Was this helpful? 👍 👎

Maintenance guide

ON THIS PAGE

Maintenance guide

This guide describes how to perform recurring administrative operations in ForgeRock Access Management Java Agent.

About ForgeRock Identity Platform™ software

ForgeRock Identity Platform serves as the basis for our simple and comprehensive Identity and Access Management solution. For more information, visit https://www.pingidentity.com.

Auditing

Java Agent logs audit events for security, troubleshooting, and regulatory compliance. Logs are written in UTF-8. Store audit event logs in the following ways:

Remotely

Log audit events to the audit event handler configured in the AM realm. In an environment with several AM servers, agents write audit logs to the AM server that satisfies the agent request for client authentication or resource authorization.

Java Agent cannot log audit events remotely if:

- AM's Audit Logging Service is disabled.
- No audit event handler is configured in the realm where the agent is configured.
- All audit event handlers configured in the realm where the agent is configured are disabled.

For more information about audit logging in AM, see Setting up audit logging in AM's *Security guide*.

Locally

Log audit events in JSON format to a file in the agent installation directory, `/java_agents/agent_type/logs/audit/` .

Remotely and locally

Log audit events:

- To a file in the agent installation directory.
- To the audit event handler configured in the AM realm in which the agent profile is configured.

The following is an example of an agent log record:

```
{
   "timestamp":"...",
   "eventName":"AM-ACCESS-OUTCOME",
   "transactionId":"608831c4-7351-4277-8a5f-b1a83fe2277e",
   "userId":"id=demo,ou=user,dc=openam,dc=forgerock,dc=org",
   "trackingIds":[
      "fd5c8ccf-7d97-49ba-a775-76c3c06eb933-82095",
      "fd5c8ccf-7d97-49ba-a775-76c3c06eb933-82177"
   ],
   "component":"Java Policy Agent",
   "realm":"/",
   "server":{
      "ip":"127.0.0.1",
      "port":8020
   },
   "client":{
      "ip":"127.0.0.1",
      "port":55180
   },
   "request":{
      "protocol":"HTTP/1.1",
      "operation":"GET"
   },
   "http":{
      "request":{
         "secure":false,
         "method":"GET",
         "path":"http://my.example.com:8020/examples/",
         "headers":{
            "referer":[
               "https://am.example.com:8443/am/oauth2/authorize?scope[...]"
            ],
            "accept-language":[
               "en,en-US;q=0.8,da;q=0.6,fr;q=0.4"
            ],
            "host":[
               "my.example.com:8020"
            ],
            "upgrade-insecure-requests":[
               "1"
            ],
            "connection":[
               "keep-alive"
            ],
            "cache-control":[
               "max-age=0"
            ],
            "accept-encoding":[
               "gzip, deflate"
            ],
            "user-agent":[
               "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6)
AppleWebKit/537.36 (KHTML, like Gecko)[...]"
            ],
            "accept":[

"text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,
image/apng,*/*;q=0.8"
            ]
         },
         "cookies":{
            "am-auth-jwt":"eyJ0eXAiOiJKV1QiLCJhbGciOi[...]"
            "i18next":"en",
            "amlbcookie":"01",
            "iPlanetDirectoryPro":"Ts2zDkGUqgtkoxR[...]"
         }
      }
   },
   "response":{
      "status":"DENIED"
   },
   "_id":"fd5c8ccf-7d97-49ba-a775-76c3c06eb933-81703"
}
```

NOTE

Local audit logs do not have an `_id` attribute, which is an internal AM id.

The audit log format adheres to the log structure shared across the ForgeRock Identity Platform. For more information about the audit log format, see Audit log format in AM's *Security guide*.

Java Agent supports propagation of the transaction ID across the ForgeRock Identity Platform, using the HTTP header `X-ForgeRock-TransactionId`. For more information about configuring the header, see Configuring the trust transaction header system property in AM's *Security guide*.

Configure audit logging

By default, Java Agent does not write audit log records. To configure audit logging, perform this procedure. The agent in this example is in remote configuration mode.

1. In the AM console, select **Realms** > realm name > **Applications** > **Agents** > **Java** > agent name > **Global** > **Audit**.
2. In Audit Access Types, select the type of messages to log. For example, select `LOG_ALL` to log access allowed and access denied events.
3. In Audit Log Location, select whether to write the audit logs locally to the agent installation (`LOCAL`), remotely to AM (`REMOTE`), or to both places (`ALL`). For example, keep `REMOTE` to log audit events to the AM instances.
4. To log audit messages locally, enable Enable Local Audit Log Rotation to rotate the audit log files when they reach a maximum size.
5. If you enabled Enable Local Audit Log Rotation, specify the maximum size of the audit log files in Local Audit Log Rotation Size.

Monitoring

The following sections describe how to set up and maintain monitoring in your deployment, to ensure appropriate performance and service availability.

Monitoring the Prometheus endpoint

All ForgeRock products automatically expose a monitoring endpoint where Prometheus can scrape metrics, in a standard Prometheus format.

For information about installing and running Prometheus, see the Prometheus documentation.

By default, no special setup or configuration is required to access metrics at this endpoint. The following example queries the Prometheus endpoint for a route.

Tools such as Grafana are available to create customized charts and graphs based on the information collected by Prometheus. For more information on installing and running Grafana, see the Grafana website.

Prometheus performance metrics are provided by an endpoint configured in the protected web application's `web.xml` file. The endpoint must be accessible to the Prometheus server that uses the performance data.

When the example in Exposing endpoints for Prometheus and Common REST is configured, the Prometheus endpoint is available at `https://mydomain.example.com/myapp/metrics/prometheus`.

Monitoring the Common REST monitoring endpoint

Common REST performance metrics are provided by an endpoint configured in the protected web application’s `web.xml` file. The endpoint must be accessible to the REST client that uses the performance data.

When the example in Exposing endpoints for Prometheus and Common REST is configured, the Common REST performance monitoring endpoint is available at `https://mydomain.example.com/myapp/metrics/crest`.

Exposing endpoints for Prometheus and Common REST

Use the following procedure to expose endpoints for Prometheus or Common REST.

