OpenShift Enterprise
2
Troubleshooting Guide

Troubleshooting OpenShift Enterprise

Red Hat OpenShift Documentation
Team

OpenShift Enterprise 2 Troubleshooting Guide

Troubleshooting OpenShift Enterprise

Red Hat OpenShift Documentation Team

Legal Notice

Abstract

The OpenShift Enterprise Troubleshooting Guide provides information on resolving common problems with OpenShift Enterprise installations. This guide provides the following information to help diagnose and fix general problems: Configuration of standard Linux components and corresponding log files Configuration of OpenShift Enterprise components and corresponding log files Recognizing common system problems Error messages that may occur when creating applications This guide is intended for experienced system administrators.

Table of Contents

Chapter 1. Introduction to OpenShift Enterprise

OpenShift Enterprise by Red Hat is a Platform as a Service (PaaS) that provides developers and IT organizations with an auto-scaling, cloud application platform for deploying new applications on secure, scalable resources with minimal configuration and management overhead. OpenShift Enterprise supports a wide selection of programming languages and frameworks, such as Java, Ruby, and PHP. Integrated developer tools, such as Eclipse integration, JBoss Developer Studio, and Jenkins, support the application life cycle.

Built on Red Hat Enterprise Linux, OpenShift Enterprise provides a secure and scalable multi-tenant operating system for today's enterprise-class applications while providing integrated application runtimes and libraries.

OpenShift Enterprise brings the OpenShift PaaS platform to customer data centers, enabling organizations to implement a private PaaS that meets security, privacy, compliance, and governance requirements.

1.1. What's New in Current Release

For a complete list of all the new features available in the current release of OpenShift Enterprise, see the current edition of the *OpenShift Enterprise Release Notes* at https://access.redhat.com/site/documentation. New features that are available in the current release are documented in the respective sections of this book.

Chapter 2. Log Files and Validation Scripts

This chapter helps you locate various log files and other information that can aid in troubleshooting some of the common issues with OpenShift Enterprise.

2.1. Configuration and Log Files for Standard Linux Components

OpenShift Enterprise uses mostly standard Linux components such as networking, httpd, SELinux, and others. Any prior administration experience with these components will be helpful in troubleshooting common issues with your OpenShift Enterprise deployment.

2.1.1. General Information

General information can be found in the **`/var/log/messages`** file. This serves as a good starting point to investigate issues that might not be logged anywhere else.

The **`/var/log/httpd/access_log`** file shows whether your web request was received by the host.

The **`/var/log/httpd/error_log`** file can be helpful in troubleshooting certain problems on broker and node hosts.

The **`/var/log/audit/audit.log`** file is useful for finding problems that might be caused by SELinux violations.

The **`/var/log/secure`** file logs user and SSH interactions. Because users can SSH into their gears, and all Git requests also authenticate using SSH, this file is useful for checking interaction with gears on node hosts.

2.1.2. Networking

DNS

The best place for Linux operators to begin troubleshooting DNS problems on broker, node, or client hosts is the **`/etc/resolv.conf`** file. On client hosts running other operating systems, look in the appropriate network configuration file.

If your OpenShift Enterprise installation uses a BIND server, this should be listed in the **`/etc/resolv.conf`** file as the first nameserver.

On client hosts, the first nameserver listed in the **`/etc/resolv.conf`** file should point to your OpenShift Enterprise installation, either receiving updates from it, or delegating the domain to the nameserver of your installation.

If a hostname of your OpenShift Enterprise installation does not resolve correctly, use the following command to find the server the response is actually coming from:

```
# dig hostname
```

The application hostname is a CNAME for the node host DNS record. However, for a scaled application, this command will only show which node host contains the HAProxy gear; other gears could reside on different node hosts.

BIND

If you are running a BIND server on the broker (or supporting) host, the configuration information is contained in the **/var/named/dynamic** directory. The zone file syntax is ***domain*.com.db.zone**; so if the domain of your OpenShift Enterprise installation is *example.com*, the zone file name would be **example.com.db.zone**. However, not all changes will be in the zone file. Recent changes can be contained in a binary journal file.

Use the following command to view the entire zone according to the nameserver:

```
# dig domain axfr
```

DHCP

For broker and node hosts, DHCP is currently only supported if the host IPs are pinned, meaning they do not change during lease renewal. This also applies to nameservers, in that they should also not change if pinned.

