Maintenance guide

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

About ForgeRock Identity Platform™ software

ForgeRock® Identity Platform serves as the basis for our simple and comprehensive Identity and Access Management solution. We help our customers deepen their relationships with their customers, and improve the productivity and connectivity of their employees and partners. For more information about ForgeRock and about the platform, see https://www.forgerock.com.

Auditing

Web 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.

Web 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, refer to Setting up audit logging in AM's *Security guide*.

Locally

Log audit events in JSON format to `web_agents/agent_type/instances/agent_n/logs/debug/audit.log` . The following is an example agent log file: `/web_agents/nginx22_agent/instances/agent_1/logs/audit/audit.log` .

Remotely and locally

Log audit events:

- To `web_agents/agent_type/instances/agent_n/logs/debug/audit.log`
- To the audit event handler configured in the AM realm in which the agent profile is configured.

The following is an example agent log record:

```
{
   "timestamp":"2017-10-30T11:56:57Z",
   "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":"Web Policy Agent",
   "realm":"/",
   "server":{
      "ip":"127.0.0.1",
      "port":8020
   },
   "request":{
      "protocol":"HTTP/1.1",
      "operation":"GET"
   },
   "http":{
      "request":{
         "secure":false,
         "method":"GET",
         "path":"http://my.example.com:8020/examples/",
         "cookies":{
            "am-auth-jwt":"eyJ0eXAiOiJKV1QiLCJhbGciOi[...]"
            "i18next":"en",
            "amlbcookie":"01",
            "iPlanetDirectoryPro":"Ts2zDkGUqgtkoxR[...]"
         }
      }
   },
   "response":{
      "status":"DENIED"
   },
   "_id":"fd5c8ccf-7d97-49ba-a775-76c3c06eb933-81703"
}
```

NOTE

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, refer to Audit log format in AM's *Security guide*.

Web 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, refer to Configuring the trust transaction header system property in AM's *Security guide*.

By default, Web Agent does not write audit log records. To configure audit logging, perform the following procedure:

Configure audit logging

By default, Web 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 `agent.conf`, set values for the following properties:
   - Local Agent Audit File Name
   - Local Audit Log Rotation Size

   NOTE

   After changing a bootstrap property, restart the web server where the agent runs.

2. On the AM admin UI, select **REALMS** > Realm Name > **Applications** > **Agents** > **Web** > Agent Name.
3. On the **Global** tab, select the following options to configure audit:
   - Agent Debug Level
   - Audit Access Types
   - Audit Log Location

Monitoring

IMPORTANT

IMPORTANT

The monitoring endpoint described in this section is deprecated. Use it only for diagnostics, in conjunction with Forgerock support.

A monitoring endpoint provides access to metrics for operations within the agent and between the agent an AM.

The monitoring endpoint is protected by HTTP Basic Authentication; to access it, provide the agent URL, and the AM agent profile name and password. Always use HTTPS for secure connections to client applications.

Metrics are displayed as a JSON response, with the fields described in Summary of monitoring metrics.

Access the monitoring endpoint

1. Install a Web Agent as described in the Installation guide, and use the agent to protect a web application. For example, set up the example in Policy enforcement.
2. Access the agent monitoring endpoint as follows, where `https://agent.example.com:443` is the agent URL, and `web-agent` is the AM agent profile name.

```
$ curl https://agent.example.com:443/agent/monitor -u web-agent

Enter host password for user 'web-agent':
```

3. Enter the AM agent profile password to display the metrics:

```
{
  "cache-invalidation": {
    "policy": 0,
    "profile": 1
  },
  "policy-decisions": {
    "neu": 0,
    "local": 0,
    "remote": 2,
    "cache": 0
    },
  "gc": {
    "runs": 1,
    "released": 0,
    "release-deferred": 0,
    "fill": 0.000000
  },
  "cache-operations": {
    "writes": 0,
    "rewrites": 2,
    "reads": 2,
    "misses": 0,
    "deletes": 0,
    "free-deferred": 0,
    "write-faults": 0,
    "expired": 0
  },
  "connections": {
    "added": 2,
    "reused": 3
  }
}
```

Summary of monitoring metrics

Metric	Submetric	Count of
`cache-invalidation`	`policy`	Number of policy change notifications received from AM.
	`profile`	Number of agent configuration change notifications received from AM.
`policy-decisions`	`neu`	Number of requests that were not enforced by the agent because of the not-enforced URL lists.
	`local`	Number of policy decisions the agent makes locally.
	`remote`	Number of policy decisions the agent requests from AM.
	`cache`	Number of policy decisions the agent takes from the cache.

