Configure and Run the HQ Agent

Version: 37

Date: 9/9/10

Creator: Marie McGarry

Table of Contents

1. Understand Agent Environment and Operation

These topics that explain key facts about agent configuration, startup, and communication:

- Section 1.1, “Agent Directory Structure”
- Section 1.2, “Agent Launcher and Agent Startup”
- Section 1.3, “Agent - Server Communication”
- Section 1.4, “Agent Configuration”

1.1. Agent Directory Structure

The structure of the HQ Agent installation directory is shown below.

```
agent-4.x.y
    bin
    bundles
        agent-4.x.y-nnnn
    conf
    data
    log
    wrapper
    lib
    sbin
```

The root of the directory structure shown above as `agent-4.x.y` is typically referred to in HQ documentation as the *AgentHome* directory. Key agent directories include:

- `AgentHome/bin` - contains agent start scripts: `hq-agent.sh` and `hq-agent.bat`
- `AgentHome/conf` - contains `agent.properties`
- `AgentHome/data` - this is the directory where the agent saves the connection properties it uses to contact the HQ Server.
- `AgentHome/log` - contains:
  - `agent.log`
  - `agent.startup.log`
  - `wrapper.log`

1.2. Agent Launcher and Agent Startup

- Section 1.2.1, “Agent Launcher”
- Section 1.2.2, “What Happens When an Agent Starts Up”
- Section 1.2.3, “Automatic Restart Behavior”

This section describes the HQ Agent launcher and the agent startup process.

> **Don't Need the Background Explanation?**
>
> See:
>
> - Section 5, “Start, Stop, and Other Agent Operations”
> - Section 2, “Configure Agent - Server Communication”

1.2.1. Agent Launcher

The HQ Agent launcher is based on the Java Service Wrapper (JSW), a configurable tool that allows Java applications to be run as an NT service or Unix daemon process. It includes fault correction software to automatically restart crashed or frozen JVMs.

JSW runs as a native executable; it invokes and monitors the HQ Agent's JVM, based on configuration information provided to the wrapper at startup. In this way, the wrapper supports restarts of the JVM process without stopping the wrapper process itself. The JSW process acts as a watchdog for the JVM process, periodically pinging it for availability.

For more information about the features of the Java Services Wrapper, and how to configure its behavior, see http://wrapper.tanukisoftware.org/doc/english/download.jsp.

1.2.2. What Happens When an Agent Starts Up

An HQ Agent needs to know how to connect to the HQ Server, and the HQ Server needs to know how to connect to the HQ Agent — each component needs the IP address and listen port (and other connection properties) to use to establish a connection with the other. You must provide this information the first time an agent starts up after installation.

There are two ways to provide the agent-server connection properties:

- In `agent.properties` — Before starting the agent for the first time, configure the HQ Server connection properties in the `agent.properties` for the agent, as described in Section 2.2, “Configure Agent - Server Communication in Properties File”. When you start the agent, it will read the connection data from `agent.properties`. This configuration method is preferable if you have many agents to install, as it speeds the process and reduces the chance of error.
- Interactively — If you do not configure the HQ Server connection properties in the `agent.properties` for the agent, upon startup, the agent prompt for the properties in the command shell, as described in Section 2.1, “Configure Agent - Server Communication Interactively”.

When you start an HQ Agent:

1. The agent checks to see if there is a `/data` directory that contains the HQ Server's connection properties.

- At first startup, the `/data` directory will not exist — it is created after the first successful connection between agent and server is established.
- If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.

2. If the agent did not find the `/data` directory, it looks for an `agent.properties` file in a hidden directory named `.hq` in the home directory of the user that runs the agent.
   - This directory will not exist unless you have previously created it. This location for the properties file is supported to ensure configuration data is not lost in the event that the complete agent installation is overwritten in an upgrade.
   - If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.
3. If the agent did not find the agent-server connection properties in an `agent.properties` file in the hidden `/.hq` directory, it looks at the `agent.properties` file in its `/conf` directory.
   - If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.
4. If the agent did not find connection properties in the{{agent.properties}} file in its `/conf` directory, it prompts for the properties to be supplied interactively in the command shell.
5. Upon obtaining the agent-server connection properties, the agent attempts to connect to the HQ Server.
6. Once communications between agent and server have been succesfully established:
   - The agent saves the HQ Server connection settings in `AgentHome/data`.
   - The HQ Server saves the HQ Agent's connection settings in the HQ database.

1.2.3. Automatic Restart Behavior

As described above, JSW process periodically pings the agent JVM process.

If the JVM process exits with a non-zero exit code, indicating the JVM is hung or crashed, the JSW will restarts the JVM process. Messages in the Agent and JSW logs indicate that the JSW has restarted the Agent.I f the JSW cannot restart the JVM process, it will try again, up to a total of five times. You can configure different actions for the JSW, including the action it takes upon a particular JVM process exit code, and the number of times that it will attempt to restart the process.

1.3. Agent - Server Communication

This section is an overview of the communications between an HQ Agent and the HQ Server. The default communication between an agent and the server is bi-directional — some communication is initiated by the agent, some by the server:

1.3.1. Agent-to-Server Communication

Upon startup, an HQ Agent initiates a communications channel with the HQ Server. The agent continuously batches and sends monitoring results for the platform to the server. Agent-to-server data flows over HTTPS via a byte-encrypted XML protocol called Lather. The agent sends this data to the server:

- metric — the batch size is configurable.
- event — the batch size is configurable
- auto-discovery results — Auto-discovery results are reported after a each auto-discovery scan. Default scans run every 15 minutes, unless otherwise configured. Run-time scans run once a day, unless otherwise configured. Either type of scan can be run on-demand. For more information, see Understanding Resource Auto-Discovery and Options for Running and Controlling Resource Discovery.

1.3.2. Server-to-Agent Communication

The HQ Server initiates communication with an HQ Agent to relay the commands and data issued or configured by authorized users. The server-to-agent traffic is sent using a simple protocol directly on top of TCP, and includes:

- metric collection schedules
- resource control actions
- requests to initiate of auto-inventory scans

If your security policies dictate, you can configure the agent to initiate all communications with the HQ Server. You can configure unidirectional communications at first startup of a new 4.x agent, or an upgraded 4.x agent. If you want to change from bidirectional to unidirectional communications at a later time, see Section 2.3, “Configure Unidirectional Agent - Server Communication”.

1.4. Agent Configuration

The paragraphs that follow provide useful information about agent configuration data - how it is initially supplied, where it is saved, and how to change it.

- Section 1.4.1, “Agent Startup Configuration Data”
- Supported Locations for `agent.properties`
- Section 1.4.3, “How to Change Agent Setup Configuration Properties”

1.4.1. Agent Startup Configuration Data

The configuration choices you must supply for the agent to start up specify where and how it communicates with the HQ Server, and vice versa. As described in [What Happens When an Agent Starts Up. you can supply these setup values either interactively or in the agent's properties file.

Note that upon successful startup, for future reference, the agent saves the server connection data in `AgentHome/data` and the HQ Server stores the agent's connection data in the HQ database. The agent creates the `/data` directory upon first successful startup. On subsequent startups, the agent will look at the connection data stored in its `/data` directory to determine where and how to connect to the HQ Server.

The agent properties that govern communications are the ones that are absolutely required for an agent to start up. In addition, there are a number of other agent properties that you can use to configure optional agent features and behaviors. Unlike the setup properties, which the agent obtains from its data directory and the HQ database, the properties that control optional agent behaviors are persisted only in `agent.properties`.

For a complete list of agent properties, see Section 11, “Agent Properties”. The setup properties related to communications with the HQ server are those whose name starts with "agent.setup"; see Communication Properties Reference.

1.4.2. Supported Locations for agent.properties

`agent.properties` is installed in `AgentHome/conf`.

Note however, that agent first honors an external (from the agent installation) location for the properties file: an `.hq` directory under the home directory of the user under which the agent runs. If that directory does not exist, you can create it. (Under Windows, you can only create directory names with a leading period (.) in the DOS window (`mkdir`}.

Storing agent.properties external to the agent installation directory is useful, because some upgrade scenarios will overwrite the /conf directory. Specifically, upgrading an agent by installing a full agent package will overwrite your previous agent installation. This is relevant when you first upgrade an agent from 3.2.x or 3.1.x to 4.x, or if you choose to upgrade a 4.x agent to a later version by installing a full agent package. In these cases, if you don't keep the properties file in the .hq/ directory, back it up prior to upgrade, and restore it after upgrade.

Once you have an operational 4.0 agent installation, you can upgrade it by installing an agent bundle only - you can perform this upgrade operation from the HQ user interface or using a manual procedure, if you prefer. When you upgrade the agent bundle only, the agent's /conf directory is not updated---this method preserves the agent.properties file within the agent installation through the upgrade.