If DHCP is in use, networking parameters will update at boot time, or at lease renewal. If DNS resolution on an OpenShift Enterprise host stops working after initial installation, look in the **/etc/dhcp/dhclient-*network-interface*.conf** file to verify the nameservers provided by the DHCP service are being overwritten when a new lease is obtained.

If your configuration in the **/etc/resolv.conf** file is overwritten with incorrect values, check your configuration in the **dhclient-*network-interface*.conf** file.

2.1.3. SELinux

When error messages indicate access denials but standard Linux file permissions appear to allow access, it could be due to SELinux policy. Use the following method for troubleshooting SELinux issues or verify whether the problem is SELinux related. Red Hat recommends this procedure for finding all SELinux related problems.

Procedure 2.1. To Troubleshoot SELinux Issues:

1. As root, run the following command to set SELinux to permissive mode:

   ```
   # setenforce 0
   ```

2. Retry the failing action. If the action succeeds then the issue is SELinux related.

3. Run the following command to set SELinux back to enforcing mode:

   ```
   # setenforce 1
   ```

4. Check the **/var/log/audit/audit.log** file for any SELinux violations.

This allows the offending action to proceed and log everything that otherwise would have been denied with SELinux enforced. In enforcing mode, not all denials are logged in the audit log, and the first denial generally blocks the action from proceeding to what might be further denials.

2.1.4. Control Groups on Node Hosts

Control groups (cgroups) enable you to allocate resources such as CPU time, system memory, and network bandwidth among user-defined groups of tasks (processes) running on a system. When the **cgconfig** service is running correctly on a node host, you see the following:

- The **/etc/cgconfig.conf** file exists with the SELinux label: **system_u:object_r:cgconfig_etc_t:s0**.

- The **`/etc/cgconfig.conf`** file joins CPU, cpuacct, memory, freezer, and net_cls in the **`/croup/all`** directory.
- The **`/cgroup`** directory exists, with the SELinux label: **`system_u:object_r:cgroup_t:s0`**.
- The *cgconfig* service is running.

When the *cgred* service is running correctly, you see the following:

- The **`/etc/cgrules.conf`** file exists with the SELinux label: **`system_u:object_r:cgrules_etc_t:s0`**
- The *cgred* service is running.

If there are gears running on the node host, you also see the following:

- A line for each gear in the **`/etc/cgrules.conf`** file.
- A directory for each gear in the **`/cgroup/all/openshift`** directory.
- All processes with the gear UUID are listed in the gear's **`cgroup.procs`** file. This file is located in the **`/cgroup/all/openshift/`*`gear_UUID`*** directory.

Important

If you created the configuration files interactively as a root user, the SELinux user label would be ***`unconfined_u`*** and not ***`system_u`***. For example, the SELinux label in **`/etc/cgconfig.conf`** would be ***`unconfined_u:object_r:cgconfig_etc_t:s0`***.

2.1.5. Pluggable Authentication Modules

The *pam_limits* module controls access to system resources. Specifically, OpenShift Enterprise uses the **`nproc`** value to control the number of processes a given account can create.

The default value for a new gear is configured in the **`/etc/openshift/resource_limits.conf`** file on the node host:

```
limits_nproc=2048
```

When a new gear is created, a ***`84-gear_UUID`*`.conf`** file is created on the node host, in the **`/etc/security/limits.d`** directory. Replace *gear_UUID* with the UNIX account name for the gear. This contains a rule set that defines the limits for that UNIX account. The first field of each line in the file is the gear UUID.

The **`nproc`** limit for an individual gear is increased by changing the value in the ***`84-gear_UUID`*`.conf`** file:

```
# PAM process limits for guest
# see limits.conf(5) for details
#Each line describes a limit for a user in the form:
#
#<domain> <type> <item> <value>
32ec916eeaa04032b1481af5037a6dfb hard nproc 250
```

If a gear process is failing, check the **`nproc`** limit.

2.1.6. Disk Quotas

Verify that the mount point for the **`/var/lib/openshift`** directory has the **`usrquota`** option enabled in the **`/etc/fstab`** file, and has been mounted. Remount the directory if necessary using the command shown below, and check the output.

```
# mount -o remount filesystem
```

Use the following command to verify that the quotas are configured correctly:

```
# repquota -a
```

2.1.7. iptables

Network firewalls are implemented with iptables. Use the following command to view the current policies:

```
# iptables -L
```

The screen outputs for the **`iptables -L`** command for both a broker host and a node host are shown below.

iptables Policy for Broker Host

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state
RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:domain
ACCEPT     udp  --  anywhere             anywhere            state NEW udp
dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:https
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:http
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:61613
REJECT     all  --  anywhere             anywhere            reject-with
icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  anywhere             anywhere            reject-with
icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

iptables Policy for Node Host

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state
RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:https
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:http
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpts:35531:65535
REJECT     all  --  anywhere             anywhere            reject-with
icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  anywhere             anywhere            reject-with
icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

2.2. Configuration and Log Files for OpenShift Components

This section provides you with the locations of various configuration and log files for OpenShift Enterprise and related components.

