WebSEAL server instance is `webseal2-webseald`.

Then,

- The *server_name* is `webseal2-webseald`.
- The *host_name* is `example.dallas.ibm.com`.
- The name of the server instance is `webseal2-webseald-example.dallas.ibm.com`.

show
Lists all enabled components or indicates whether a specific component is enabled. If you specify the *component* option and the component is enabled, the output lists that component; otherwise, no output is displayed. If you do not specify the *component* option, the output lists all enabled components.

Return codes

0
The command completed successfully.

1
The command failed. See the Messages topics in the Knowledge Center for more information.

Examples

- The following example uses the **stats list** command to lists all enabled components on the ivacld-mogman.admogman.com authorization server:

```
#pdadmin sec_master> server task ivacld-mogman.admogman.com stats list

pd.ras.stats.monitor
pd.log.EventPool.queue
```

- The following example:
  - Uses the **status on** command to enable gathering of statistics for the pd.log.EventPool.queue component on the ivacld-mogman.admogman.com authorization server.
  - Sets the reporting frequency to 30 days, that is, 2592000 seconds.
  - Sets the destination to the myEPstats.log log file.

```
#pdadmin sec_master> server task ivacld-mogman.admogman.com stats on \
pd.log.EventPool.queue 2592000 file path=myEPstats.log
```

See also

“server list” on page 202

Index

F

G

H

L

M

O

P

Q

R

S

T

U

V

W

X