Metric	Submetric	Count of
`gc`	`runs`	Number of garbage collection runs.
	`released`	Number of cache entries released during garbage collection runs.
	`release-defered`	Number of entries with release deferred until the next garbage collection run.
	`fill`	Floating point value between 0 and 1, representing the proportion of cache that is free after the most the recent garbage collection.
`cache-operations`	`writes`	Number of writes to cache.
	`rewrites`	Number of updates to cache.
	`reads`	Number of reads from cache.
	`misses`	Number of failed searches of the cache.
	`deletes`	Number of deletes from cache.
	`free-deferred`	Number of cache removals deferred until the next garbage collection run.
	`write-faults`	Number of cache writes that fail because the cache is full.
	`rewrites`	Number of expired cache entries.
`connections`	`added`	Number of new connections made.
	`reused`	Number of times existing connections were reused.

Notifications

AM sends the following notifications to Web Agent through WebSockets:

Configuration notifications

When the administrator makes a change to a hot-swappable agent configuration property, AM sends a notification to the agent to reread the agent profile from AM.

Configuration notifications apply when the agent profile is stored in AM’s configuration data store.

For more information about the cache, refer to Configuration cache.

Session Notifications

When a client logs out, or a CTS-based session expires, AM sends a notification to the agent to remove the client’s entry from the session cache.

For more information about the cache, refer to Session and policy decision cache.

Policy Notifications

When an administrator changes a policy, AM sends a notification to the agent to flush the session and policy decision cache, and the policy cache. Enable Notifications controls whether the AM server sends notifications to connected agents. It is enabled by default.

For more information about the cache, refer to Session and policy decision cache and Policy cache.

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

The AM advanced server configuration property, `org.forgerock.openam.notifications.agents.enabled`, controls whether the AM server sends notifications to connected agents. This property is enabled by default.

Disable notifications

CAUTION

Notifications are enabled by default. Before disabling notifications, consider the impact on security if the agent is not notified of changes in AM.

1. On the AM admin UI, select **REALMS** > Realm Name > **Applications** > **Agents** > **Web** > Agent Name.
2. On the **Global** tab, deselect the following options to disable notifications:
   - Enable Notifications

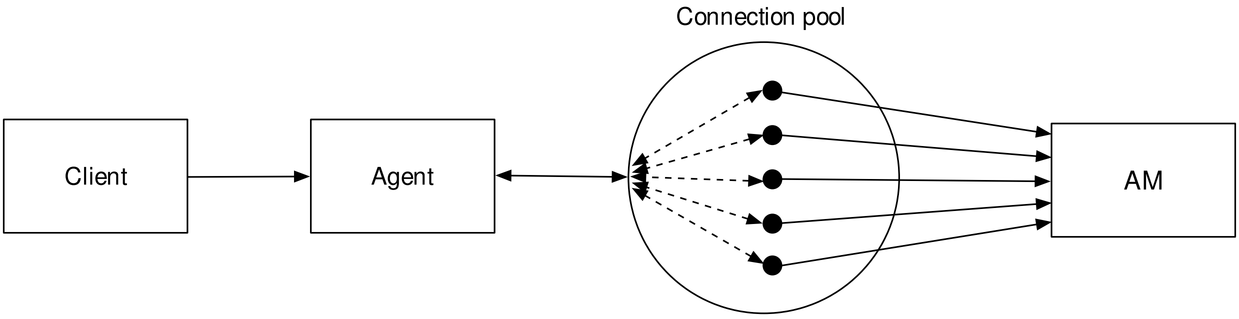
     After changing this property, restart the web server where the agent runs.
   - Enable Notifications of Agent Configuration Change