2.2.1. General Configuration

The **`/etc/openshift`** directory contains the most important configuration files for OpenShift Enterprise. These configuration files correspond to the type of installation; for example, a broker host, node host, or a client host. Check the corresponding configuration file to verify that the settings are suitable for your system.

2.2.2. Broker Host Failures

In the **`/var/log/openshift/broker/httpd/`** directory, check the **`access_log`** and **`error_log`** files when user interactions with the broker host are failing. Verify that the request was authenticated and forwarded to the broker application.

The broker application is a Rails application, and all associated logs are saved in the **`/var/log/openshift/broker/production.log`** file.

User actions for Rails are stored in the **`/var/log/openshift/broker/user_action.log`** file. This log file includes gears created and deleted by a user. However, the logs do not include gear UUIDs.

2.2.3. MCollective

MCollective is used for any node host communications or interactions with either the broker host or other nodes. This can include gear placement or removal.

Use the following command to verify that a broker can communicate with all node hosts:

```
# oo-mco ping
broker.mydomain.com                          time=134.85 ms
node.mydomain.com                            time=541.41 ms
node1.mydomain.com                           time=572.76 ms


---- ping statistics ----
3 replies max: 572.76 min: 134.85 avg: 416.34
```

All configured node hosts should be represented in the output. If you do not see a node host as expected, verify that the network and clock settings are configured correctly for that node host.

> **Note**
>
> If the **`oo-mco ping`** command is not running successfully, it could be that*openshift-origin-util-scl* is not properly installed on your machine, or that **`oo-mco ping`** is missing. Install the *openshift-origin-util-scl* package in order to run the command.

Communication between broker and node hosts is logged in the following files:

- **`/var/log/openshift/node/ruby193-mcollective.log`** on node hosts
- **`/var/log/openshift/broker/ruby193-mcollective-client.log`** on broker hosts

Communication that is dropped due to exceeding TTL is also logged in these files.

MCollective actions performed on the node can also log more details in **`/var/log/openshift/node/platform.log`** and **`/var/log/openshift/node/platform-trace.log.`**

If necessary, you can determine on which node host an application is created using the **`dig`** or **`host`** command, with the application's hostname.

2.2.4. Gears

All gear information is stored in the **`/var/lib/openshift`** directory on that gear's node host, and represented by the gear's UUID. This directory contains the following information:

- Gears themselves
- Web server configuration
- Operation directories

Use the **`ls`** command to show the contents of the **`/var/lib/openshift/.httpd.d`** directory.

```
# ls /var/lib/openshift/.httpd.d/
aliases.db   frontend-mod-rewrite-https-template.erb  idler.db   nodes.db
routes.json  sts.txt
aliases.txt  geardb.json                              idler.txt  nodes.txt
sts.db
```

Gear UUIDs are also represented as users in the **/etc/passwd** file.

2.3. Validation Scripts

OpenShift Enterprise includes validation scripts that detect and report potential problems with broker and node hosts. Red Hat recommends that these scripts are used for monitoring purposes when performing the initial installation. After initial installation, these scripts can help troubleshoot any OpenShift Enterprise problems. Validation scripts detect common user errors, as well as unexpected failures. See the *OpenShift Enterprise Administration Guide* at https://access.redhat.com/site/documentation for more information on validation scripts.

2.3.1. Broker Host Scripts

This section describes validation scripts you can use to troubleshoot issues with broker hosts.

2.3.1.1. Verifying Broker Host Configuration

Use the **oo-accept-broker** script without any options to report potential problems in the broker host configuration. The output from this script indicates how many problems are found.

2.3.1.2. Fixing Gear Discrepancies

Use the **oo-admin-chk** script without any options to compare gear records in the broker's Mongo datastore to the gears actually present on the node hosts. The script reports any discrepancies that are found.

Example 2.1. Diagnosing Problems Using oo-admin-chk

```
# oo-admin-chk
Check failed.
 FAIL - user user@domain.com has a mismatch in consumed gears (-1) and
actual gears (0)!
```

This indicates a mismatch between the number of consumed gears and the number of actual gears, which can occur under certain race conditions.