1.4.3. How to Change Agent Setup Configuration Properties

As described in Section 1.2, “Agent Launcher and Agent Startup”, you configure an agent's setup properties - the properties that begin with "agent.setup" - the first time you start it up. The properties that relate to where and how to contact the HQ server, are saved in the agent's /data directory; those related to how the server can reach the agent are stored in the HQ database.

If you need to make changes to those configuration values for an agent at a later time, you can delete the agent's / data directory, edit the agent.setup.* properties in the agent.properties file, and restart the agent. You *must* use this method if you wish to change the agent's listen port---you cannot change port settings without restarting the agent.

If you prefer to change the startup configuration (other than agent listen port) without having to restart the agent, run the following command in a command shell, while the agent is running.

```
AgentHome/hq-agent.sh setup
```

This will allow you to interactively supply new values for the setup properties, as described in Section 2.1, “Configure Agent - Server Communication Interactively”. This will this cause all properties in the properties file to be re-read, and take effect.

2. Configure Agent - Server Communication

These topics have instructions for configuring agent-server communications in `agent.properties` or interactively at first agent startup, and instructions for two Hyperic Enterprise options: unidirectional and proxied communications.

- Section 2.1, “Configure Agent - Server Communication Interactively”
- Section 2.2, “Configure Agent - Server Communication in Properties File”
- Section 2.3, “Configure Unidirectional Agent - Server Communication”
- Section 2.4, “Configure Proxy Server for Agent - Server Communication”

2.1. Configure Agent - Server Communication Interactively

These are instructions for supplying agent configuration settings the first time you start the agent.

1. Make sure that the HQ Server to which the Agent will connect is running.

2. Start the agent from the command line under your user account.

On Windows, install the HQ Agent service, and then start it with these commands:

```
hq-agent.bat install
hq-agent.bat start
```

On Unix-based platforms, assuming you have already installed, or unpacked the agent into its installation directory, enter:

```
AgentInstallationDirectory/bin/hq-agent.sh start
```

3. When the HQ Agent starts for the first time, it issues the prompts that follow so that it can establish communications with the HQ Server.

Prompt	**Notes**
Should Agent communications to HQ be unidirectional [default=no]	This prompt only appears if you are installing HQ Enterprise. To understand this option, see Section 2.3, “Configure Unidirectional Agent - Server Communication”
What is the HQ server IP address	Enter the listen address of your HQ Server. The server must be running. If the server is on the same machine as the agent, you can enter localhost. If there is a firewall blocking traffic from the agent to the server, specify the address of the firewall.
Should Agent communications to HQ always be secure [default=no].	If you want agent and server to communicate via SSL, enter yes. Otherwise, enter Return to accept the default - which is plain HTTP communications.
What is the HQ server port [default=7080]	This prompt appears if you accepted the default for the previous prompt. If your HQ Server is configured to listen on the default port of 7080, press Return. If there is a firewall blocking traffic from the agent to the server, configure it to forward traffic on TCP port 7080 (or 7443) to the host running the HQ Server.
What is the secure HQ server port [default=7443]	This prompt only appears if you answered yes to "Should Agent communications to HQ always be secure" prompt. If your HQ Server is configured to listen for SSL communications on the default port of 7443, press Return.
What is your HQ login [default=hqadmin]:	By default, the HQ server is initially configured with an administrative account with username hqadmin. Unless you have configured a different HQ user account for agent-server communications, accept the default.

Prompt	**Notes**
What is your HQ password	Enter the password for the username you supplied at the previous prompt. If you accepted the default admin user password when installing the HQ server, the password is hqadmin.
What IP should HQ use to contact the agent [default=n.n.n.n]	The prompt will show default that is the IP address the agent detected on the host. If there is another IP address on the host you prefer to use, enter it. If there is a firewall blocking traffic from the server to the agent, enter the IP address of the firewall, and configure the firewall to forward traffic intended for the HQ Agent to the listen address of the agent host.
What port should HQ use to contact the agent [default=2144]	Enter the agent port the HQ Server should use when it initiates contact with the agent. The value you supply to this prompt should be the port that the agent binds to at startup, which by default is 2144. **Note:** If your have previously edited agent.properties to explicitly define a different listen port, using the optional agent.listenPort property, that is the value you should supply to this prompt. If there is a firewall blocking traffic from the server to the agent, configure the device to forward traffic on TCP port 2144 to the HQ Agent.

Messages similar to the following are displayed upon successful startup of an agent.

```
Received temporary auth token from agent
Registering agent with HQ
HQ gave us the following agent token
1215038691323-8570363106994871928-8259195015465958356
Informing agent of new HQ server
Validating
Successfully setup agent
```

2.2. Configure Agent - Server Communication in Properties File

*Topics marked with*relate to HQ Enterprise-only features.*

*Feedback is welcome. Click **Add Comment** at the bottom of the page.*

2.2.1. File-Based Configuration of Agent-Server Communications is Efficient

It is simple to configure a single HQ Agent interactively: start it up, and respond to the prompts for locations, ports, and the other properties required for the HQ Agent and HQ Server to communicate.

However, of you have multiple HQ Agents to deploy, it is more efficient to set the values for the communication properties in properties files. For example, you can create a standard agent profile that you can copy to the agent installation, or to a location available to the agent installation.

In a standard `agent.profile` - one that you can deploy to multiple agents that report to the same HQ Server, you do not edit the properties that specify an agent's listen address and port. At first startup, if explicit values for IP address and port are not set, the HQ Agent - which detects the network interfaces on the platform - uses the first detected interface as its listen address, and port 2144 or 2443 as its listen port, depending on whether you configure the agent for plain text or SSL communications.

Whether you use a standard agent profile, or create a unique `agent.properties` file for each agent, be sure to copy the file into the `AgentHome/conf` directory, or to the hidden `HqUserHome/.hq` directory, if it exists. (The agent will honor the settings in an `agent.properties` in the HQ users home directory over those found in a properties file in `AgentHome/conf`.

2.2.2. Procedure: Configure Agent-Server Communication Properties

Step 1: Open or create agent.properties

Make a copy of the `agent.properties` file from the agent installation.

Step 2: Uncomment Agent-Server Communication Properties}

In the{{agent.properties}} file, find the section excerpted below, and remove the hash mark (#) in front of each the properties shown at the end of the excerpt.

```
## Use the following if you'd like to have the agent setup
## automatically from these properties. The values for these
## properties are used to answer the setup questions
##
## If any of these properties are left undefined, the setup
## process will prompt for their values
##
## If the value that should be used is the default when interactive
## setup is done, use the string *default* as the value for the option

#agent.setup.camIP=localhost
#agent.setup.camPort=7080
#agent.setup.camSSLPort=7443
#agent.setup.camSecure=yes
#agent.setup.camLogin=hqadmin
#agent.setup.camPword=hqadmin
#agent.setup.agentIP=*default*
#agent.setup.agentPort=*default*
#agent.setup.resetupTokens=no
```

Note: For more information about the properties above, see Communication Properties Reference below.

Step 3: Define Communication Properties in agent.properties File

The `agent.properties` file contains properties you can configure to govern both agent-initiated and server-initiated communication.

- Specify the location and credentials the agent should use to contact the HQ Server with these properties:
  - agent.setup.camIP — Specify the address or hostname of the HQ Server.
  - agent.setup.camPort — The default value is the standard plaintext HQ listen port. You can specify a different port — as long as it is not already in use.
  - agent.setup.camSSLPort — The default value is the standard SSL HQ listen port. You can specify a different port — as long as it is not already in use.
  - agent.setup.camSecure — The default value is "yes" (use SSL). Change to "no" if you do not require the agent to use secure communications when contacting the HQ Server.
  - agent.setup.camLogin — Specify the username the agent should use when connecting to the server. If you change the value from the default value ("hqadmin"), make sure that that user account is properly configured on the HQ Server.
  - agent.setup.camPword — Specify the password the agent should use, along with the username above, when connecting to the server. Make sure that the password is the one configured in HQ for the user account.
- Specify the address or hostname and the listen port the HQ Server should use to contact the HQ Agent with these properties:
  - agent.setup.agentIP — If you leave the default setting — "*default*" — the HQ Agent will detect an IP address on the platform and choose it as its listen address.

- agent.setup.agentPort — If you leave the default setting — {"*default*" — the HQ Agent will use the default listen port (either 7080 or 7443) as its listen address. If that port is unavailable, the agent will detect a free port and choose it as its listen port.

These are the minimum properties required for agent-server communication.

Unless you want to configure other agent behaviors, save your changes and proceed to Step 6 - Copy `agent.properties` to Agent Installation

Step 4 - Configure Unidirectional Communications (Optional)

By default, agent-server communication is bi-directional. If your policies dictate that all communication between agent and server are agent-initiated, you can uncomment the `agent.setup.unidirectional` property and set it to "yes".

Unless you want to configure other agent behaviors, save your changes and proceed to Step 6 - Copy `agent.properties` to Agent Installation

Step 5 - Configure Additional Agent Behaviors (Optional)

As desired, you can configure additional agent behaviors in the `agent.properties` file. Refer to Configuring HQ Agent Behaviors.

