Dynamic Language Agent Proxy

On this page:

Related pages:

- Configure the Agent for PHP CLI Applications
- Use a Shared Proxy for PHP Agents
- Start the PHP Agent Proxy Manually
- Start the Proxy Manually for Node.js
- Sharing a Proxy Among Node.js Agents
- Start the Python Agent Proxy Manually

The AppDynamics proxy is a Java daemon process that handles communication between the following agents and the Controller:

- Node.js
- PHP
- Python
- Web Server

The proxy reports data collected by the agent to the Controller, which stores, baselines, and analyzes it.

This topic presents basic information to help you examine and resolve proxy issues that may prevent an application agent from connecting to the Controller or reporting data correctly. For detailed information about how the proxy works with a specific agent, see the appropriate **Related pages**.

Proxy Basics

Single or Multi-Tenant

For the Node.js and PHP agents, the proxy can be single-tenant (one proxy per agent) or multi-tenant (multiple agents communicating through a single proxy), depending on the way the agents are configured. The default setup for these agents is single-tenant, but there are circumstances for which a mutli-tenant proxy is required or desirable. The Apache Web Server and Python agents always communicate through a multi-tenant proxy when there are multiple agents on a machine.

Automatic or Manual Startup and Shutdown

Typically the proxy is automatically started when the application agent starts up, but in some cases you need to launch it manually. These cases vary depending on both the particular agent and the application environment.

After the proxy is started (automatically or manually) it registers with the Controller and requests the agent configuration. The agent must receive the configuration from the Controller via the proxy before it can report metrics.

If the proxy was automatically started, the agent is supposed to shut down the proxy as part of its own cleanup procedures. If the proxy was manually started, it must be shut down manually.

Proxy Logs

If the proxy is running, you can check the proxy log to examine connection issues.

The proxy log files are named proxy.<timestamp>.log.

If the proxy is not running, examine the agent logs for the typical Java startup signature to see if the proxy started.

See the agent-specific documentation for the location of these logs:

- **Node.js**: Default is /tmp/appd/logs; see Install the Node.js Agent for details.
- **PHP**: Default is <php_agent_install_dir>/logs; see Install the PHP Agent for more details.
- **Python**: Default is /tmp/appd/logs; see Python Agent Debugging and Logging for details.
- **Web Server**: <webserver_agent_install>/logs/proxy_$date.lo

Proxy Not Started Issues

The agent will not work if the proxy did not start.

Determine whether the proxy started by running this command:

```
ps aux|grep java
```

If the proxy is running, you should see "java" and "proxy" in the output, something like this:

```
/usr/lib/appdynamics-php5/proxy/jre/bin/java -server -Xmx120m -classpath /usr/lib/appdynamics-php5
/proxy/conf/logging/*:/usr/lib/appdynamics-php5/proxy/lib/*:/usr/lib/appdynamics-php5/proxy/lib/tp/*:
/usr/lib/appdynamics-php5/proxy/* -Djava.library.path=/usr/lib/appdynamics-php5/proxy/lib/tp -
Dappdynamics.agent.logs.dir=/usr/lib/appdynamics-php5/logs -Dcomm=/tmp/ad-siJ4rp -
DagentType=PHP_APP_AGENT -Dappdynamics.agent.runtime.dir=/usr/lib/appdynamics-php5/proxy com.
appdynamics.ee.agent.proxy.kernel.Prox
```

If the proxy did not start, the most common reason is insufficient permissions.

- The agent installation directory, and its proxy control subdirectory, must be readable and executable by all and writable by the directory owner:

  ```
  chmod -R 755 <agent_install_dir>
  ```

- The agent installation directory must be owned by the instrumented application user. Who this user is depends on the platform. It could be the Apache user, the python container user, the nginx user, the node.js process, etc.

  ```
  chown -R <appuser>:<appsuser> <agent_install_dir>
  ```

If the proxy did not start, set these permissions and try again.

Proxy Connection Issues

If the proxy started but you don't have a connection to the Controller, you may be using the wrong controller information.

Examine the proxy log and verify that there are no typos in the controller domain name or port and that SSL is enabled setting for an SSL connection. If there are mistakes, edit the controller/port/SSL values in the configuration. These settings are on the first page of the Agent Download and Install Wizard or the agent settings (require statement for Node.js) if you installed the agent manually.

If you have verified that the controller settings are correct and the proxy still does not connect to the Controller, telnet the SaaS Controller:

```
telnet <your_account>.saas.appdynamics.com 443
```

If you cannot access the Controller via telnet, examine the agent-registered messages in the logs that indicate why the proxy is unable to connect, such as the existence of a firewall or other obstacle at your site. Work with your administrator to see if you can resolve the issue.

If you can reach the Controller via telnet, there may be a problem on the Controller side.

Node.js proxyless option

To use the Node.js agent without the Java proxy, set the property libagent: true in your require('appdynamics') statement.

The following set of features will not be available when using the agent without a Java proxy:

- Backend configuration (only default backends are supported)
- Log file upload
- Object instance tracking