This problem can be fixed using the **oo-admin-ctl-user** command:

```
# oo-admin-ctl-user -l user@domain.com --setconsumedgears 0
```

Example 2.2. Diagnosing Problems Using oo-admin-chk

```
# oo-admin-chk
Gear 9bb07b76dca44c3b939c9042ecf1e2fe exists on node [node1.example.com,
uid:2828] but does not exist in mongo database
```

This output indicates that although a gear was destroyed from the broker host's MongoDB, it was not completely removed from the node host. This can be due to storage issues or other unexpected failures. You can fix this issue by deleting the gear from the **/var/lib/openshift** directory, and removing the user from the node host.

These are two examples of the issues that could occur when running the **`oo-admin-chk`** script. The script should be self-explanatory to resolve most problems.

2.3.2. Node Host Scripts

This section describes validation scripts you can use to troubleshoot issues with node hosts.

2.3.2.1. Verifying Node Host Configuration

Use the **`oo-accept-node`** script without any options to report potential problems in the node host configuration. The output from this script indicates how many problems are found.

2.3.3. Additional Diagnostics

The **`oo-diagnostics`** script can be run on any OpenShift Enterprise host to diagnose common problems and provide potential solutions. It can also be helpful for gathering information (particularly when run with the **`-v`** option for verbose output) to provide to Red Hat Support when opening a support case.

Chapter 3. Recognizing System Problems

This chapter describes common OpenShift Enterprise system problems, and provides information to troubleshoot and resolve these issues.

3.1. Missing Repositories

If you are having issues installing the required OpenShift Enterprise software packages, verify that you have the correct YUM repositories configured. If you are using the classic RHN subscription, you must enable the correct channels. Both the broker and node host installations require a Red Hat Enterprise Linux 6 subscription.

Furthermore, each type of installation requires the designated RHN channels and repositories. Therefore, if you are installing a broker host, RHN channels and repositories that are specific to the broker host must be enabled. Similarly, repositories and RHN channels specific to the node host must be enabled when installing a node host.

The following table shows the required repositories for OpenShift Enterprise, with a brief description of each.

Table 3.1. List of Repositories

Name of Repository	Description
Red Hat OpenShift Enterprise Infrastructure	Broker / BIND / Mongo hosts
Red Hat OpenShift Enterprise Application Node	Node hosts
Red Hat OpenShift Enterprise Client Tools	Client hosts
Red Hat OpenShift Enterprise JBoss EAP add-on	Included with EAP support purchase. See note below.
Red Hat OpenShift Enterprise Application Platform	Included with EAP support purchase. See note below.
Red Hat OpenShift Enterprise Web Server	Included with bundle purchase. See note below.

Note

These repositories are required on node hosts for JBoss cartridge support.

Important

Configuring the wrong RHN channels or repositories can result in a failed installation.

3.2. Missing Node Host

When you run the **`oo-mco ping`** command on the broker host, all node hosts should be listed in the output. Although applications on an unlisted node host can continue to operate without problems, the unlisted node hosts are not controlled by the broker host.

Node hosts may not respond to the **`oo-mco ping`** command if the clock on the broker host is not synchronized with the clock on the node host. MCollective messages have a TTL of 60 seconds. Therefore, if the clocks are not synchronized the MCollective messages can be dropped, causing communication issues.

Verify that the broker host and node host clocks are synchronized, and the *ntpd* service is enabled. All configured hosts must use the same NTP server.

Clock synchronization issues are logged in the **/var/log/openshift/node/ruby193-mcollective.log** file on the node host, and could look like the following sample screen output:

```
W, [2012-08-10T14:27:01.526544 #12179]  WARN -- : runner.rb:62:in `run'
Message 8beea9354f9784de939ec5693940d5ce from uid=48@broker.example.com
created at 1344622854 is 367 seconds old, TTL is 60
```

The node host also may not respond to the **oo-mco ping** command if ActiveMQ on the broker host cannot communicate with MCollective on the node host. Verify that the **ruby193-mcollective** service is running on the node host, and it can communicate with ActiveMQ on the broker host. If a configuration has been modified recently, use the following command to restart the **ruby193-mcollective** service:

```
# service ruby193-mcollective restart
```

3.3. Broker Application Response Failure

The broker host service is a Rails application that runs inside the **Passenger** service. There can be cases when the broker host service appears to be running, but in reality is not. If the **Passenger** service fails to start for some reason, the broker host service will not start, even if the **httpd** service is running. So even though the **service openshift-broker start** command reports success, the service may not actually be running.