1. For each protected web application that is to expose metrics, edit the web application’s `web.xml` file.

   The following Tomcat example exposes a base endpoint named `/metrics`:

   ```
   <servlet>
      <servlet-name>AgentMonitoring</servlet-name>
      <servlet-class>org.forgerock.http.servlet.HttpFrameworkServlet</servlet-class>
      <init-param>
          <param-name>application-loader</param-name>
          <param-value>guice</param-value>
      </init-param>
   </servlet>
   <servlet-mapping>
       <servlet-name>AgentMonitoring</servlet-name>
       <url-pattern>/metrics/*</url-pattern>
   </servlet-mapping>
   ```

   Choose a name for the exposed base endpoint that does not conflict with any of the built-in agent endpoints, for example `/sunwCDSSORedirectURI`.

2. Allow access to the monitoring endpoint that is protected by the agent, in one of the following ways:

   a. Configure Not-Enforced URIs to create a not-enforced URI rule for the base endpoint.

      The following example rule allows access to the metrics base endpoint:

      ```
      */metrics/*
      ```

b. Configure Not-Enforced URIs, Not-Enforced Client IP List, and Not-Enforced Compound Rule Separator to create a compound not-enforced rule for the base endpoint.

The rule allows access from only the IP addresses of the REST clients or Prometheus server.

The following example rule allows access to the `/metrics` endpoint for HTTP requests that come from the IP address range from 192.168.1.1 to 192.168.1.3:

```
192.168.1.1-192.168.1.3 | */metrics/*
```

HTTP requests from other IP addresses are not able to access the metrics base endpoint.

c. Create an authorization policy in AM to restrict access to the metrics base endpoint.

Note that the metric base endpoint does not require login credentials. You can use a policy to ensure that requests to the endpoints are authenticated against the AM instance.

For more information, see Configuring policies in AM's *Authorization guide*.

3. If the monitoring endpoint is protected by AM policies, include the required credentials.

Writing metrics to CSV files

Configure Export Monitoring Metrics to CSV to write metric information to CSV files.

Summary of metric types

Timer fields

Common REST fields

Field	Description
`_id`	Metric ID.
`_type`	Metric type.
`count`	Number of events recorded for this metric.

Field	Description
`total`	Sum of the durations recorded for this metric.
`min`	Minimum duration recorded for this metric.
`max`	Maximum duration recorded for this metric.
`mean`	Average duration recorded for this metric.
`stddev`	Standard deviation of durations recorded for this metric.
`duration_units`	Units used for measuring the durations in the metric.
`p50`	50% of the durations recorded are at or below this value.
`p75`	75% of the durations recorded are at or below this value.
`p95`	95% of the durations recorded are at or below this value.
`p98`	98% of the durations recorded are at or below this value.
`p99`	99% of the durations recorded are at or below this value.
`p999`	99.9% of the durations recorded are at or below this value.
`m1_rate`	One-minute average rate.
`m5_rate`	Five-minute average rate.
`m15_rate`	Fifteen-minute average rate.
`mean_rate`	Average rate.
`rate_units`	Units used for measuring the rate of the metric.

NOTE

Duration-based values, such as `min`, `max`, and `p50`, are weighted towards newer data. By representing approximately the last five minutes of data, the timers make it easier to see recent changes in behavior, rather than a uniform average of recordings since the server was started.

The following is an example of the `requests.granted.not-enforced` metric from the Common REST endpoint:

```
{
  "_id" : "requests.granted.not-enforced",
  "_type" : "timer",
  "count" : 486,
  "total" : 80.0,
  "min" : 0.0,
  "max" : 1.0,
  "mean" : 0.1905615495053855,
  "stddev" : 0.39274399467782056,
  "duration_units" : "milliseconds",
  "p50" : 0.0,
  "p75" : 0.0,
  "p95" : 1.0,
  "p98" : 1.0,
  "p99" : 1.0,
  "p999" : 1.0,
  "m1_rate" : 0.1819109974890356,
  "m5_rate" : 0.05433445522996721,
  "m15_rate" : 0.03155662103953588,
  "mean_rate" : 0.020858521722211427,
  "rate_units" : "calls/second"
}
```

Prometheus fields

The Prometheus endpoint does not provide rate-based statistics, as rates can be calculated from the time-series data.

Field	Description
`# TYPE`	Metric ID, and type. Note that the `Timer` metric type is reported as a `Summary` type. Formatted as a comment.
`_count`	Number of events recorded.
`_total`	Sum of the durations recorded.
`{quantile="0.5"}`	50% of the durations are at or below this value.
`{quantile="0.75"}`	75% of the durations are at or below this value.
`{quantile="0.95"}`	95% of the durations are at or below this value.
`{quantile="0.98"}`	98% of the durations are at or below this value.

Field	Description
`{quantile="0.99"}`	99% of the durations are at or below this value.
`{quantile="0.999"}`	99.9% of the durations are at or below this value.

NOTE

Duration-based quantile values are weighted towards newer data. By representing approximately the last five minutes of data, the timers make it easier to see recent changes in behavior, rather than a uniform average of recordings since the server was started.

The following is an example of the `ja_requests{access=granted,decision=allowed-by-policy}` metric from the Prometheus endpoint:

```
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.5",} 0.013000000000000001
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.75",} 0.022000000000000002
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.95",} 0.022000000000000002
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.98",} 0.022000000000000002
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.99",} 0.022000000000000002
ja_requests_seconds{access="granted",decision="allowed-by-policy",quantile="0.999",} 1.1380000000000001
ja_requests_count{access="granted",decision="allowed-by-policy",} 7.0
ja_requests_seconds_total{access="granted",decision="allowed-by-policy",} 1.21
```

Gauge fields

Common REST fields

Metric for a numerical value that can increase or decrease. The value for a gauge is calculated when requested, and represents the state of Metric at that specific time.

Field	Description
`_id`	Metric ID.
`_type`	Metric type.
`value`	Current value of the metric.