After completing your edits,save your changes and copy the updated properties file to desired agent or agents, as described in the next section.

Step 6 - Copy agent.properties to Agent Installation

2.2.3. Communication Properties Reference

agent.setup.camIP

Description

You can use this property to define for the agent the IP address of the HQ server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

The value can be provided as an IP address or a fully qualified domain name. To identify an server on the same host as the server, set the value to 127.0.0.1.

If there is a firewall between the agent and server, specify the address of the firewall, and configure the firewall to forward traffic on port 7080, or 7443 if you use the SSL port, to the HQ Server.

Default

Commented out, localhost.

agent.setup.camPort

Description

You can use this property to define for the agent, at first startup after installation, what server port to use for non-secure communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, 7080.

agent.setup.camSSLPort

Description

You can use this property to define for the agent, at first startup after installation, what server port to use for SSL communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, 7443

agent.setup.camSecure

Description

You can use this property to define for the agent, at first startup after installation, whether to communicate with the server over SSL. If you set this property to yes, all agent-server communications will be use the SSL secure port.

If acceptable in your environment, non-SSL communication offers improved performance for agent-server communications.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, value of yes

agent.setup.camLogin

Description

You can use this property to define for the agent, at first startup after installation, the HQ username to use when registering itself with the server. The permission required on the server for this initialization is Create, for Platforms.

A login from the agent to the server is only required during the initial configuration of the agent.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, hqadmin

agent.setup.camPword

Description

You can use this property to define for the agent, at first startup after installation, the password for the user specified by agent.setup.camLogin.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, hqadmin**.**

agent.setup.agentIP

Description

This specifies the IP address that the HQ Server will use to contact the HQ Agent. If the agent is on the same host as the server, value of 127.0.0.1 is valid.

If there is a firewall between the server and agent, specify the IP address of the firewall, and configure the firewall to forward traffic intended for the agent to the agent's listen address, which can be configured with agent.listenIP.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to *default*. If you use the agent.setup.* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value *default*, the HQ Server will contact the agent using the IP address that SIGAR detects on the agent host.

agent.setup.agentPort

Description

This specifies the port (on the IP address configured with agent.setup.agentIP) on the agent on which the HQ Server will communication with the agent.

If there is a firewall between the agent and the server, set agent.setup.agentPort to the appropriate port on the firewall, and configure the firewall to forward traffic intended for the agent to the agent listen port, which can be configured with.

The agent reads this value only in the event that it cannot find its connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to *default*. If you use the agent.setup.* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value *default*, the HQ Server will contact the agent on port 2144, unless SIGAR detects it is not available, in which case another default is selected.

agent.setup.resetupToken

Description

You can use this property to define for the agent, at first startup after installation, whether the agent will create a new token to use to authenticate with the server each time it starts up. Regenerating a token is useful if the Agent cannot connect to the server because the token has been deleted or corrupted.

Regardless of the value of this property, an agent will generate a token the first time it is started after installation.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to no.

agent.setup.unidirectional

Available only in HQ Enterprise

Description

Enables the unidirectional communications between HQ Agent and HQ Server, in HQ Enterprise. For more information, see Section 2.3, “Configure Unidirectional Agent - Server Communication”.

Default

Commented out, defaults to no.

2.3. Configure Unidirectional Agent - Server Communication

Available only in HQ Enterprise
If your security policies dictate, you can configure the agent to initiate all communications with the HQ Server. You can configure unidirectional communications at first startup. Unidirectional communications are always via SSL.

This section has instructions for changing agent communications from bidirectional to unidirectional and vice versa after the agent has already been configured.

- Section 2.3.1, "Changing from Bidirectional to Unidirectional Communications"
- Section 2.3.2, "Changing from Unidirectional to Bidirectional Communications"

2.3.1. Changing from Bidirectional to Unidirectional Communications

Note: These instructions apply to HQ Enterprise 4.1 and later.

1. Stop the agent.
2. Remove the agent's `\data` directory.
   - Removing the `\data` directory will cause the agent, at next startup, to look for the startup settings it needs to connect to the HQ Server in its agent.properties file; if the properties file doesn't contain them, it will prompt for settings in the shell.
3. Configure the agent for unidirectional communications using one of these methods:
   - If your practice is to provide all agent startup properties in the properties file, edit `agent.properties` to set `agent.setup.unidirectional=yes`, and start the agent.
   - If your practice is to configure the agent startup properties interactively, start the agent, and respond "yes" when asked if the agent should be configured for unidirectional communications.
4. In the HQ user interface, navigate to the platform's Inventory tab and click **Edit** in the "Type & Network Properties" section.
   - In the edit view for "Type & Network Properties", the "Agent Connection" drop-down list will show your currently selected port for bidirectional communications, something like 10.2.0.213:2144, where 10.2.0.213 is the IP address of the platform, and 2144 is the bidirectional port number previously used.
5. Expand the drop-down list and select the entry that shows the same IP address, and "-1" as the port:
6. 10.2.0.213:-1
   - Your agent will now use unidirectional communications.

2.3.2. Changing from Unidirectional to Bidirectional Communications

1. Stop the agent.

2. Remove the agent's `data` directory.

    - Removing the `data` directory will cause the agent, at next startup, to look for the startup settings it needs to connect to the HQ Server in its `agent.properties` file; if the properties file doesn't contain them, it will prompt for settings in the shell.

3. Configure the agent for bidirectional communications using one of these methods:

    - If your practice is to provide all agent startup properties in the properties file, edit `agent.properties` to set `agent.setup.unidirectional=no`, and start the agent.

    - If your practice to configure the agent startup properties interactively, start the agent, and when prompted for communications direction, respond "no" when asked if the agent should be configured to run in unidirectional mode.

4. In the HQ user interface, navigate to the platform's Inventory tab and click Edit in the "Type & Network Properties" selection.

5. Select the appropriate agent in the "Agent Connection" drop down.

    - In the edit view for "Type & Network Properties", the "Agent Connection" drop-down list will show your currently selected port for unidirectional communications, something like 10.2.0.213:-1, where 10.2.0.213 is the IP address of the platform, and -1 is the port number.

6. Expand the drop-down list and select the entry that shows the same IP address, and "2144" as the port (or the port you are configured to use, if not the default), for example, 10.2.0.213:2144

    - Your agent will now use bidirectional communications.

2.4. Configure Proxy Server for Agent - Server Communication

This page lists the agent properties for configuring the HQ Agent to connect to the HQ Server via a proxy server.

agent.proxyHost

Available only in HQ Enterprise

Description

The host name or IP address of the proxy server that the agent must connect to first when establishing a connection to the HQ server.

Default

none

agent.proxyPort

Available only in HQ Enterprise

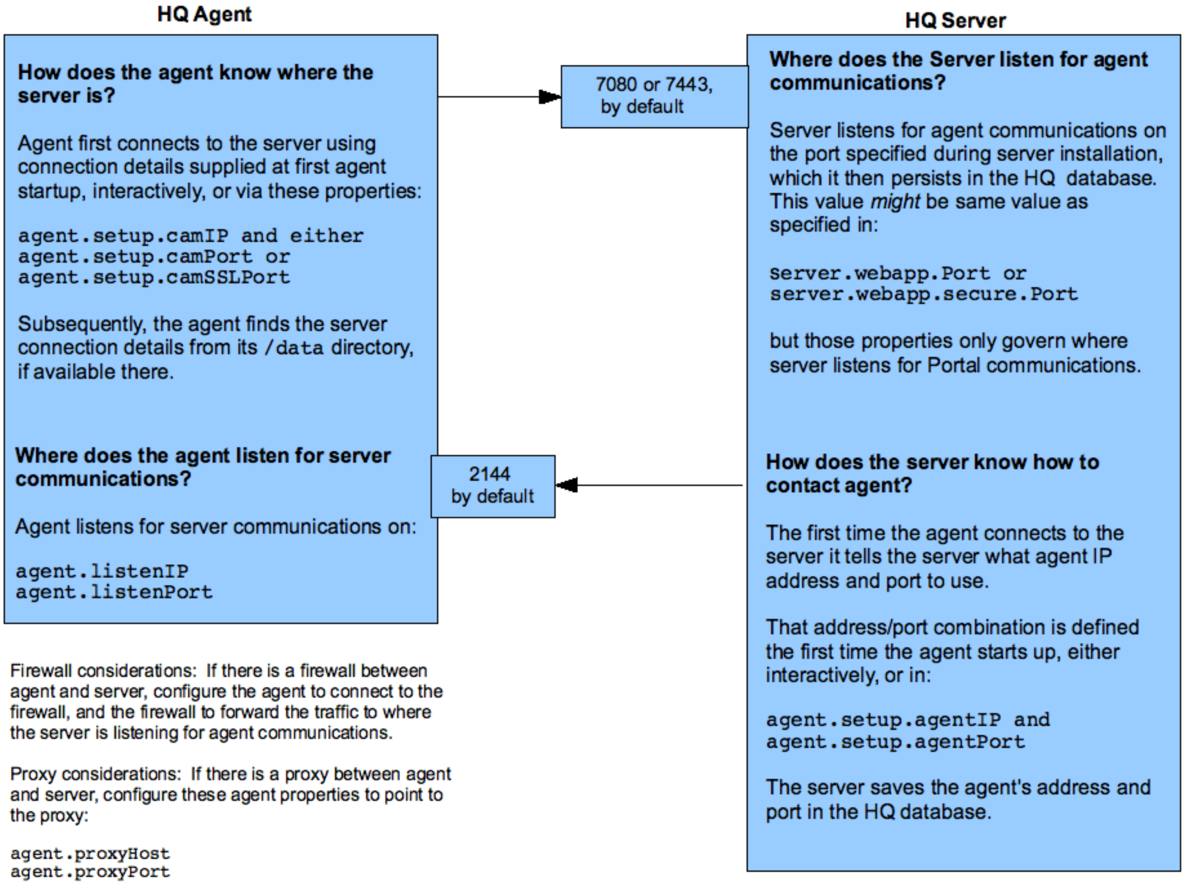
Description

The port number of the proxy server that the agent must connect to first when establishing a connection to the HQ server.