Failures and problems with the **Passenger** service are logged in the **/var/www/openshift/broker/httpd/logs/error_log** file on the broker host, as shown in the following screen output:

```
[Wed Oct 17 23:48:04 2012] [error] *** Passenger could not be initialized
because of this error: Unable to start the Phusion Passenger watchdog
(/usr/lib64/gems/exts/passenger-3.0.17/agents/PassengerWatchdog): Permission
denied (13)
```

This shows that the **Passenger** service has failed to start. This can be caused by dependency issues with the **RubyGems** package, which often occurs when **Bundler** attempts to regenerate the **/var/www/openshift/broker/Gemfile.lock** file.

Run the following commands to verify whether this problem exists:

```
# cd /var/www/openshift/broker/
# bundle --local
 Could not find rack-1.3.0 in any of the sources
```

This shows that the specified dependency was not found. Updating all Ruby gems, and restarting the **openshift-broker** service could resolve this issue.

3.3.1. Missing Gems with Validation Scripts

When running validation scripts, you could also experience problems relating to **Bundler** and **RubyGems** dependencies. This is because the validation scripts, such as **oo-admin-chk**, use the broker Rails configuration and also depend on the **/var/www/openshift/broker/Gemfile.lock** file, as shown in the following sample output:

```
# oo-admin-chk
 Could not find rack-1.3.0 in any of the sources
 Run `bundle install` to install missing gems.
```

Restarting the **openshift-broker** service will regenerate the **Gemfile.lock** file, and could solve this issue. Be sure to run the **yum update** command before restarting the **openshift-broker** service.

Warning

Do not run the **bundle install** command as the output asks you to do. Running this command will download and install unsupported and untested software packages, resulting in problems with your OpenShift Enterprise installation.

3.4. DNS Propagation Fails when Creating an Application

Creating an application can fail if the application's hostname cannot be resolved. This problem typically occurs if a client host in an OpenShift Enterprise deployment does not have the broker host correctly configured as a nameserver, either directly or through delegation. An example of such a failure is shown in the following sample screen output:

```
$ rhc app create -a myapp -t jbossas-7
Creating application: myapp in domain
Now your new domain name is being propagated (this might take a minute)...
........
   retry # 3 - Waiting for DNS: myapp-domain.example.com
```

Eventually the process will timeout while attempting to resolve the application's hostname.

This issue can indicate that broker DNS updates are failing, or that the client host does not have the correct nameserver configured to receive DNS updates from the broker host.

This problem can also occur if the **PUBLIC_HOSTNAME** setting in the **/etc/openshift/node.conf** file on the node host is incorrectly configured.

Note

The **oo-admin-chk** script on the broker host can help detect this problem.

3.5. Developers Connecting to a Gear are Disconnected Immediately

When connecting directly to a gear, for example when a **git clone** is performed, a developer can authenticate successfully, but then be disconnected by the remote host. This could be due to PAM being misconfigured. An example of this error is shown in the output below.

```
$ rhc app create -n apps -t php -a testing

Creating application 'testing'
==============================
  Scaling:   no
  Gear Size: default
  Cartridge: php
  Namespace: apps
Password: ********

Your application's domain name is being propagated worldwide (this might
take a minute)...
The authenticity of host 'testing-apps.example.com (x.x.x.x)' can't be
established.
RSA key fingerprint is [...].
Are you sure you want to continue connecting (yes/no)? yes
Initialized empty Git repository in /home/test/testing/.git/
done

Error in git clone - Warning: Permanently added 'testing-apps.example.com'
(RSA) to the list of known hosts.
Traceback (most recent call last):
  File "/usr/bin/oo-trap-user", line 134, in <module>
 read_env_vars()
  File "/usr/bin/oo-trap-user", line 64, in read_env_vars
 fp = open(os.path.expanduser('~/.env/') + env, 'r')
IOError: [Errno 13] Permission denied:
'/var/lib/openshift/a7a330ee62ae467ca6d74cd0ce29742a/.env/OPENSHIFT_APP_NAME
'

fatal: The remote end hung up unexpectedly
```

Ensure that PAM is correctly configured by performing the steps located in the *Configuring PAM* section of the *OpenShift Enterprise Deployment Guide*. Note that ***pam_selinux*** should be changed to ***pam_openshift*** in **/etc/pam.d/sshd**, and a line with ***pam_namespace.so*** should be at the end of each file modified. If your change management system overwrote these settings, ensure that your system will retain the correctly modified files in the future.