The following is an example of the `jvm.used-memory` metric from the Common REST endpoint:

```
{
  "_id" : "jvm.used-memory",
  "_type" : "gauge",
  "value" : 2.13385216E9
}
```

Prometheus fields

Field	Description
`# TYPE`	Metric ID, and type. Formatted as a comment.
`{Metric ID}`	Current value. Large values may be represented in scientific E-notation.

The following is an example of the `ja_jvm_used_memory_bytes` metric from the Prometheus endpoint:

```
# TYPE ja_jvm_used_memory_bytes gauge
ja_jvm_used_memory_bytes 1.418723328E9
```

Distinct counter

Metric providing an estimate of the number of *unique* values recorded.

For example, this could be used to estimate the number of unique users who have authenticated, or unique client IP addresses.

> NOTE
>
> The `DistinctCounter` metric is calculated per instance of AM, and cannot be aggregated across multiple instances to get a site-wide view.

Common REST fields

Field	Description
`_id`	Metric ID.
`_type`	Metric type. Note that the `distinctCounter` type is reported as a `gauge` type. The output formats are identical.
`value`	Calculated estimate of the number of unique values recorded in the metric.

The following is an example of the `authentication.unique-uuid.success` metric from the Common REST endpoint:

```
{
  "_id" : "authentication.unique-uuid.success",
  "_type" : "gauge",
  "value" : 3.0
}
```

Prometheus fields

Field	Description
`# TYPE`	Metric ID, and type. Note that the `distinctCounter` type is reported as a `gauge` type. The output formats are identical. Formatted as a comment.
`{Metric ID}`	Calculated estimate of the number of unique values recorded in the metric.

The following is an example of the `ja_notenforced_ip_unmatched_cache_size` metric from the Prometheus endpoint:

```
# TYPE ja_notenforced_ip_unmatched_cache_size gauge
ja_notenforced_ip_unmatched_cache_size 3.0
```

Summary of exposed metrics

Java Agent exposes the monitoring metrics described in this section.

Audit handler metrics

Metric	Prometheus name	Description
`audit.access.generate`	`ja_audit_generate{topic=access}`	Time taken to generate an audit object. (Timer)
`audit.handler.<handler-type>.default.access.<outcome>`	`ja_audit{handler-type=<handler-type>,name=default,topic=access,outcome=<outcome>}`	Time taken to audit outcomes, both locally to the agent and remotely in AM. (Timer)

Labels:

`<handler-type>`

`am-delegate` . Remote auditing performed by AM. (Prometheus: `am_delegate`)

`json` . Local audit logging using JSON.

`<outcome>`

`success`

`failure`

Endpoint and REST SDK metrics

Metric	Prometheus name	Description
`session-info`	`ja_session_info`	Time taken to retrieve user session information from AM. (Timer)
`user-profile`	`ja_user_profile`	Time taken to retrieve the user profile information from AM. (Timer)
`policy-decision`	`ja_policy_decision`	Time taken to retrieve policy decisions from AM. (Timer)

JSON Web Token (JWT) metrics

Metric	Prometheus name	Description
`jwt.cache.size`	`ja_jwt_cache_size`	Size of the JWT cache. (Gauge)

Metric	Prometheus name	Description
`jwt.cache.eviction`	`ja_jwt_cache_eviction`	The eviction count for the JWT cache. (Gauge)
`jwt.cache.load-count`	`ja_jwt_cache_load_count`	The load count for the JWT cache. (Gauge)
`jwt.cache.load-time`	`ja_jwt_cache_load_time`	The load time for the JWT cache, in milliseconds. (Gauge)
`jwt.cache.hit`	`ja_jwt_cache{outcome=hit}`	The hit count for the JWT cache. (Gauge)
`jwt.cache.miss`	`ja_jwt_cache{outcome=miss}`	The miss count for the JWT cache. (Gauge)

JVM metrics

> TIP
>
> To get Metric name used by Prometheus, prepend `ja_` to the names below, and replace period (`.`) and hyphen (`-`) characters with underscore (`_`) characters. For example, the `jvm.available-cpus` metric is named `ja_jvm_available_cpus` in Prometheus.

Name	Description
`jvm.available-cpus`	Number of processors available to the Java virtual machine. (Gauge)
`jvm.class-loading.loaded`	Number of classes loaded since the Java virtual machine started. (Gauge)
`jvm.class-loading.unloaded`	Number of classes unloaded since the Java virtual machine started. (Gauge)
`jvm.garbage-collector.PS-MarkSweep.count`	Number of collections performed by the "parallel scavenge mark sweep" garbage collection algorithm. (Gauge)
`jvm.garbage-collector.PS-MarkSweep.time`	Approximate accumulated time taken by the "parallel scavenge mark sweep" garbage collection algorithm. (Gauge)

Name	Description
`jvm.garbage-collector.PS-Scavenge.count`	Number of collections performed by the "parallel scavenge" garbage collection algorithm. (Gauge)
`jvm.garbage-collector.PS-Scavenge.time`	Approximate accumulated time taken by the "parallel scavenge" garbage collection algorithm. (Gauge)
`jvm.memory-usage.heap.init`	Amount of heap memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.heap.max`	Maximum amount of heap memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.heap.committed`	Amount of heap memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.heap.used`	Amount of heap memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.total.init`	Amount of memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.total.max`	Maximum amount of memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.non-heap.init`	Amount of non-heap memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.non-heap.max`	Maximum amount of non-heap memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.non-heap.committed`	Amount of non-heap memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.non-heap.used`	Amount of non-heap memory used by the Java virtual machine. (Gauge)

Name	Description
`jvm.memory-usage.pools.Code-Cache.init`	Amount of "code cache" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Code-Cache.max`	Maximum amount of "code cache" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Code-Cache.committed`	Amount of "code cache" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.Code-Cache.used`	Amount of "code cache" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.init`	Amount of "compressed class space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.init`	Maximum amount of "compressed class space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.committed`	Amount of "compressed class space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.used`	Amount of "compressed class space" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.Metaspace.init`	Amount of "metaspace" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Metaspace.max`	Maximum amount of "metaspace" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Metaspace.committed`	Amount of "metaspace" memory that is committed for the Java virtual machine to use. (Gauge)