Default

none

3. Agent Server Communications Diagram



How does the agent know where the server is?

Agent first connects to the server using connection details supplied at first agent startup, interactively, or via these properties:

```
agent.setup.camIP and either
agent.setup.camPort or
agent.setup.camSSLPort
```

Subsequently, the agent finds the server connection details from its `/data` directory, if available there.

Where does the agent listen for server communications?

Agent listens for server communications on:

```
agent.listenIP
agent.listenPort
```

Firewall considerations: If there is a firewall between agent and server, configure the agent to connect to the firewall, and the firewall to forward the traffic to where the server is listening for agent communications.

Proxy considerations: If there is a proxy between agent and server, configure these agent properties to point to the proxy:

```
agent.proxyHost
agent.proxyPort
```

Where does the Server listen for agent communications?

Server listens for agent communications on the port specified during server installation, which it then persists in the HQ database. This value *might* be same value as specified in:

```
server.webapp.Port or
server.webapp.secure.Port
```

but those properties only govern where server listens for Portal communications.

How does the server know how to contact agent?

The first time the agent connects to the server it tells the server what agent IP address and port to use.

That address/port combination is defined the first time the agent starts up, either interactively, or in:

```
agent.setup.agentIP and
agent.setup.agentPort
```

The server saves the agent's address and port in the HQ database.

4. Deploying Multiple HQ Agents

This section has recommendations for how to install multiple agents in a large HQ environment.

4.1. Establish Installation Environment

4.1.1. Set up an Install Server

Choose a machine that can access all target platforms from which to perform remote installation. We refer to this as the "install server". On the install server, create a user account, for instance "hyperic", with permissions required to SSH into each target platform without a password.

4.1.2. Create Accounts and Installation Directories on Target Platforms

On each platform to which you'll install an agent:

- Create an account that is identical to the one you created on the install server.
- Create identical installation directories, for example, `/home/hyperic`.
- On Windows systems, create an `.hq` direcotyr in the user home directory that contains the installation directory you just created. You must open a command shell, and use the mkdir command to create the `.hq` directory. (This directory is created automatically when you install the HQ Agent on Unix-like systems.

4.2. Create Standard Agent Properties File

To enable mass agent deployment, you create an `agent.properties` that defines the agent properties required for the agent to start up and connect with the HQ Server. If you supply the necessary information in the properties file, each HQ Agent will find its setup configuration at startup, rather querying for it interactively.

At a minimum, you must define the HQ Server address and port. In addition, you can configure optional agent behaviors that are controlled by agent properties. For more information, see Section 2.2, “Configure Agent - Server Communication in Properties File”.

The first time you start the agent, it will obtain the server connection information it needs to contact the server and register itself.

4.3. Perform Remote Agent Installations

4.3.1. Install and Start Agents One-by-One

Follow these steps to install agents one-by-one. To install to all target platforms at once, see Deploy and Start Multiple Agents at Once below.

1. Log on to your installation account on the install server.
2. SSH to the remote platform.
3. Copy the agent archive to the agent host.
4. Unpack the agent archive.
5. Copy the `agent.properties file` to the `/.hq` directory under the home directory of the standard agent installation user account.
6. If you are upgrading an 3.x agent to 4.x, stop the 3.x agent, if necessary.
7. Start the new agent.

4.3.2. Deploy and Start Multiple Agents at Once

Deploy an agent to each host listed in `hosts.txt` by

1. Create a `hosts.txt` file on your install server that maps hostname to IP address for each platform to which you wish to install the agent.
2. Open a command shell on the install server.
3. Entering the following commandd in the shell, supplying the correct name of the agent package in the export command:

```
$ export AGENT=hyperic-hq-agent-4.0.0-x86-linux.tgz
$ for host in `cat hosts.txt`; do scp $AGENT $host: && ssh $host "tar zxf $AGENT && ./
hyperic-hq-agent-4.0.0/hq-agent.sh start"; done
```

If target hosts have sequential names (for example, host001, host002, host003, etc), you can skip the hosts.txt file and use the seq command like this:

```
$ export AGENT=hyperic-hq-agent-4.0.0-x86-linux.tgz
$ for i in `seq 1 9`; do scp $AGENT host$i: && ssh host$i "tar
zxf $AGENT && ./hyperic-hq-agent-4.0.0/hq-agent.sh start"; done
```

4.4. Verify Successful Agent Startup

After successfully registering itself with the HQ Server, an HQ Agent runs an autoscan, and should discover its host platform, and supported servers managed products. Check the Auto-Discovery portlet in the HQ Dashboard to verify the the platforms were discovered.

If you have problems, see Section 9, “Troubleshoot Agent Problems”.

5. Start, Stop, and Other Agent Operations

*Topics marked with*relate to HQ Enterprise-only features.*

*Feedback is welcome. Click **Add Comment** at the bottom of the page.*

5.1. Run the Agent Launcher from the Command Line

You initiate the agent launcher and agent lifecycle commands with the `hq-agent.sh`, or hq-`agent.bat` script, in the `AgentHome/bin` directory.

5.1.1. Running hq-agent.sh

1. Open a command shell or terminal window.
2. Enter a command of this form:

```
sh hq-agent.sh command
```

Where command is one of the following:

- start — Starts the agent as a daemon process.
- stop — Stops the agent's JVM process.
- restart — Stops and then starts the agent's JVM process
- status — Queries the status of the agent's JVM process.
- dump — Runs a thread dump for the agent process, and writes the results to the `agent.log` file in `AgentHome/log`.
- ping — Pings the agent process.
- setup — Causes the HQ Agent to prompts you for the agent-server connection properties, allowing you to change the values that were provided at first agent startup.

5.1.2. Running hq-agent.bat

1. Open a terminal window.
2. Enter a command of this form:

```
hq-agent.bat command
```

Where command is one of the following:

- start - starts the agent as an NT service
- stop - stops the agent as an NT service
- restart - stops and then starts the agent's JVM process
- install - installs the agent NT service
- remove - removes the agent's service from the NT service table
- query - queries the current status of the agent NT service (status
- ping - pings the agent process for availability

- setup - prompts for setup configuration for the agent process

5.2. Run the Agent Launcher from the HQ User Interface

In HQ Enterprise, you can issue selected commands to an running HQ Agent.

Available only in HQ Enterprise
Agent control commands are available on the **Views** tab for an HQ Agent or a group of agents in inventory.

- Restart an Agent from the HQ User Interface [29]
- Ping an Agent from the HQ User Interface [29]
- Upgrade an Agent from the HQ User Interface [29]
- Push a Resource Plugin to the Agent from the HQ User Interface [30]

5.2.1. Restart an Agent from the HQ User Interface

The **restart** command invokes the agent's Java Service Wrapper's restart command. The restart command shuts down the JVM process in which the agent runs, waits for the process to terminate cleanly, and spawns a new JVM process for the agent. During the restart process, the agent's metric collection and resource control functionality will be interrupted.

The restart command happens asynchronously. To verify that the restart succeeded you can go to the page for the agent in the HQ Portal and check its availability. Alternatively, you could configure an alert for the agent that fires when the agent's availability changes.

To restart an agent:

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select **restart** from the pull-down list
3. Click **Execute**.

5.2.2. Ping an Agent from the HQ User Interface

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select **ping** from the pull-down.
3. Click **Execute**.

5.2.3. Upgrade an Agent from the HQ User

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select the **upgrade** command from the pull-down.
3. On the pull-down list of agent bundles that appears, select the desired bundle
4. Click **Execute**.
   - The selected agent bundle is transferred from the HQ Server to the target agent(s).

- The agent expands the bundle locally.
- The agent updates the local bundle property.
- The server restarts the agent.