3.6. Gears Not Idling

The **oddjob** daemon must be running on node hosts for gear idling to work correctly. Error messages for gear idling issues are logged in the **/var/log/httpd/error_log** file on the node host. The following error message, from the **error-log** file, shows that the **oddjob** daemon is not running.

```
org.freedesktop.DBus.Error.ServiceUnknown: The name
com.redhat.oddjob_openshift was not provided by any .service files
```

Use the following commands to start the **oddjob** daemon, and enable it to start at boot:

```
# service oddjobd start
# chkconfig oddjobd on
```

3.7. cgconfig Service Fails to Start

If the **cgconfig** service fails to start, look for AVC messages in the **/var/log/audit/audit.log** SELinux audit log file. The error messages could indicate incorrect SELinux labels in the following files and directories:

- **/etc/cgconfig.conf**
- **/etc/cgrules.conf**
- **/cgroup**

Use the **restorecon -v *filename*** command to restore the correct SELinux labels for each of the files:

```
# restorecon -v /etc/cgconfig.conf
# restorecon -v /etc/cgrules.conf
# restorecon -rv /cgroup
```

This restores the correct SELinux labels in the **/etc/cgrules.conf** file.

Next, start the **cgconfig** service using the following command:

```
# service cgconfig start
```

3.8. MongoDB Failures

If MongoDB is not configured correctly, you will experience failures with the OpenShift client tools. If this is the case, verify your MongoDB configuration.

Use the following command to verify that the MongoDB service is running:

```
# service mongod status
```

If the **mongod** service is not running, look in the **/var/log/mongodb/mongodb.log** file for information. Look for duplicate configuration lines, which cause problems with MongoDB, and result in the *multiple_occurences* error message. Verify that there are no duplicate configuration lines in the **/etc/mongodb.conf** file to enable the **mongod** service to start.

For testing a MongoDB connection, look in the **/etc/openshift/broker.conf** file for MongoDB configuration details such as database host, port, name, user, and password.

Example 3.1. Example MongoDB Configuration

```
MONGO_HOST_PORT="localhost:27017"
MONGO_USER="mongouser"
MONGO_PASSWORD="mongopassword"
MONGO_DB="openshift_broker"
MONGO_SSL="false"
```

With the **mongod** service running, use the following command to connect to the database, replacing configuration settings accordingly:

```
# mongo localhost:27017/openshift_broker -u mongouser -p mongopassword
```

The MongoDB command prompt is displayed.

3.9. Jenkins Build Failures

If a gear running Jenkins is already present when the **`AUTH_SALT`** setting is changed in the **`/etc/openshift/broker.conf`** file, subsequent Jenkins builds will initially fail with the following:

```
remote: Executing Jenkins build.
remote:
remote: You can track your build at https://jenkins-
namespace.example.com/job/myapp-build
remote:
remote: Waiting for build to schedule..........................
remote: **BUILD FAILED/CANCELLED**
remote: Please see the Jenkins log for more details via rhc-tail-files
remote: !!!!!!!!
remote: Deployment Halted!
remote: If the build failed before the deploy step, your previous
remote: build is still running.  Otherwise, your application may be
remote: partially deployed or inaccessible.
remote: Fix the build and try again.
remote: !!!!!!!!
```

Checking the Jenkins application's logs will reveal the following invalid credential messages:

```
# rhc tail jenkins
...
WARNING: Caught com.openshift.client.InvalidCredentialsOpenShiftException:
Your credentials are not authorized to access
"https://broker.example.com/broker/rest/user". Will retry 4 more times
before canceling build.
com.openshift.client.InvalidCredentialsOpenShiftException: Your credentials
are not authorized to access "https://broker.example.com/broker/rest/user"
...
```

To address these issues, first restart the broker service:

```
# service openshift-broker restart
```

Then run the **`oo-admin-broker-auth`** tool to rekey the gears' authorization tokens. To rekey the tokens for all applicable gears, run the tool with the **`--rekey-all`** option:

```
# oo-admin-broker-auth --rekey-all
```

See the command's **`--help`** output and man page for additional options and more detailed use cases.

When the gears are successfully rekeyed, future builds should continue as normal.

3.10. Outdated Cartridge List

If a newly installed cartridge is not immediately available, it could be due to an outdated, cached cartridge list. The first time the REST API is accessed, the broker host uses MCollective to retrieve the list of available cartridges from a node host. By default, this list is cached for six hours in a production environment. If the installed cartridges are modified, the cache must be cleared either manually, or by waiting until the cache expires before developers can access the updated list.