Name	Description
`jvm.memory-usage.pools.Metaspace.used`	Amount of "metaspace" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.init`	Amount of "parallel scavenge eden space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.max`	Maximum amount of "parallel scavenge eden space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.committed`	Amount of "parallel scavenge eden space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.used-after-gc`	Amount of "parallel scavenge eden space" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.used`	Amount of "parallel scavenge eden space" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.init`	Amount of "parallel scavenge old generation" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.max`	Maximum amount of "parallel scavenge old generation" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.committed`	Amount of "parallel scavenge old generation" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.used-after-gc`	Amount of "parallel scavenge old generation" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)

Name	Description
`jvm.memory-usage.pools.PS-Old-Gen.used`	Amount of "parallel scavenge old generation" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.init`	Amount of "parallel scavenge survivor space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.max`	Maximum amount of "parallel scavenge survivor space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.committed`	Amount of "parallel scavenge survivor space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.used-after-gc`	Amount of "parallel scavenge survivor space" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.used`	Amount of "parallel scavenge survivor space" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.total.committed`	Amount of memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.total.used`	Amount of memory used by the Java virtual machine. (Gauge)
`jvm.thread-state.blocked.count`	Number of threads in the BLOCKED state. (Gauge)
`jvm.thread-state.count`	Number of live threads including both daemon and non-daemon threads. (Gauge)
`jvm.thread-state.daemon.count`	Number of live daemon threads. (Gauge)
`jvm.thread-state.new.count`	Number of threads in the NEW state. (Gauge)

Name	Description
`jvm.thread-state.runnable.count`	Number of threads in the RUNNABLE state. (Gauge)
`jvm.thread-state.terminated.count`	Number of threads in the TERMINATED state. (Gauge)
`jvm.thread-state.timed_waiting.count`	Number of threads in the TIMED_WAITING state. (Gauge)
`jvm.thread-state.waiting.count`	Number of threads in the WAITING state. (Gauge)

Not-enforced rule metrics

Metric	Prometheus name	Description
`notenforced-uri.matched.cache.size`	`ja_notenforced_uri_matched_cache_size`	Size of the not-enforced URI matched cache. (Gauge)
`notenforced-uri.matched.cache.eviction`	`ja_notenforced_uri_matched_cache_eviction`	Eviction count for the not-enforced URI matched cache. (Gauge)
`notenforced-uri.matched.cache.load-count`	`ja_notenforced_uri_matched_cache_load_count`	Load count for the not-enforced URI matched cache. (Gauge)
`notenforced-uri.matched.cache.load-time`	`ja_notenforced_uri_matched_cache_load_time`	Load time for the not-enforced URI matched cache, in milliseconds. (Gauge)
`notenforced-uri.matched.cache.hit`	`ja_notenforced_uri_matched_cache{outcome=hit}`	Hit count for the not-enforced URI matched cache. (Gauge)
`notenforced-uri.matched.cache.miss`	`ja_notenforced_uri_matched_cache{outcome=miss}`	Miss count for the not-enforced URI matched cache. (Gauge)
`notenforced-uri.unmatched.cache.size`	`ja_notenforced_uri_unmatched_cache_size`	Size of the not-enforced URI unmatched cache. (Gauge)

Metric	Prometheus name	Description
`notenforced-uri.unmatched.cache.eviction`	`ja_notenforced_uri_unmatched_cache_eviction`	Eviction count for the not-enforced URI unmatched cache. (Gauge)
`notenforced-uri.unmatched.cache.load-count`	`ja_notenforced_uri_unmatched_cache_load_count`	Load count for the not-enforced URI unmatched cache. (Gauge)
`notenforced-uri.unmatched.cache.load-time`	`ja_notenforced_uri_unmatched_cache_load_time`	Load time for the not-enforced URI unmatched cache, in milliseconds. (Gauge)
`notenforced-uri.unmatched.cache.hit`	`ja_notenforced_uri_unmatched_cache{outcome=hit}`	Hit count for the not-enforced URI unmatched cache. (Gauge)
`notenforced-uri.unmatched.cache.miss`	`ja_notenforced_uri_unmatched_cache{outcome=miss}`	Miss count for the not-enforced URI unmatched cache. (Gauge)
`notenforced-ip.matched.cache.size`	`ja_notenforced_ip_matched_cache_size`	Size of the not-enforced IP matched cache. (Gauge)
`notenforced-ip.matched.cache.eviction`	`ja_notenforced_ip_matched_cache_eviction`	Eviction count for the not-enforced IP matched cache. (Gauge)
`notenforced-ip.matched.cache.load-count`	`ja_notenforced_ip_matched_cache_load_count`	Load count for the not-enforced IP matched cache. (Gauge)
`notenforced-ip.matched.cache.load-time`	`ja_notenforced_ip_matched_cache_load_time`	Load time for the not-enforced IP matched cache, in milliseconds. (Gauge)
`notenforced-ip.matched.cache.hit`	`ja_notenforced_ip_matched_cache{outcome=hit}`	Hit count for the not-enforced IP matched cache. (Gauge)
`notenforced-ip.matched.cache.miss`	`ja_notenforced_ip_matched_cache{outcome=miss}`	Miss count for the not-enforced IP matched cache. (Gauge)