The configuration settings in the agent's `/conf/agent.properties` file are preserved.

5.2.4. Push a Resource Plugin to the Agent from the HQ User Interface

This **push plugin** command sends new and changed resource plugins to the target agent(s). Pushing plugins to an agent results in an agent restart.

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select the **push plugin** command from the pull-down.
3. On the pull-down list of plugins that appears, select the desired plugin.
4. Click **Execute**.
    - The selected plugin is transferred from the HQ Server to the target agent(s)
    - The server restarts the agent(s).

5.3. Run the Agent Without the Java Service Wrapper

If you run an HQ Agent on a system that does not support the Java Service Wrapper, or for other reasons prefer not to use the wrapper, you can start the agent without the wrapper.

The `hq-agent-nowrapper.sh` agent start script in `AgentHome/bundles/agent-x.y.z/bin`

Because `hq-agent-nowrapper.sh` does not fork itself into the background, run it using `nohup`:

```
nohup AgentHome/bundles/agent-x.y.z/bin/hq-agent-nowrapper.sh &
```

6. Configure Auto-Discovery Scanning and Reporting

This pages lists agent properties that control auto-inventory scanning and reporting.

autoinventory.defaultScan.interval.millis

Description

Specifies how frequently the agent performs a default autoinventory scan.

The default scan detects servers and platform services, typically using the process table or the Windows registry. Default scans are less resource-intensive than runtime scans.

Default

Commented out, set to 86,400,000 milliseconds, or 1 day

autoinventory.runtimeScan.interval.millis

Description

Specifies how frequently the agent performs a runtime scan.

A runtime scans may use more resource-intensive methods to detect services than a default scan. For instance, a runtime scan may involve issuing an SQL query or looking up an MBean.

Default

86,400,000 milliseconds, or 1 day

agent.eventReportBatchSize

Description

The maximum number of events that an HQ Agent will send per contact with the HQ Server.

When the agent detects an event, it puts it in a queue of events to report to the server. As long as there are events in the queue, the agent continuously creates and sends batches of events to the HQ Server. The `agent.eventReportBatchSize` sets the (maximum) number of events the agent will put in a batch.

Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 100 events per contact with the server.

agent.maxBatchSize

Description

The maximum number of metrics that the HQ Agent will send per contact with the HQ Server.

An HQ Agent puts the metrics it collects into a queue of metrics to report to the server. As long as there are metrics in the queue, the agent continuously creates and sends batches of metrics to the HQ Server. The `agent.maxBatchSize` property sets the (maximum) number of metrics the agent will put in a batch.

Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 500 metrics per contact with the server.

7. Configure Plugin Loading

By default, at startup, an HQ Agent loads all of the plugins in its plugin directory - `AgentHome/bundles/agent-x.y.z-nnnn/pdk/plugins`.

You can reduce the agent's memory footprint by configuring it to load only the plugins you use. You can either specify a list of plugins to exclude, or configure a list of the plugins to load.

plugins.exclude

Description

Use this property to specify plugins that you do not wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

Usage

Supply a comma-separated list of plugins to exclude, for example:

```
{{plugins.exclude=jboss,apache,mysql}}
```

plugins.include

Description

Use this property to specify plugins that you do wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

Usage

Supply a comma-separated list of plugins to include, for example:

```
plugins.include=weblogic,apache
```

8. Configure Agent Logging

*Topics marked with***relate to HQ Enterprise-only features.*

*Feedback is welcome. Click **Add Comment** at the bottom of the page.*

8.1. Agent Log Files

Agent log files are stored in the `AgentHome/log` directory. They include:

- `agent.log`
- `agent.startup.log`
- `wrapper.log` - The agent JSW-based launcher writes messages to this log.

8.2. Configure Agent Log Name or Location

Use these properties to change the name or location of the agent log file:

agent.logDir

Description

You can add this property to the `agent.properties` file to specify the directory where the HQ Agent will write its log file. Unless you specify a fully qualified path, `agent.logDir` is evaluated relative to the agent installation directory.

This property does not exist in `agent.properties` unless you explicitly add it. To change the location for the agent log file, add `agent.logDir` to `agent.properties` and enter a path relative to the agent installation directory, or if desired, a fully qualified path.

Note that the name of the agent log file is configured with the agent.logFile property.

Default

`agent.logDir` does not exist in `agent.properties` unless you explicitly add it. The default behavior is equivalent to this setting:

```
agent.logDir=log
```

resulting in the agent log file being written to the `AgentHome/log` directory.

agent.logFile

Description

Specifies the path and name of the agent log file.

Default

Note that in `agent.properties`, the default setting for `agent.LogFile` is made up of a variable and a string.

```
agent.logFile=${agent.logDir}\agent.log
```

- `agent.logDir` is a variable - it supplies the value of an identically named agent property. By default, the value of `agent.logDir` is `log`, interpreted relative to the agent installation directory.
- `agent.log` is the name for the agent log file.

So, by default, the agent log file is named `agent.log`, and is written to the `AgentHome/log` directory.

If you want the agent to the log to a different directory, you must explicitly add the agent.logDir property to `agent.properties`.

8.3. Configure Agent Logging Level

Use this property to control what severity of messages that agent writes to the agent log file:

agent.logLevel

Description
Specifies the level of detail of the messages the Agent writes to the log file. Allowable values are: INFO and DEBUG.
Default
INFO

8.4. Redirect System Messages to the Agent Log

Use these properties to redirect system-generated messages to the agent log file:

agent.logLevel.SystemErr

Description

Redirects System.err to agent.log. Commenting out this setting will cause System.err to be directed to agent.log.startup.

Default

ERROR

agent.logLevel.SystemOut

Description

Redirects System.out to agent.log. Commenting out this setting will cause System.out to be directed to agent.log.startup.

Default

```
INFO
```

After editing the `agent.properties` file, save your changes and restart the HQ Agent.

8.5. log4j Properties

The log4j properties in `agent.properties` are excerpted below.

Agent log4j Properties

```
log4j.rootLogger=${agent.logLevel}, R
log4j.appender.R.File=${agent.logFile}
log4j.appender.R.MaxBackupIndex=1
log4j.appender.R.MaxFileSize=5000KB
log4j.appender.R.layout.ConversionPattern=%d %-5p [%t] [%c{1}] %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R=org.apache.log4j.RollingFileAppender
##
## Disable overly verbose logging
##
log4j.logger.httpclient.wire=ERROR
log4j.logger.org.apache.commons.httpclient=WARN
log4j.logger.org.hyperic.hq.measurement.agent.server.SenderThread=INFO
log4j.logger.org.hyperic.hq.agent.server.AgentDListProvider=INFO
log4j.logger.org.hyperic.hq.agent.server.MeasurementSchedule=INFO
log4j.logger.org.hyperic.hq.measurement.agent.server.ScheduleThreadTrace=INFO
log4j.logger.org.hyperic.util.schedule.ScheduleTrace=INFO
##
## Only log errors from naming context
##
log4j.category.org.jnp.interfaces.NamingContext=ERROR
log4j.category.org.apache.axis=ERROR
```

9. Troubleshoot Agent Problems

9.1. Agent Log Files

If you have problems with an agent, look for messages in `agent.log`, `agent.log.startup`, and `wrapper.log`.

To can increase the level of agent logging by setting `agent.logLevel=DEBUG`. Note however that debug logging is very verbose and uses more system resources. Hyperic recommends running the agent in DEBUG mode only when troubleshooting a problem.

For more information, see Section 8, “Configure Agent Logging”.

9.2. Troubleshooting Unresponsive Agents

Under high load, the agent may become unresponsive. If this occurs and there are no coincident errors or warnings in the agent log that indicate another explanation, it may be that the agent JVM was starved for memory, and unresponsive to a ping from the Java Service Wrapper (JSW).
In that case the wrapper.log file will contain an entry like this:

```
ERROR | wrapper | 2009/01/15 02:15:18 | JVM appears hung: Timed out waiting
for signal from JVM.
```

To resolve the problem, you can configure the JSW to give the agent more time to respond to startup and ping requests.

Increase the JSW's timeout period from 30 seconds to 300. To do so, add this property to `AGENT_HOME/bundles/agent-4.0.xxxx/conf/wrapper.conf`.

```
wrapper.ping.timeout=300
```

This will cause the JSW to wait longer for a ping response before determining that the JVM is hung.

Increase the agent's startup timeout from 30 seconds to 300. This will give the agent more time to start up before wrapper gives up on it and kills the process. To do so, add this property value:

```
wrapper.startup.timeout=300
```

to

```
AGENT_HOME_/bundles/agent-4.0.xxxx/conf/wrapper.conf
```

9.3. Configure Agent Startup Script Timeout

You can control how long an the startup script for an agent will run before timing out if the startup sequence is not successful — for example, if the HQ Agent cannot connect to the HQ Server — using the `agent.startupTimeOut` property, defined below:

agent.startupTimeOut

Description

The number of seconds that the agent startup script will wait before determining that the agent did not startup successfully. If the agent is not determined to be listening for requests within this period of time, an error is logged, and the startup script times out.