Connection pooling

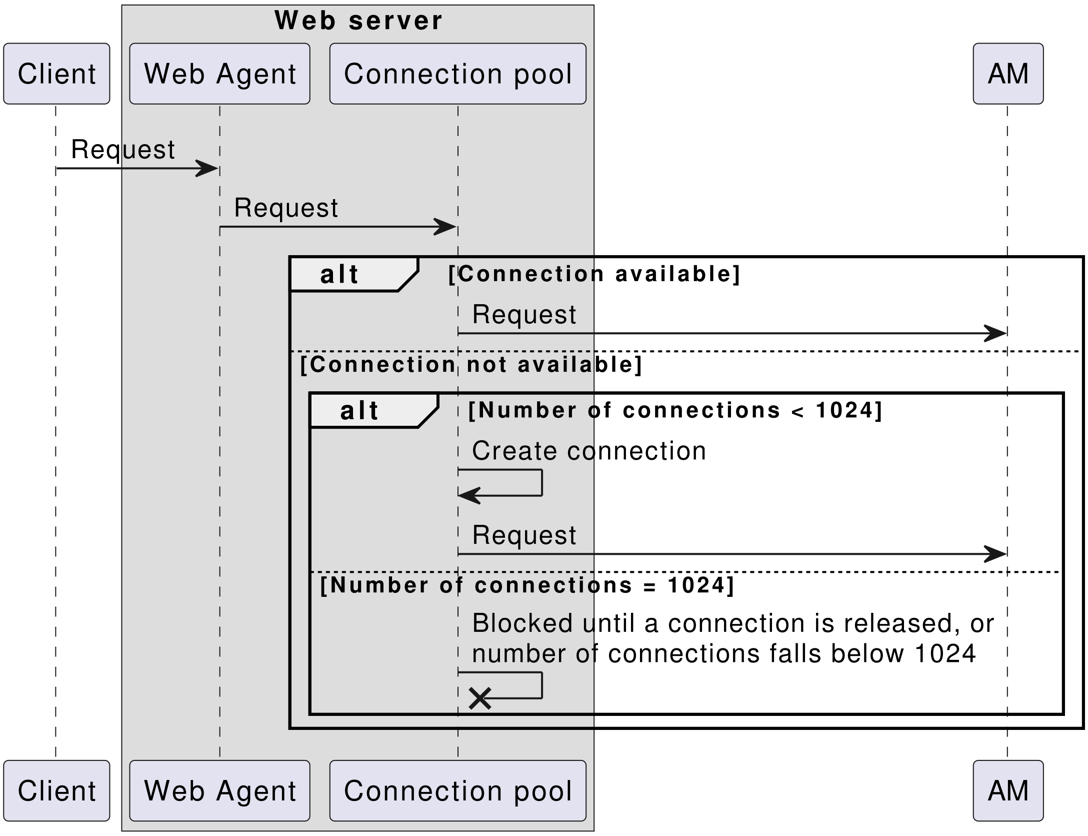
Use a connection pool between Web Agent and AM to cache and reuse connections, and so reduce the overhead of creating new connections. The agent can use an array of connections concurrently, with multiple request threads.

To enable connection pooling, set Enable Connection Pooling to `true`. 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:



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


Web server
Client
Web Agent
Connection pool
AM
Request
Request
alt
[Connection available]
Request
[Connection not available]
alt
[Number of connections < 1024]
Create connection
Request
[Number of connections = 1024]
Blocked until a connection is released, or
number of connections falls below 1024
Client
Web Agent
Connection pool
AM

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 fewer than 1024 connections are in use, the agent creates and uses a new connection. If 1024 connections are already in use, the request waits until an existing connection is released, or a new connection can be created.

When 1024 connections are in use, the agent creates additional temporary connections. Connections can be closed by AM/IDC, but the agent reopens them when it detects that they are closed.

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

Consider the following points for connection pooling:

- The connection pool can contain up to 1024 cached connections
- When more than 1024 connections are required, the agent creates temporary connection.
- By default, connections timeout after four seconds of waiting for a response. To change this value, configure Connection Timeout
- Tune Connection Timeout so that it is:
  - Long enough for systems to respond, and therefore prevent unnecessary failures
  - As short as possible to minimize the time to wait after a network failure
- To reduce the overhead of making new connections and SSL handshakes, set the HTTP keep-alive headers for AM containers or reverse proxies to longer than Connection Timeout.