IdPLogging

Customizing Identity provider Logs

The Identity Provider uses the Logback logging system. The Logback manual provides an exhaustive set of directions and available options that may be configured. This document does not attempt to replicate this information but instead provides Shibboleth specific information, as it pertains to logging, and instructions for performing simple, common, tasks.

Logging Configuration

The logging configuration for the IdP is located at **$IDP_HOME/conf/logging.xml**. This file is checked for changes every 10 minutes by default and is reloaded if changes have been made. This means a deployer can keep the logging level at WARN until a problem occurs and then change the logging to DEBUG to get more information if the problem persists, all without restarting the IdP.

Log Files

By default Shibboleth 2.0 Identity Providers write to three log files.

`idp-access.log` contains a log entry for each time the IdP is accessed, whether information was ever sent back or not. These messages include request time, remote host making the request, server host name and port, and the request path. This log is written in the machine parsable format `requestTime|remoteHost|serverHost|serverPort|requestPath|`.

`idp-audit.log` contains a log entry for each time the IdP sends data to a relying party. These messages include the audit event time, IdP and relying party IDs, request and response binding, communication profile ID, request and response ID, principal name, authentication method, and released attribute of the current user. This log is written in the machine parsable format `auditEventTime|requestBinding|requestId|relyingPartyId|messageProfileId|assertingPartyId|responseBinding|responseId|principalName|authNMethod|releasedAttributeId1,releasedAttributeId2,|nameIdentifier|assertion1ID,assertion2ID,|`
Note the name identifier and assertion IDs were added in V2.1.

`idp-process.log` contains messages logged during the normal operation of the IdP. This log is meant to be human readable and contains messages that indicate what the IdP is currently doing, encountered errors, warning messages that may indicate potential problems, etc.

All logging messages are "rolled over" at midnight each night, if the IdP is running, or the next time the IdP starts up after that.

Compression

Log files can be automatically compressed by adding the suffix ".zip" or ".gz" to the `<FileNamePattern>` of an log file appender in **$IDP_HOME/conf/logging.xml**.
Consult the the Logback documentation for further information.

Useful Loggers

The following, coarse grained, loggers provide useful information in most situations:

Category	Description
Shibboleth-Access	The logger to which shibboleth access messages (think HTTP access logs) are written
Shibboleth-Audit	The logger to which shibboleth audit messages are written
PROTOCOL_MESSAGE	The logger to which incoming and outgoing XML protocol messages are logged
org.opensaml	Messages related only to receiving, parsing, evaluating security of, producing, and encoding SAML messages. Note, this produces **a lot** of log messages, especially at IdP startup.
org.opensaml.saml2.metadata.provider	Information regarding metadata loading, refreshing, and querying.
edu.internet2.middleware.shibboleth	Messages related to all the non-SAML message parsing/encoding work; profile handling, authentication, attribute resolution and filtering

edu.internet2.middleware.shibboleth.idp.authn	IdP messages related only to authentication
edu.internet2.middleware.shibboleth.common.relyingparty	IdP messages related to relying party configuration in use
edu.internet2.middleware.shibboleth.common.attribute	IdP messages related only to attribute resolution and filtering

Logging Levels

The logback system defines 5 logging levels (TRACE, DEBUG, INFO, WARN, ERROR). As you progress from the highest level (ERROR) to the lowest level (TRACE) the amount of information logged increases (dramatically so on the DEBUG and TRACE levels). Each level also logs all messages of the levels above it. For example, INFO also logs WARN and ERROR messages.

Mapped Diagnostic Context

Logback, the default logging service used in the IdP, supports a functionality known as the Mapped Diagnostic Context (MDC). Information stored in the MDC is available to every logging message (after the point the information is stored) and can be accessed by using the format `%mdc{KEY}`. Currently the IdP makes the following information available via the MDC:

MDC KEY	Description
idpSessionId	Unique identifier for the user's IdP session. This information is available once a session has been created.
JSESSIONID	The Servlet container JSESSIONID attribute
clientIP	The IP address of the remote user-agent. This is the user's browser for front-channel requests and the SP for back-channel requests.

Example Logging Pattern using the IdP Session ID

```
<Pattern>%date{HH:mm:ss.SSS} - %level [%logger:%line] - [%mdc
{idpSessionId}] - %msg%n</Pattern>
```

Example Logging Pattern using Client IP in Audit Log

```
<Pattern>%msg%mdc{clientIP}|%n</Pattern>
```

Examples

Preventing Access and Audit Log Entry Duplication

Because the appender associated with the idp-process.log is attached to the root logger, any messages logged to any logger will be output there by default. This includes entries logged to the idp-audit.log and idp-access.log, which will by default be duplicated in idp-process.log. This happens because, by default, when a Logback logging event fires at a given logger, it also fires at all the logger's parent loggers in the hierarchial logger tree. For more details see the Logback manual.

To prevent duplication of the audit and/or access log entries to idp-process.log, add the attribute `additivity="false"` to the Shibboleth-Access and Shibboleth-Audit logger elements:

```
<logger name="Shibboleth-Access" level="ALL" additivity="false">
    <appender-ref ref="IDP_ACCESS" />
</logger>

<logger name="Shibboleth-Audit" level="ALL" additivity="false">
    <appender-ref ref="IDP_AUDIT" />
</logger>
```

Adding a Warn-Level Log

You can create a log that captures all warnings, errors, etc. from the main process log, but excludes certain "common" warnings. The techniques shown can be adapted to filter messages out of the other logs, if so desired. The example also limits the size of the stack traces from Java exceptions.

```
<!-- Add an appender below the existing ones. -->
<appender name="IDP_WARN" class="ch.qos.logback.core.rolling.
RollingFileAppender">
    <!-- Example of filtering out messages by searching the log message. --
>
    <filter class="ch.qos.logback.core.filter.EvaluatorFilter">
       <evaluator>
            <expression>event.getMessage().indexOf("trying its failover
connector") >= 0</expression>
        </evaluator>
        <OnMismatch>NEUTRAL</OnMismatch>
        <OnMatch>DENY</OnMatch>
    </filter>
    <!-- Suppress anything above below WARN. -->
    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
        <level>WARN</level>
    </filter>
    <File>/opt/shibboleth-idp/logs/idp-warn.log</File>
    <rollingPolicy class="ch.qos.logback.core.rolling.
TimeBasedRollingPolicy">
        <FileNamePattern>/opt/shibboleth-idp/logs/idp-warn-%d{yyyy-MM-dd}.
log.gz</FileNamePattern>
    </rollingPolicy>
    <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">
        <charset>UTF-8</charset>
        <Pattern>%date{HH:mm:ss.SSS} - %level [%logger:%line] - %msg%n%ex
{3}</Pattern>
    </encoder>
</appender>

<!-- Modify root logging category at end of file and add new appender. -->
<root level="ERROR">
    <appender-ref ref="IDP_PROCESS" />
    <appender-ref ref="IDP_WARN" />
</root>
```