Default

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to timeout after 300 seconds.

After editing the `agent.properties` file, save your changes and restart the HQ Agent.

9.4. Configure Agent Connection Timeout

Lather, the connection protocol for agent-to-server communication, is configured such that agent connections time out after five minutes. If the agent log indicates lots of timeouts, you can increase the timeout period from 300000 to 900000 in this file:

```
hq-engine/server/default/deploy/lather-jboss.sar/jboss-lather.war/WEB-INF/web.xml
```

in this stanza:

```
<init-param>
     <param-name>org.hyperic.lather.execTimeout</param-name>
     <param-value>900000</param-value>
</init-param>
```

9.5. Agent Having a Hard Time Keeping up with Metric Volume

If an HQ Agent's queue of metrics grows to a certain level over a period of time, the following warning message is written to the `agent.log` file:

```
The Agent is having a hard time keeping up with the frequency of metrics
taken. Consider increasing your collection interval.
```

You can alleviate the problem by increasing the collection intervals for the metrics collected for a resource type. For most metrics, the default is every 5 or 10 minutes. You can change the collection interval for all metrics collected for a resource type on the **Administration** > **Monitoring Defaults** page for the resource type.

Want to Change a Whole Bunch of Metric Collection Intervals?

If you want to change metric templates in bulk, or without using the HQ user interface, you can change metric collection settings, including collection intervals, for for a resource type with the HQApi **metricTemplate** command. You can use the **metricTemplate sync** option from the command line or in a script, as desired. For more information, see HQApi metrictemplate command.

Note also that, if an agent that logs metric volume warnings is monitoring a large number of remote services over the network (for example, HTTP, FTP, SNMP, or another server type whose protocol the agent supports), you can spread the load around — configure a different agent to monitor some of the network services.

10. Tweak the Agent to Enable a Resource Plugin

These topics describe how to configure the agent to enable a particular plugin to perform one or more of its management functions:

10.1. Configure Agent Account Privileges under Solaris 10

To auto-discover certain products under Solaris 10, the HQ Agent must run as root or you need explicitly grant additional permissions to the account where the agent runs. For background information, see Solving Auto-Discovery Problems.

Under Solaris 10's Least Privilege Model (LPM), default privileges are minimal. The HQ Agent must be able to read /proc/$pid/ files on the platform. Problems with auto-discovery on Solaris 10 may be the result of insufficient privileges. Depending on your account privilege implementation you may need to grant the proc_zone privilege to the agent account.

For example, you could add this line to `/etc/user_attr`, to grant `proc_owner` privilege the hq user and deny the `proc_session` privilege:

```
hq::::type=normal;defaultpriv=basic,proc_owner,!proc_session
```

Note: After changing account privileges, the user needs to re-login.

Your approach for enabling agent access to /proc/$pid/ files will depend on your company's LPM implementation and best practices.

For related information, see The Least Privilege Model in the Solaris 10 OS and process privilege model.

10.2. Configure Agent HTTP Request Header

If you monitor a remote HTTP server, it is useful to configure the HTTP request header for agent HTTP requests.

http.useragent

Describes HQ 4.3 feature

Description

The `http.useragent` property defines the value for the User-Agent request header in HTTP requests issued by the HQ Agent. By default, the User-Agent in agent requests includes the HQ Agent version - and so change upon agent upgrade. If a target HTTP server is configured to block requests with an unknown User-Agent, agent requests will fail after agent upgrade.

You can use `http.useragent` to define a User-Agent value that will be consistent across upgrades.

Note: `agent.properties` does not contain this property by default. You must explicitly add it to the file.

Default

```
Hyperic-HQ-Agent/Version
```

For example:

```
Hyperic-HQ-Agent/4.1.2-EE
```

10.3. Configure Agent to Monitor JBoss

jboss.installpath

Description

To enable the agent to monitor JBoss, specify the location of the JBoss root directory.

Default

/usr/local/jboss-4.0.0

10.4. Configure Agent to Monitor WebSphere Application Server

This page has agent properties related to monitoring WebSphere.

websphere.installpath

Description

To enable the agent to monitor WebSphere, specify the location of the WebSphere jars.

Default

/opt/WebSphere/AppServer

websphere.useext

Description

This property is required to enable management of WebSphere 6.0 and 6.1.

Do **not** define the `websphere.useext` property to monitor WebSphere 7.

Usage

Add the following property definition to the `agent.properties file` for an an HQ Agent that will manage WebSphere 6.0 or 6.1.

```
websphere.useext=true
```

10.5. Configure HQ Agent to Manage WebLogic Server

This page lists agent properties related to managing WebLogic Server.

10.5.1. Specify WebLogic Server Installation Path

weblogic.installpath

Description

To enable the agent to monitor WebLogic 8.1, specify the location server/lib/weblogic.jar

Default

/usr/local/bea/weblogic-8.1

10.5.2. Configure HQ Agent for Two-Way SSL with WebLogic Server

In Two-Way SSL, each side presents an SSL certificate to the other - the client must trust the server cert and the server must trust the client cert. The process is:

1. HQ Agent initiates a connection with the Administration Server.
2. The Administration server presents its certificate and asks the HQ Agent for its certificate.
3. The HQ Agent presents its certificate (*client2certs.pem*), using the private key (*clientkey.pem* and *clientkey*) to encrypt the communication.

To enable the HQ Agent to use Two-Way-SSL with a Weblogic Administration Server, you specify the client cert the HQ Agent presents to the Administration Server by adding these properties to `agent.properties`.

- weblogic.auth.method=ssl2ways
- weblogic.ssl2ways.key=*ClientKey.pem* (client private key)
- weblogic.ssl2ways.key.pass=*ClientKey* (passphrase for client private key)
- weblogic.ssl2ways.cert=*Client2Certs.pem* (client certificate)

Weblogic docs: Using Two-Way SSL Authentication

weblogic.auth.method

View Source
Describes HQ 4.3 feature

Description

`weblogic.auth.method` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add the following line to the `agent.properties` file to specify that the agent will use Two-Way-SSL for communications with the Administration Server.

```
weblogic.auth.method=ssl2ways
```

Default

None.

weblogic.ssl2ways.cert

Describes HQ 4.3 feature

Description

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

Default

None.

weblogic.ssl2ways.key

Describes HQ 4.3 feature

Description

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

Default

None.

weblogic.ssl2ways.key.pass

Describes HQ 4.3 feature

Description

`weblogic.ssl2ways.key.pass` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.key.pass` to the `agent.properties` file and set its value to the passphrase for the client private key:

weblogic.ssl2ways.key.pass=*ClientKey*

where *ClientKey* is the passphrase for the client private key.

Default

None.

10.6. Configure the Data to Log for Windows Events

This page describes the use of the `platform.log_track.eventfmt` agent property to customize the content of events that the HQ Agent logs for Windows Events, when log tracking is enabled for a Windows resource.

platform.log_track.eventfmt

Describes HQ 4.3 feature

Description

Specifies the content and format of the Windows event attributes that an HQ Agent includes when logging a Windows event as an event in HQ. `agent.properties` does not contain the `platform.log_track.eventfmt` property, you must explicitly add it if you want to tailor the data logged for Windows events.

Default Behavior

When Windows log tracking is enabled, an entry of this form is logged for events that match the criteria you specified on the resource's **Configuration Properties** page:

[Timestamp] Log Message (EventLogName):EventLogName:EventAttributes

where:

- **Timestamp** - is when the event occurred
- **Log Message** - is an text string
- **EventLogName** - is the Windows event log type, "System", "Security", or "Application".
- **EventAttributes** - a colon delimited string made of the Windows event **Source** and **Message** attributes.

For example, this log entry:

04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: Print: Printer HP LaserJet 6P was paused.

is for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The Windows event **Source** and **Message** attributes, are "Print" and "Printer HP LaserJet 6P was paused.", respectively.

Configuration

In HQ 4.3 and later, you can use the parameters below to configure the Windows event attributes that the agent writes for a Windows event. Each parameter maps to Windows event attribute of the same name.

- `%user%` - The name of the user on whose behalf the event occurred.
- `%computer%` - The name of the computer on which the event occurred.
- `%source%` - The software that logged the Windows event.
- `%event%` - A number identifying the particular event type.
- `%message%` - The event message.
- `%category%` - An application-specific value used for grouping events.

For example, with this property setting:

```
platform.log_track.eventfmt=%user%@%computer%   %source%:%event%:%message%
```

the HQ Agent will write the following data when logging Windows event:

```
04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: HP_Admistrator@Office
Print:7:Printer HP LaserJet 6P was paused.
```

This entry is for as for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The software associated with the event was running as "HP_Administrator" on the host "Office". The Windows event's **Source,Event,** and **Message** attributes, are "Print", "7", and "Printer HP LaserJet 6P was paused.", respectively.