Metric	Prometheus name	Description
`notenforced-ip.unmatched.cache.size`	`ja_notenforced_ip_unmatched_cache_size`	Size of the not-enforced IP unmatched cache. (Gauge)
`notenforced-ip.unmatched.cache.eviction`	`ja_notenforced_ip_unmatched_cache_eviction`	Eviction count for the not-enforced IP unmatched cache. (Gauge)
`notenforced-ip.unmatched.cache.load-count`	`ja_notenforced_ip_unmatched_cache_load_count`	Load count for the not-enforced IP unmatched cache. (Gauge)
`notenforced-ip.unmatched.cache.load-time`	`ja_notenforced_ip_unmatched_cache_load_time`	Load time for the not-enforced IP unmatched cache, in milliseconds. (Gauge)
`notenforced-ip.unmatched.cache.hit`	`ja_notenforced_ip_unmatched_cache{outcome=hit}`	Hit count for the not-enforced IP unmatched cache. (Gauge)
`notenforced-ip.unmatched.cache.miss`	`ja_notenforced_ip_unmatched_cache{outcome=miss}`	Miss count for the not-enforced IP unmatched cache. (Gauge)
`notenforced-compound.matched.cache.size`	`ja_notenforced_compound_matched_cache_size`	Size of the not-enforced compound matched cache. (Gauge)
`notenforced-compound.matched.cache.eviction`	`ja_notenforced_compound_matched_cache_eviction`	Eviction count for the not-enforced compound matched cache. (Gauge)
`notenforced-compound.matched.cache.load-count`	`ja_notenforced_compound_matched_cache_load_count`	Load count for the not-enforced compound matched cache. (Gauge)
`notenforced-compound.matched.cache.load-time`	`ja_notenforced_compound_matched_cache_load_time`	Load time for the not-enforced compound matched cache, in milliseconds. (Gauge)
`notenforced-compound.matched.cache.hit`	`ja_notenforced_compound_matched_cache{outcome=hit}`	Hit count for the not-enforced compound matched cache. (Gauge)

Metric	Prometheus name	Description
`notenforced-compound.matched.cache.miss`	`ja_notenforced_compound_matched_cache{outcome=miss}`	Miss count for the not-enforced compound matched cache. (Gauge)
`notenforced-compound.unmatched.cache.size`	`ja_notenforced_compound_unmatched_cache_size`	Size of the not-enforced compound unmatched cache. (Gauge)
`notenforced-compound.unmatched.cache.eviction`	`ja_notenforced_compound_unmatched_cache_eviction`	Eviction count for the not-enforced compound unmatched cache. (Gauge)
`notenforced-compound.unmatched.cache.load-count`	`ja_notenforced_compound_unmatched_cache_load_count`	Load count for the not-enforced compound unmatched cache. (Gauge)
`notenforced-compound.unmatched.cache.load-time`	`ja_notenforced_compound_unmatched_cache_load_time`	Load time for the not-enforced compound unmatched cache, in milliseconds. (Gauge)
`notenforced-compound.unmatched.cache.hit`	`ja_notenforced_compound_unmatched_cache{outcome=hit}`	Hit count for the not-enforced compound unmatched cache. (Gauge)
`notenforced-compound.unmatched.cache.miss`	`ja_notenforced_compound_unmatched_cache{outcome=miss}`	Miss count for the not-enforced compound unmatched cache. (Gauge)

Policy decision metrics

Metric	Prometheus name	Description
`policy-decision.cache.size`	`ja_policy_decision_cache_size`	Size of the policy decision cache. (Gauge)
`policy-decision.cache.eviction`	`ja_policy_decision_cache_eviction`	Eviction count for the policy decision cache. (Gauge)
`policy-decision.cache.load-count`	`ja_policy_decision_cache_load_count`	Load count for the policy decision cache. (Gauge)

Metric	Prometheus name	Description
`policy-decision.cache.load-time`	`ja_policy_decision_cache_load_time`	Load time for the policy decision cache, in milliseconds. (Gauge)
`policy-decision.cache.hit`	`ja_policy_decision_cache{outcome=hit}`	Hit count for the policy decision cache. (Gauge)
`policy-decision.cache.miss`	`ja_policy_decision_cache{outcome=miss}`	Miss count for the policy decision cache. (Gauge)

POST data preservation metrics

Metric	Prometheus name	Description
`pdp.cache.size`	`ja_pdp_cache_size`	Size of the POST data preservation cache. (Gauge)
`pdp.cache.eviction`	`ja_pdp_cache_eviction`	Eviction count for the POST data preservation cache. (Gauge)
`pdp.cache.load-count`	`ja_pdp_cache_load_count`	Load count for the POST data preservation cache. (Gauge)
`pdp.cache.load-time`	`ja_pdp_cache_load_time`	Load time for the POST data preservation cache, in milliseconds. (Gauge)
`pdp.cache.hit`	`ja_pdp_cache{outcome=hit}`	Hit count for the POST data preservation cache. (Gauge)
`pdp.cache.miss`	`ja_pdp_cache{outcome=miss}`	Miss count for the POST data preservation cache. (Gauge)

Request metrics

Metric	Prometheus name	Description
`requests.<access>.<decision>`	`ja_requests{access=<access>,decision=<decision>}`	Rate of granted/denied requests and their decision. (Timer)

Labels:

`<access>`

`granted`

`denied`

`<decision>`

`not-enforced` : Request matched a not-enforced rule.

`no-valid-token` : Request did not have a valid SSO token or an OpenID Connect JWT.

`allowed-by-policy` : Request matched a policy, which allowed access.

`denied-by-policy` : Request matched a policy, which denied access.

`am-unavailable` : The AM instance was not reachable.

`agent-exception` : An internal error (exception) occurred within the agent.

Session information metrics

Metric	Prometheus name	Description
`session-info.cache.size`	`ja_session_info_cache_size`	Size of the session information cache. (Gauge)
`session-info.cache.eviction`	`ja_session_info_cache_eviction`	Eviction count for the session information cache. (Gauge)
`session-info.cache.load-count`	`ja_session_info_cache_load_count`	Load count for the session information cache. (Gauge)
`session-info.cache.load-time`	`ja_session_info_cache_load_time`	Load time for the session information cache, in milliseconds. (Gauge)
`session-info.cache.hit`	`ja_session_info_cache{outcome=hit}`	Hit count for the session information cache. (Gauge)

Metric	Prometheus name	Description
`session-info.cache.miss`	`ja_session_info_cache{outcome=miss}`	Miss count for the session information cache. (Gauge)