In addition, the Management Console has its own cached list of installed cartridges that must be cleared for any changes to be observed.

Use the following command on each broker in your OpenShift Enterprise installation to clear the broker cache only:

```
# oo-admin-broker-cache --clear
```

Use the following command on each broker in your OpenShift Enterprise installation to clear the broker and Console cache (the **`--console`** option implies **`--clear`**):

```
# oo-admin-broker-cache --console
```

Chapter 4. Error Messages when Creating Applications

This chapter describes error messages you could encounter when creating applications on OpenShift Enterprise, and possible resolutions.

4.1. cpu.cfs_quota_us: No such file

The **`rhc app create`** command can fail to create an application if *cgroups* are not working properly. These error messages are logged in the **`/var/log/openshift/node/ruby193-mcollective.log`** file on the node host, and can look like the following:

```
/cgroup/all/openshift/*/cpu.cfs_quota_us: No such file
```

See Section 2.1.4, “Control Groups on Node Hosts” for instructions on how to verify that *cgroups* are correctly configured.

4.2. Password Prompt

When you create a new application on OpenShift Enterprise, the client host attempts to clone the remote Git repository, which is located in the application's gear on a node host. The SSH authentication process is invoked to log in to the hostname of the application's gear, which is a newly created CNAME (alias) for the public hostname of the node host.

To successfully clone the remote Git repository of the new application, **`PUBLIC_HOSTNAME`** must be configured correctly in the **`/etc/openshift/node.conf`** file on the node host.

Sometimes when creating an application, the process halts and you are prompted for a password, as shown in the following sample screen output:

```
The authenticity of host 'myapp-domain.example.com (::1)' can't be
established.
RSA key fingerprint is 88:49:43:d2:e9:b4:4d:84:a1:d6:8a:30:85:73:d7:7f.
Are you sure you want to continue connecting (yes/no)? yes
e9bdfc309bef4c13889a21ddbea45f@myapp-domain.example.com's password:
```

This can occur when **`PUBLIC_HOSTNAME`** resolves to the wrong IP address. In this case, **`PUBLIC_HOSTNAME`** is set to **`localhost.localdomain`**, as shown in the sample screen output below.

```
PUBLIC_HOSTNAME=localhost.localdomain
```

In this example the application's gear CNAME is created using **`localhost.localdomain`** as the hostname for the node host. When Git attempts to authenticate using the gear user ID and SSH key, the SSH authentication fails because the application gear does not exist on **`localhost.localdomain`**, and you are prompted for a password.

The first line of the sample screen output states that the IP address for the application's gear is **`(::1)`**, which is pointing to localhost, and is not a valid IP for an application's gear. Verify that the IP address of an application's gear is a valid IP address of the node host.

In cases where **`PUBLIC_HOSTNAME`** fails to resolve at all as a FQDN, DNS resolution times out and the Git clone process fails.

Note

The **oo-admin-chk** script on the broker host can help detect this problem.

4.3. Communication Issue after Node Host Reboot

After rebooting a node host, the **rhc app create** command can fail to create an application, resulting in the following error:

```
An error occurred while communicating with the server. This problem may only
be temporary. Check that you have correctly specified your OpenShift server
'https://broker.example.com/broker/rest/domain/domain-name/applications'.
```

However, when verifying **MCollective** connectivity, **oo-mco** commands on the broker host may continue to find the rebooted node host without any issues:

```
# oo-mco ping

node.example.com                          time=170.19 ms

---- ping statistics ----
1 replies max: 170.19 min: 170.19 avg: 170.19
```

The application creation failure can be due to a communication issue between **ActiveMQ** on the broker host (or dedicated **ActiveMQ** broker, if applicable) and **MCollective** on the rebooted node host. To resolve this issue, restart the **activemq** service:

```
# service activemq restart
```

Note

See https://bugzilla.redhat.com/show_bug.cgi?id=1028382 for more information on this issue and the described workaround.