After you configure the content of the log entry written for a Windows event, when you configure an alert definition for a Windows resource, you can create an alert condition based on the message content, including the custom fields you have configured. For more information, see Define Alert Condition Set.

11. Agent Properties

The properties supported in the `agent.properties` file are defined below. Note that not all supported properties appear in the default `agent.properties` file. To use a property that is not defined in `agent.properties`, you must add it explicitly.

For more information, see Section 1.4, “Agent Configuration”.

> **Looking for a property that is not listed on this page?**
>
> If your `agent.properties` file contains a property not listed here, please add a comment to this page. One explanation, although uncommon, is the use of special agent properties to override the descriptor-defined value for a `<property>` for resource instances on a particular platform. Such properties have names that reflect a resource type attribute and value. See Overriding the Value of a Resource `property` at Platform Level.

11.1. agent.eventReportBatchSize

Description

The maximum number of events that an HQ Agent will send per contact with the HQ Server.

When the agent detects an event, it puts it in a queue of events to report to the server. As long as there are events in the queue, the agent continuously creates and sends batches of events to the HQ Server. The `agent.eventReportBatchSize` sets the (maximum) number of events the agent will put in a batch.

Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 100 events per contact with the server.

11.2. agent.listenIp

Description

The IP address to which the agent binds at startup. The default value allows the agent to listen on all IP addresses on the the agent host.

Default

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to listen on all IP addresses on its host. This behavior is equivalent to to setting this property to an asterisk, like this:

*

11.3. agent.listenPort

Description

The port on the agent's listen address to which the agent binds at startup.

Default

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to listen on port 2144.

11.4. agent.logDir

Description

You can add this property to the `agent.properties` file to specify the directory where the HQ Agent will write its log file. Unless you specify a fully qualified path, `agent.logDir` is evaluated relative to the agent installation directory.

This property does not exist in `agent.properties` unless you explicitly add it. To change the location for the agent log file, add `agent.logDir` to `agent.properties` and enter a path relative to the agent installation directory, or if desired, a fully qualified path.

Note that the name of the agent log file is configured with the agent.logFile property.

Default

`agent.logDir` does not exist in `agent.properties` unless you explicitly add it. The default behavior is equivalent to this setting:

`agent.logDir=log`

resulting in the agent log file being written to the `AgentHome/log` directory.

11.5. agent.logFile

Description

Specifies the path and name of the agent log file.

Default

Note that in `agent.properties`, the default setting for `agent.LogFile` is made up of a variable and a string.

```
agent.logFile=${agent.logDir}\agent.log
```

- `agent.logDir` is a variable - it supplies the value of an identically named agent property. By default, the value of `agent.logDir` is `log`, interpreted relative to the agent installation directory.
- `agent.log` is the name for the agent log file.

So, by default, the agent log file is named `agent.log`, and is written to the `AgentHome/log` directory.

If you want the agent to the log to a different directory, you must explicitly add the agent.logDir property to `agent.properties`.

11.6. agent.logLevel

Description
Specifies the level of detail of the messages the Agent writes to the log file. Allowable values are: INFO and DEBUG.
Default
INFO

11.7. agent.logLevel.SystemErr

Description

Redirects System.err to agent.log. Commenting out this setting will cause System.err to be directed to agent.log.startup.

Default

ERROR

11.8. agent.logLevel.SystemOut

Description

Redirects System.out to agent.log. Commenting out this setting will cause System.out to be directed to agent.log.startup.

Default

INFO

11.9. agent.maxBatchSize

Description

The maximum number of metrics that the HQ Agent will send per contact with the HQ Server.

An HQ Agent puts the metrics it collects into a queue of metrics to report to the server. As long as there are metrics in the queue, the agent continuously creates and sends batches of metrics to the HQ Server. The `agent.maxBatchSize` property sets the (maximum) number of metrics the agent will put in a batch.

Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 500 metrics per contact with the server.

11.10. agent.proxyHost

Available only in HQ Enterprise

Description

The host name or IP address of the proxy server that the agent must connect to first when establishing a connection to the HQ server.

Default

none

11.11. agent.proxyPort

Available only in HQ Enterprise

Description

The port number of the proxy server that the agent must connect to first when establishing a connection to the HQ server.

Default

none

11.12. agent.setup.agentIP

Description

This specifies the IP address that the HQ Server will use to contact the HQ Agent. If the agent is on the same host as the server, value of 127.0.0.1 is valid.

If there is a firewall between the server and agent, specify the IP address of the firewall, and configure the firewall to forward traffic intended for the agent to the agent's listen address, which can be configured with agent.listenIP.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to *default*. If you use the agent.setup.* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value *default*, the HQ Server will contact the agent using the IP address that SIGAR detects on the agent host.

11.13. agent.setup.agentPort

Description

This specifies the port (on the IP address configured with agent.setup.agentIP) on the agent on which the HQ Server will communication with the agent.

If there is a firewall between the agent and the server, set agent.setup.agentPort to the appropriate port on the firewall, and configure the firewall to forward traffic intended for the agent to the agent listen port, which can be configured with.

The agent reads this value only in the event that it cannot find its connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to *default*. If you use the agent.setup.* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value *default*, the HQ Server will contact the agent on port 2144, unless SIGAR detects it is not available, in which case another default is selected.

11.14. agent.setup.camIP

Description

You can use this property to define for the agent the IP address of the HQ server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

The value can be provided as an IP address or a fully qualified domain name. To identify an server on the same host as the server, set the value to 127.0.0.1.

If there is a firewall between the agent and server, specify the address of the firewall, and configure the firewall to forward traffic on port 7080, or 7443 if you use the SSL port, to the HQ Server.

Default

Commented out, localhost.

11.15. agent.setup.camLogin

Description

You can use this property to define for the agent, at first startup after installation, the HQ username to use when registering itself with the server. The permission required on the server for this initialization is Create, for Platforms.

A login from the agent to the server is only required during the initial configuration of the agent.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, hqadmin

11.16. agent.setup.camPort

Description

You can use this property to define for the agent, at first startup after installation, what server port to use for non-secure communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, 7080.

11.17. agent.setup.camPword

Description

You can use this property to define for the agent, at first startup after installation, the password for the user specified by agent.setup.camLogin.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, hqadmin**.**

11.18. agent.setup.camSecure

Description

You can use this property to define for the agent, at first startup after installation, whether to communicate with the server over SSL. If you set this property to yes, all agent-server communications will be use the SSL secure port.

If acceptable in your environment, non-SSL communication offers improved performance for agent-server communications.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, value of yes

11.19. agent.setup.camSSLPort

Description

You can use this property to define for the agent, at first startup after installation, what server port to use for SSL communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

Commented out, 7443

11.20. agent.setup.resetupToken

Description

You can use this property to define for the agent, at first startup after installation, whether the agent will create a new token to use to authenticate with the server each time it starts up. Regenerating a token is useful if the Agent cannot connect to the server because the token has been deleted or corrupted.

Regardless of the value of this property, an agent will generate a token the first time it is started after installation.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

Default

As installed, agent.properties contains a commented out statement that sets the value to no.

11.21. agent.setup.unidirectional

Available only in HQ Enterprise

Description

Enables the unidirectional communications between HQ Agent and HQ Server, in HQ Enterprise. For more information, see Section 2.3, “Configure Unidirectional Agent - Server Communication”.

Default

Commented out, defaults to no.

11.22. agent.startupTimeOut

Description

The number of seconds that the agent startup script will wait before determining that the agent did not startup successfully. If the agent is not determined to be listening for requests within this period of time, an error is logged, and the startup script times out.

Default

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to timeout after 300 seconds.

11.23. agent.storageProvider.info

Description

The `agent.storageProvider.info` property configures where the HQ Agent's local datastore is located, its maximum size, and rules for maintaining it.

The agent stores HQ Server connection information in the datastore, and in addition, this is where the agent spools the metric and event data when the HQ Server is unresponsive or unavailable.

The datastore is contains disklists — a index and a datafile for each.

Use this property if:

- You want to store less spooled than the default limit of 100 MB. For example, in a high volume environment, you might want to avoid the "congestion" that might occur when a thousand or more agents start sending spooled data to the HQ Server after a period of server unavailability.

- Your policy requires that absolutely all spooled data be retained, even in the event that the HQ Server is down for a long time and the volume of spooled data on your agents gets vary large. (Note that the growth rate is a function of how many metrics you collect.)

Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior is equivalent to this setting:

```
agent.storageProvider.info=${agent.dataDir}|m|100|20|50
```

The default setting causes the following:

- `${agent.dataDir}` — Create the datastore in the directory specified by `agent.dataDir` property. If `agent.dataDir` is not specified, the default location for the datastore is `AgentHome/data`.
- `m` — Specifies units for the values that follow: "m" is MB, "k" is KB.
- `100` - Specifies "datafile size"; the default "datafile size" is 100 MB. Change the value if you want a smaller or larger "datafile size" threshold
- `20` — Specifies the "checksize"; when "datafile size" reaches the value of "checksize", the datafile will be checked for unused blocks. **Note**: The default value of 20 MB is rarely changed.
- `50` — Specifies "checkpercentage", the maximum (0 - 100) percentage of free space allowed in the file. This value is only significant when the "datafile size" is greater than "checkSize". **Note**: The default value of 50% is rarely changed.