SSO token to JWT exchange metrics

Metric	Prometheus name	Description
`sso-exchange.cache.size`	`ja_sso_exchange_cache_size`	Size of the SSO token exchange cache. (Gauge)
`sso-exchange.cache.eviction`	`ja_sso_exchange_cache_eviction`	Eviction count for the SSO token exchange cache. (Gauge)
`sso-exchange.cache.load-count`	`ja_sso_exchange_cache_load_count`	Load count for the SSO token exchange cache. (Gauge)
`sso-exchange.cache.load-time`	`ja_sso_exchange_cache_load_time`	Load time for the SSO token exchange, in milliseconds. (Gauge)
`sso-exchange.cache.hit`	`ja_sso_exchange_cache{outcome=hit}`	Hit count for the SSO token exchange cache. (Gauge)
`sso-exchange.cache.miss`	`ja_sso_exchange_cache{outcome=miss}`	Miss count for the SSO token exchange cache. (Gauge)

Websocket metrics

Metric	Prometheus name	Description
`websocket.last-received`	`ja_websocket_last_received`	Number of milliseconds since anything was received over the websocket, for example a ping or a notification. (Gauge)

Metric	Prometheus name	Description
`websocket.last-sent`	`ja_websocket_last_sent`	Number of milliseconds since anything was sent over the websocket. (Gauge)
`websocket.config-change.received`	`ja_websocket_config_change_received`	Number of configuration change notifications received. Note that some may be ignored if the realm or agent name are not applicable. (DistinctCounter)
`websocket.config-change.processed`	`ja_websocket_config_change_processed`	Number of configuration change notifications processed, that were not ignored. (DistinctCounter)
`websocket.policy-change.received`	`ja_websocket_policy_change_received`	Number of policy change notifications received. Note that some may be ignored if the realm or agent name are not applicable. (DistinctCounter)
`websocket.policy-change.processed`	`ja_websocket_policy_change_processed`	Number of policy change notifications processed, that were not ignored. (DistinctCounter)
`websocket.session-logout.received`	`ja_websocket_session_logout_received`	Number of session logout notifications received. Note that some may be ignored if the realm or agent name are not applicable. (DistinctCounter)
`websocket.session-logout.processed`	`ja_websocket_session_logout_processed`	Number of session logout notifications processed, that were not ignored. (DistinctCounter)

Metric	Prometheus name	Description
`websocket.ping-pong`	`ja_websocket_ping_pong`	Ping/pong round trip time. (Timer)

Logging

Logs in Java Agent and third-party dependencies are recorded using the Logback implementation of the Simple Logging Facade for Java (SLF4J) API.

By default, the logback configuration is stored in `/path/to/java_agents/agent_type/Agent_n/config/agent-logback.xml`. For information about how to change the location, see Agent configuration.

The agent supports the following log levels:

- `TRACE`
- `INFO`
- `WARN`
- `ERROR`
- `ON` (deprecated, this setting uses the `TRACE` log level)
- `OFF` (deprecated, this setting uses the `ERROR` log level)

For a full description of logging options, see the Logback website.

Setting the log level

The agent maintains the **last** log level it reads from either `agent-logback.xml` or Agent Debug Level.

TIP

To debug code during the property-reading phase of startup, set the log level in `agent-logback.xml` to `TRACE`.

Remote configuration mode

In remote configuration mode, the agent sets the log level as follows:

- At start up, the agent reads `agent-logback.xml`, and uses the configured log level.
- When the agent receives the first request, it reads `AgentBootstrap.properties`. If org.forgerock.agents.debug.level specifies a different log level, the agent changes

the log level.

- The agent then reads the AM properties. If Agent Debug Level specifies different log level, the agent changes the log level.
- While the agent is running, if either of the following events occurs the agent changes the log level to the value configured by the event:
  - Agent Debug Level is updated in AM. The log level is changed immediately.
  - `agent-logback.xml` is updated in the agent, and the update is taken into account as described in Changing `agent-logback.xml`.

Local configuration mode

In local configuration mode, the agent sets the log level as follows:

- At start up, the agent reads `agent-logback.xml`, and uses the configured log level.
- When the agent receives the first request, it reads `AgentBootstrap.properties` and then `AgentConfiguration.properties`.
  - If either file specifies a log level in org.forgerock.agents.debug.level, the agent uses that log level.
  - If both files specify a log level, the agent uses the **last** log level it reads, from `AgentConfiguration.properties`.
- While the agent is running, if either of the following events occurs the agent changes the log level to the value configured by the event:
  - The property org.forgerock.agents.debug.level is changed in `AgentConfiguration.properties`, and the time configured by the property org.forgerock.agents.config.reload.seconds elapses.
  - `agent-logback.xml` is updated in the agent, and the update is taken into account as described in Changing `agent-logback.xml`.

Default logging in `agent-logback.xml`

By default, `agent-logback.xml` is configured as follows:

- When the agent starts, log messages are recorded for the agent.
- Log messages are written to a text file in `/path/to/java_agents/`*`agent_type`*`/Agent_`*`n`*`/logs/debug`, where `/path/to/java_agents/`*`agent_type`*`/Agent_`*`n`* is the installation directory.
- The log level is `ERROR`.
- Logs are rolled over once a day, or more frequently if the file reaches 100 MB.

All aspects of the logs are defined by the following reference `agent-logback.xml` deployed by the agent installer:

```
<configuration debug="true" scan="true" scanPeriod="60 seconds">
    <appender name="ROLLING" class="org.forgerock.agents.shaded.ch.qos.logback.core.rolling.RollingFileAppender">

        <file>/full/path/to/logs/debug/directory/log.txt</file>
        <rollingPolicy class="org.forgerock.agents.shaded.ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">

            <!-- rollover daily -->
            <fileNamePattern>log-%d{yyyy-MM-dd}.%i.txt</fileNamePattern>

            <!-- each file should be at most 100MB, keep 30 days worth of history, but at most 10GB -->
            <maxFileSize>100MB</maxFileSize>
            <maxHistory>30</maxHistory>
            <totalSizeCap>10GB</totalSizeCap>
        </rollingPolicy>
        <encoder>
            <pattern>%level %msg%n</pattern>
        </encoder>
    </appender>

    <root level="ERROR">
        <appender-ref ref="ROLLING" />
    </root>
</configuration>
```

Changing `agent-logback.xml`

To change the logging behavior, update `agent-logback.xml` .