Chapter 5. Debugging Problems with Specific Applications

5.1. Common Resources

If you are having problems with an application gear, look in the **/etc/passwd** file for information unique to that particular gear. You will see an account for the gear, represented with the gear's UUID. This file also provides the path to the login shell for the application's gear. The following sample screen output shows how gears are represented in the **/etc/passwd** file.

```
........
haproxy:x:188:188::/var/lib/haproxy:/sbin/nologin
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash
mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash
jenkins:x:498:498:Jenkins Continuous Build
server:/var/lib/jenkins:/bin/false
def4330dff68444b96846dd225a0a617:x:500:500:OpenShift
guest:/var/lib/openshift/def4330dff68444b96846dd225a0a617:/usr/bin/oo-trap-
user
c9279521cffd4a5ba1118f1b6ac2d6d6:x:500:500:OpenShift
guest:/var/lib/openshift/def4330dff68444b96846dd225a0a617:/usr/bin/oo-trap-
user
e16a4a4c2c1144c3815f19ba36ea9d32:x:500:500:OpenShift
guest:/var/lib/openshift/def4330dff68444b96846dd225a0a617:/usr/bin/oo-trap-
user
```

Below is a list of directories and log files to help you diagnose problems with specific applications:

- The **/var/lib/openshift/*gear_UUID*** directory on the node host is the home directory for each application gear. Check the SELinux contexts.

- The **/var/lib/openshift/.httpd.d/*gear_UUID**** directory on the node host is the operations directory for each application gear. It contains the **httpd** configuration for that particular application gear.

- The **/var/log** directory on the node host contains the **ruby193-mcollective.log** file.

- Searching the **/var/log/openshift** directory on the node host for the gear's user UUID using **grep** could help you find problems with application gears that generate error messages.

- The **/var/log/openshift/user_action.log** file on the broker host contains logs of user actions.

5.2. Rails Applications

When debugging Rails applications, do not run the Rails console as **root** as some OpenShift API calls are cached under **/var/www/openshift/broker/tmp/cache** and are owned by the user who runs the console. When the cache expires, the broker attempts to invalidate the cache. Since the broker runs as the **apache** user it is unable to clear the root-owned files and returns 500 errors.

To avoid this problem, run the Rails console as the **apache** user:

```
# su --shell=/bin/bash -l apache
$ cd /var/www/openshift/console
$ ./script/rails console production
```

Chapter 6. Technical Support

This chapter describes how to get technical support, report bugs, and participate in the development for OpenShift Enterprise.

6.1. Reporting Bugs

If you think you have experienced a bug in your OpenShift Enterprise installation, you are encouraged to contact a support representative before you file a bug report. This will help us identify whether the bug is valid so that it receives the proper attention, and it will also ensure that a workaround or a fix is issued in a timely manner.

If you believe the bug is valid, or it has been verified to be valid, use *Bugzilla* to report the bug, using OpenShift Enterprise as the product name.

6.2. Getting Help

As an administrator for a supported OpenShift Enterprise installation, you can contact Red Hat technical support for assistance. Install the **sos** RPM, and use the following command to create an archive of relevant host information to include with your support request.

```
# sosreport
```

Report the problems for review by OpenShift Enterprise engineers.

All OpenShift Enterprise users are invited and encouraged to participate in community support by watching and posting to the *OpenShift* forums. You can also join the *#openshift* IRC channel, on *irc.freenode.net*.

6.3. Participating in Development

OpenShift Origin serves as the upstream open source project for OpenShift Enterprise, and you are invited to *get involved* in the ongoing development.

If you have a bug fix, or would like to implement a feature, submit a pull request to include your change for release. The OpenShift repositories are located on GitHub, with the origin-server of main interest to OpenShift administrators. Your submissions could be incorporated into the OpenShift Enterprise product either individually as required, or with a subsequent release.

Appendix A. Revision History

Revision 2.1-1 **Wed Jun 11 2014** **Bilhar Aulakh**

BZ 1098586: Corrected `nprocs` to be `nproc` in Section 2.1.5, “Pluggable Authentication Modules”.

Revision 2.1-0 **Fri May 16 2014** **Julie Wu**

OpenShift Enterprise 2.1 release.

BZ 1063859: Updated log file locations to `/var/log/openshift/node/ruby193-mcollective.log`

BZ 1064417: Updated Section 3.7, “cgconfig Service Fails to Start” and Section 2.1.4, “Control Groups on Node Hosts”.

BZ 1070351: Updated Section 2.1.4, “Control Groups on Node Hosts”.

Revision 2.0-1 **Tue Jan 14 2014** **Brice Fallon-Freeman**

OpenShift Enterprise 2.0.1 release.

BZ 1044676: Updated log file locations in Section 2.2, “Configuration and Log Files for OpenShift Components”.

Revision 2.0-0 **Tue Dec 10 2013** **Bilhar Aulakh**

OpenShift Enterprise 2.0 release.

Added "Communication Issue after Node Host Reboot" section.