The data and index files are maintained according to these rules:

When the size of the datafile exceeds "checksize" and the percentage of free space exceeds "checkpercentage", the excess free blocks will be removed by truncating the data and index files.

Note: Because truncation is performed, it is possible that even when the criteria are met, free space may not be deleted. This is a recoverable situation though, because new blocks will be inserted at the beginning of the data file.

If the size of the datafile exceeds the value of "datafile size", **all** data is deleted from storage. So, periodic maintenance is required to prevent the datafile from exceeding the limit. Increasing the frequency too much can result in unnecessary CPU cycles.

11.24. autoinventory.defaultScan.interval.millis

Description

Specifies how frequently the agent performs a default autoinventory scan.

The default scan detects servers and platform services, typically using the process table or the Windows registry. Default scans are less resource-intensive than runtime scans.

Default

Commented out, set to 86,400,000 milliseconds, or 1 day

11.25. autoinventory.runtimeScan.interval.millis

Description

Specifies how frequently the agent performs a runtime scan.

A runtime scans may use more resource-intensive methods to detect services than a default scan. For instance, a runtime scan may involve issuing an SQL query or looking up an MBean.

Default

86,400,000 milliseconds, or 1 day

11.26. http.useragent

Describes HQ 4.3 feature

Description

The `http.useragent` property defines the value for the User-Agent request header in HTTP requests issued by the HQ Agent. By default, the User-Agent in agent requests includes the HQ Agent version - and so change upon agent upgrade. If a target HTTP server is configured to block requests with an unknown User-Agent, agent requests will fail after agent upgrade.

You can use `http.useragent` to define a User-Agent value that will be consistent across upgrades.

Note: `agent.properties` does not contain this property by default. You must explicitly add it to the file.

Default

```
Hyperic-HQ-Agent/Version
```

For example:

```
Hyperic-HQ-Agent/4.1.2-EE
```

11.27. jboss.installpath

Description

To enable the agent to monitor JBoss, specify the location of the JBoss root directory.

Default

/usr/local/jboss-4.0.0

11.28. log4j Properties

```
log4j.rootLogger=${agent.logLevel}, R
log4j.appender.R.File=${agent.logFile}
log4j.appender.R.MaxBackupIndex=1
log4j.appender.R.MaxFileSize=5000KB
log4j.appender.R.layout.ConversionPattern=%d %-5p [%t] [%c{1}] %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R=org.apache.log4j.RollingFileAppender
##
## Disable overly verbose logging
##
log4j.logger.httpclient.wire=ERROR
log4j.logger.org.apache.commons.httpclient=WARN
```

```
log4j.logger.org.hyperic.hq.measurement.agent.server.SenderThread=INFO
log4j.logger.org.hyperic.hq.agent.server.AgentDListProvider=INFO
log4j.logger.org.hyperic.hq.agent.server.MeasurementSchedule=INFO
log4j.logger.org.hyperic.hq.measurement.agent.server.ScheduleThreadTrace=INFO
log4j.logger.org.hyperic.util.schedule.ScheduleTrace=INFO
##
## Only log errors from naming context
##
log4j.category.org.jnp.interfaces.NamingContext=ERROR
log4j.category.org.apache.axis=ERROR
```

11.29. platform.log_track.eventfmt

Describes HQ 4.3 feature

Description

Specifies the content and format of the Windows event attributes that an HQ Agent includes when logging a Windows event as an event in HQ. `agent.properties` does not contain the `platform.log_track.eventfmt` property, you must explicitly add it if you want to tailor the data logged for Windows events.

Default Behavior

When Windows log tracking is enabled, an entry of this form is logged for events that match the criteria you specified on the resource's **Configuration Properties** page:

[Timestamp] Log Message (EventLogName):EventLogName:EventAttributes

where:

- **Timestamp** - is when the event occurred
- **Log Message** - is an text string
- **EventLogName** - is the Windows event log type, "System", "Security", or "Application".
- **EventAttributes** - a colon delimited string made of the Windows event **Source** and **Message** attributes.

For example, this log entry:

04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: Print: Printer HP LaserJet 6P was paused.

is for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The Windows event **Source** and **Message** attributes, are "Print" and "Printer HP LaserJet 6P was paused.", respectively.

Configuration

In HQ 4.3 and later, you can use the parameters below to configure the Windows event attributes that the agent writes for a Windows event. Each parameter maps to Windows event attribute of the same name.

- `%user%` - The name of the user on whose behalf the event occurred.
- `%computer%` - The name of the computer on which the event occurred.
- `%source%` - The software that logged the Windows event.
- `%event%` - A number identifying the particular event type.

- `%message%` - The event message.
- `%category%` - An application-specific value used for grouping events.

For example, with this property setting:

```
platform.log_track.eventfmt=%user%@%computer% %source%:%event%:%message%
```

the HQ Agent will write the following data when logging Windows event:

```
04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: HP_Admistrator@Office Print:7:Printer HP
LaserJet 6P was paused.
```

This entry is for as for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The software associated with the event was running as "HP_Administrator" on the host "Office". The Windows event's **Source,Event,** and **Message** attributes, are "Print", "7", and "Printer HP LaserJet 6P was paused.", respectively.

After you configure the content of the log entry written for a Windows event, when you configure an alert definition for a Windows resource, you can create an alert condition based on the message content, including the custom fields you have configured. For more information, see Define Alert Condition Set.

11.30. plugins.exclude

Description

Use this property to specify plugins that you do not wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

Usage

Supply a comma-separated list of plugins to exclude, for example:

```
{{plugins.exclude=jboss,apache,mysql}}
```

11.31. plugins.include

Description

Use this property to specify plugins that you do wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

Usage

Supply a comma-separated list of plugins to include, for example:

```
plugins.include=weblogic,apache
```

11.32. sigar.mirror.procnet

Description

mirror /proc/net/tcp on linux

Default

true

11.33. snmpTrapReceiver.listenAddress

Description
Use this property to specify the port on which the HQ Agent listens for SNMP traps. By default, the agent is not configured to listen for SNMP traps. You must add this property to `agent.properties` to enable the agent to receive traps.

Typically SNMP uses the UDP port 162 for trap messages. This port is in the privileged range, so an agent listening for trap messages on it must run as root (or as an Administrative user on Windows).

If you prefer to run the the agent under the context of a non-administrative user, you can configure it to listen for trap messages on an unprivileged port.

Usage
Specify an IP address (or 0.0.0.0 to specify all interfaces on the platform) and the port for UDP communications in this format:

```
snmpTrapReceiver.listenAddress=udp:IP_address/port
```

To enable the HQ Agent to receive SNMP traps on an unprivileged port, specify port 1024 or above. The following setting allows the agent to receive traps on any interface on the platform, on UDP port 1620.

```
snmpTrapReceiver.listenAddress=udp:0.0.0.0/1620
```

11.34. weblogic.auth.method

View Source
Describes HQ 4.3 feature

Description

`weblogic.auth.method` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add the following line to the `agent.properties` file to specify that the agent will use Two-Way-SSL for communications with the Administration Server.

```
weblogic.auth.method=ssl2ways
```

Default

None.

11.35. weblogic.installpath

Description

To enable the agent to monitor WebLogic 8.1, specify the location server/lib/weblogic.jar

Default

/usr/local/bea/weblogic-8.1

11.36. weblogic.ssl2ways.cert

Describes HQ 4.3 feature

Description

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

Default

None.

11.37. weblogic.ssl2ways.key

Describes HQ 4.3 feature
Description

`weblogic.ssl2ways.key` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

`weblogic.ssl2ways.key` to the `agent.properties` file and set its value to the location of client's private key:

```
weblogic.ssl2ways.key=clientKey.pem
```

where:

clientkey.pem is the **path to the private key** the HQ Agent presents to the Administration Server that the agent manages.

Default

None.

11.38. weblogic.ssl2ways.key.pass

Describes HQ 4.3 feature

Description

`weblogic.ssl2ways.key.pass` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.key.pass` to the `agent.properties` file and set its value to the passphrase for the client private key:

weblogic.ssl2ways.key.pass=*ClientKey*

where *ClientKey* is the passphrase for the client private key.

Default

None.

11.39. websphere.installpath

Description

To enable the agent to monitor WebSphere, specify the location of the WebSphere jars.

Default

/opt/WebSphere/AppServer

11.40. websphere.useext

Description

This property is required to enable management of WebSphere 6.0 and 6.1.

Do **not** define the `websphere.useext` property to monitor WebSphere 7.

Usage

Add the following property definition to the `agent.properties file` for an an HQ Agent that will manage WebSphere 6.0 or 6.1.

```
websphere.useext=true
```