To take the change into account, stop and restart the agent, or include a scan and an interval configuration in `agent-logback.xml` . In the following example, `agent-logback.xml` is scanned every 60 seconds:

```
<configuration scan="true" scanPeriod="60 seconds" ... >
  ...
```

For a full description of the options for logging, see the Logback website.

Use shaded Logback classnames, to ensure that the agent uses a unique classname compared to other deployed filters and web applications in the container. For example, use

```
org.forgerock.agents.shaded.ch.qos.logback.core.rolling.RollingFil
eAppender
```

instead of

```
ch.qos.logback.core.rolling.RollingFileAppender
```

Managing errors in `agent-logback.xml`

If `agent-logback.xml` is removed and not replaced, or if a replacement `agent-logback.xml` contains errors, logs are written to the standard output for the container. For example, Tomcat logs are written to `catalina.out`.

To log an error when `agent-logback.xml` fails to load, include a debug configuration:

```
<configuration debug="true" ... >
  ...
```

When `agent-logback.xml` contains errors, messages like these written to the standard output for the container.

```
ERROR in ch.qos.logback.core.joran.spi.Interpreter@20:72 …
ERROR in ch.qos.logback.core.joran.action.AppenderRefAction …
```

Changing the log level in agent-logback.xml

The global log level is set by default to `ERROR` by the following line of the default `agent-logback.xml`:

```
<root level="ERROR">
```

To change the global log level, configure the same line with another log level and take the changed file into account, as described in Changing `agent-logback.xml`.

To change the log level for a single object type without changing it for the rest of the configuration, add a logger defined by the shaded classname of the object, and set its log level.

The following line in `agent-logback.xml` changes the log level for evaluating not-enforced rules, but does not change the log level of other classes or packages:

```
<logger
name="org.forgerock.agents.shaded.com.sun.identity.agents.common.N
otEnforcedRuleHelper" level="TRACE" />
```

Changing the character set and format of log messages

By default, logs use the system default character set where the agent is running. To use a different character set, or change the pattern of the log messages, edit `agent-logback.xml` to change the `encoder` configuration.

The following lines add the date to log messages, and change the character set:

```
<encoder>
   <pattern>%d{yyyyMMdd-HH:mm:ss} | %-5level | %thread |
%logger{20} | %message%n%xException</pattern>
   <charset>UTF-8</charset>
</encoder>
```

Limiting repeated log messages

To keep log files clean and readable, and to prevent log overflow attacks, limit the number of repeat log messages. Add a custom `agent-logback.xml` with a `DuplicateMessageFilter`. This filter detects duplicate messages, and after the specified number of repetitions, drops repeated messages.

The following example allows 5 repetitions of a log message, holds the following 10 repeated messages in the cache, and then discards subsequent repeated messages:

```
<turboFilter
class="org.forgerock.agents.shaded.ch.qos.logback.classic.turbo.Du
plicateMessageFilter" allowedRepetitions="5" CacheSize="10" />
```

The DuplicateMessageFilter has the following limitations:

- Filters out **all** duplicate messages. It does not filter per logger, or logger instance, or logger name.
- Detects repetition of raw messages, meaning that some non-identical messages are considered as repetition.

For example, when the agent is running at `TRACE` level, the following message is written to the logs every 20 seconds when there is no other activity over the Websocket:

```
TRACE 08:22:00 (20) Sending ping to check connection is still
alive
```

This message is produced by the logging statement:

```
logger.trace("{} ({}) Sending ping to check connection is
still alive", timeStamp, lastSent.getSeconds());
```

Because the message text `{} ({}) Sending ping to check connection is still alive` is always the same, it is considered as repeat. Only the first 5 repeats are logged.

- Does not limit the lifespan of the cache. After the specified number of repetitions is reached, the repeated log messages never appear again, even if they are frequently hit.

Changing the log level in the AM console

To change the log level in the AM console

1. In the AM console, go to **Realms** > realm name > **Applications** > **Agents** > **Java**, and select your Java Agent.
2. On the **Global** tab, select **Agent Debug Level**, and select a level.

Alternatively, set Agent Debug Level as a custom property.

Notifications

AM sends the following notifications to Java Agent through WebSockets:

Configuration notifications

When the administrator makes a change to an agent configuration property, AM sends a notification to the agent. The agent flushes the configuration cache, and rereads the agent profile from AM. For more information about the cache, see Configuration cache.

Configuration notifications apply when you store the agent profile in AM's configuration data store. For information, see Enable Notifications of Agent

Configuration Change.

Session notifications

When a client logs out, or a CTS-based session expires, AM sends a notification to the agent to remove that entry from the session cache. For information, see Enable Notification of Session Logout. For more information about the cache, see Session Cache.

Policy notifications

When an administrator changes a policy, AM sends a notification to the agent to flush the policy cache. For information, see Enable Notification of Policy Changes. For more information about the cache, see Policy Cache.

Notifications are enabled by default. To disable notifications, unsubscribe separately to each type of notification. If Autonomous mode is `true`, notifications and many other features are automatically disabled.

In configurations with load balancers and reverse proxies, make sure that the load balancers and reverse proxies support WebSockets.

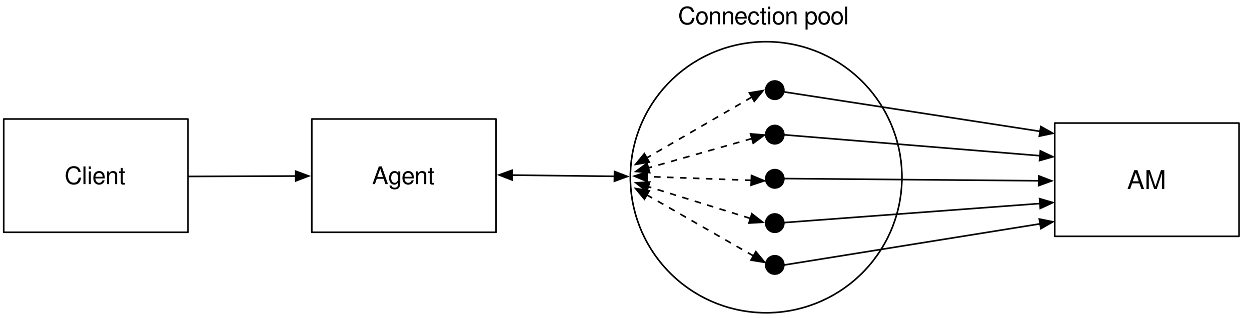
For more information about properties that configure notifications, see Notifications in the *Properties reference*.

Tuning connections

Use a connection pool between Java Agent and AM to reuse connections, and so reduce the overhead of creating new connections.

Connection pooling can improve performance in some circumstances. Test and tune the performance of your deployment with connection pooling before you use it in a production environment.

The following image shows the architecture of a connection pool:

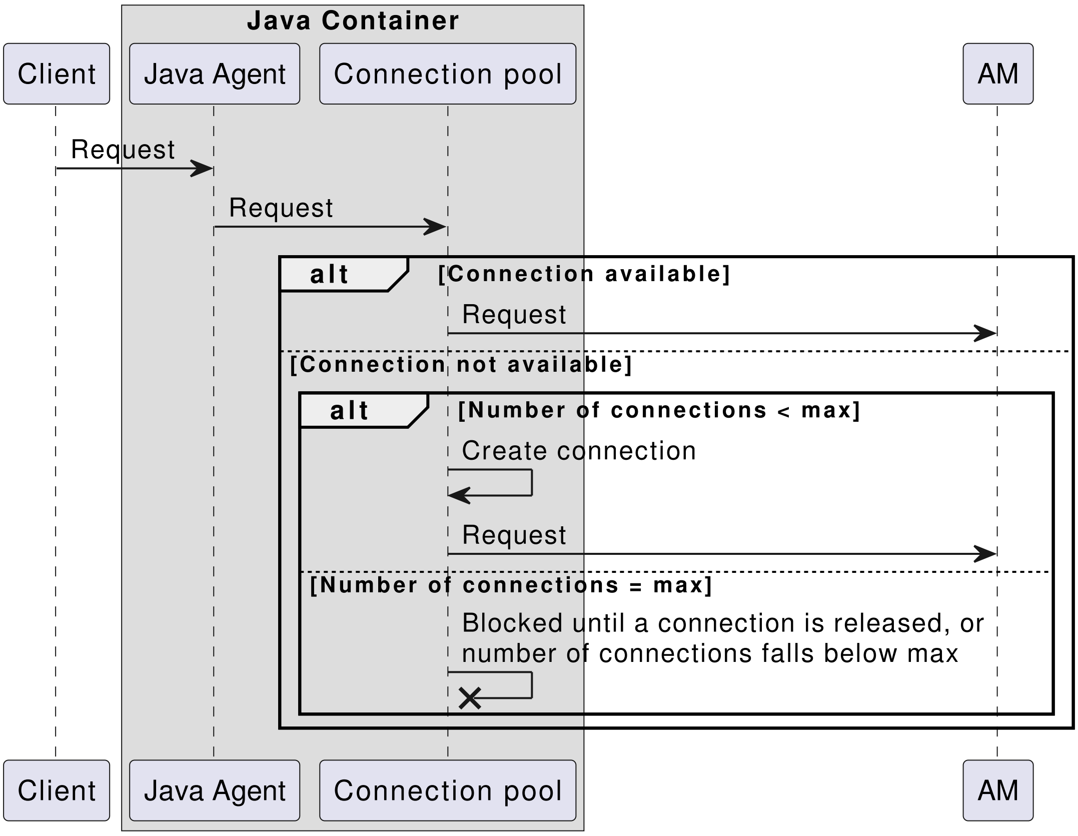


When a client makes a request, the agent intercepts the request, and uses the connection pool to connect to AM. If a connection is available, the agent uses that connection. The client is unaware of the connection reuse.

If a connection is not available, and the maximum number of connections is not reached, the agent creates and uses a new connection. When the maximum number of allowed connections is reached, the request waits until an existing connection is released, or a new connection can be created.

When the request is complete, the agent closes the connection to the pool but the physical connection is retained by the pool. The connection is then available to requests with the same connection parameters.

The following image shows the flow of information when a request is treated in a connection pool:


Java Container
Client
Java Agent
Connection pool
AM
Request
Request
alt
[Connection available]
Request
[Connection not available]
alt
[Number of connections < max]
Create connection
Request
[Number of connections = max]
Blocked until a connection is released, or
number of connections falls below max
Client
Java Agent
Connection pool
AM

When Enable Connection Pooling is `true`, use the following properties to tune the connection pool:

Property	Description
Max HTTP Connection Count	By default, the connection pool can contain up to 1000 HTTP connections. Change the value of this property according to the AM load, as follows: • If AM is likely to be overloaded, reduce the maximum number of connections to reduce the load on AM. • If AM is not likely to be overloaded, use the default or a higher value so that connections are available when they are requested.

Property	Description
HTTP Connection Timeout	By default, connections wait for up to 10 seconds for a response before they are considered stale. Tune this property to allow enough time for systems to respond, and therefore prevent unnecessary failures, but to minimize the time to wait after a network failure.
HTTP Socket Timeout	By default, when a continuous data flow between the connection pool and AM is interrupted, the socket is considered stale. Tune this property to allow enough time for systems to respond, and therefore prevent unnecessary failures, but to minimize the time to wait after a socket failure.
Enable HTTP Connection Reuse	By default, after a request is complete the connection is returned to the connection pool. The physical connection is retained and can be reused by another request with the same connection parameters. Reuse connections to reduce the overhead of creating a new connection each time one is requested. If you are using the connection pool only as a throttling mechanism, to limit the total number of simultaneous connections, it is not necessary to reuse connections.
Enable HTTP Connection State	When Enable HTTP Connection Reuse is `true`, connection reuse is enabled by default for Apache HTTP Client.
Enable HTTP Retry	By default, requests are retried up to three times after failure. If requests are likely to pass on retry, enable this property to reduce the overhead of unnecessarily killing and remaking connections. If requests are likely to fail on retry, disable this property to reduce the retry time.

Was this helpful? 👍 👎