Red Hat Mobile Application Platform 4.4

Installing RHMAP

For Red Hat Mobile Application Platform 4.4

Last Updated: 2018-10-08

Red Hat Mobile Application Platform 4.4 Installing RHMAP

For Red Hat Mobile Application Platform 4.4

Legal Notice

Abstract

This document provides an overview of the installation procedure for a complete RHMAP instance.

Table of Contents

CHAPTER 1. OVERVIEW

Depending on your RHMAP subscription, you must either:

- Install only the MBaaS on your infrastructure, with the Core managed and hosted by Red Hat.
- Install both the MBaaS and the Core on your infrastructure.

Installation of an RHMAP cluster consists of the following steps:

- Preparing Infrastructure for Installation - Install Red Hat Enterprise Linux, register with Red Hat Subscription Manager, install OpenShift Container Platform on each node of the cluster. Install the RHMAP RPMs.
- Playbook pre-req checks - Verify the environment before installation.
- Docker Installation - Optional installation method, using Docker image
- Provisioning rhmap core, or Section 5.3.1, “Accessing the Hosted Core” - depending on your subscription.
- Provisioning rhmap mbaas - you can install one or more MBaaSes.
- Finishing the installation — set up email configuration, logging, or connecting the Build Farm.

CHAPTER 2. PREPARING INFRASTRUCTURE FOR INSTALLATION

1. Determine which type of installation you require:

Installation Type	Replicaset	Core	MBaaS	Purpose
Single core and 1-node mbaas	no	1	1-node	Proof of Concept (POC) - all data and code stored on your infrastructure.
Standalone 1-node mbaas	no	Hosted	1-node	POC - Some data and code stored in a shared environment. Application data is stored on your infrastructure.
Single core with a 3-node MbaaS	yes	1	3-node	Production ready - All data and code stored on your infrastructure.
3-node mbaas on its own	yes	Hosted	3-node	Production ready - Some data and code stored in a shared environment. Application data is stored on your infrastructure.

2. Make sure your infrastructure satisfies hardware requirements.
   Use the Red Hat Mobile Application Platform 4.x sizing tool to determine how many nodes with how many processors, and how much RAM and storage are required to run RHMAP. Alternatively, see the Infrastructure Sizing Considerations for Installation of RHMAP MBaaS for a full reference of all configurations.

3. Install Red Hat Enterprise Linux (RHEL).

NOTE

See the OpenShift Container Platform Tested Integrations site to determine which version of RHEL to install.

   Install RHEL on each machine that will serve as a node in the OpenShift cluster for the Core or MBaaS. For more information, see the RHEL Installation Guide.

4. Register all cluster nodes using Red Hat Subscription Manager (RHSM) and attach the nodes to the RHMAP subscription.
   For each node in the cluster:

   a. Register the node with RHSM.
      Replace **<username>** and **<password>** with the user name and password for your Red Hat account.

```
sudo subscription-manager register --username=<username> --
password=<password>
```

The output is similar to the following:

```
Registering to: subscription.rhn.redhat.com:443/subscription
The system has been registered with ID: abcdef12-3456-7890-1234-
56789012abcd
```

b. List the available subscriptions.

```
sudo subscription-manager list --available
```

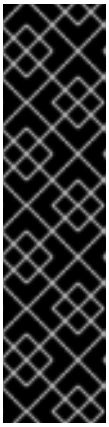
c. Find the pool ID for an RHMAP subscription and attach that pool. The pool ID is listed with the product subscription information.

```
sudo subscription-manager attach --pool=<pool_id>
```

The output is similar to the following:

```
Successfully attached a subscription for: Red Hat Mobile
Application Platform
```

5. Install OpenShift version 3.3, 3.4 or 3.5.
See the Installation and Configuration guide for detailed installation procedures.



IMPORTANT

Consider the following points when performing the installation:

- One of the options documented for OpenShift installation is to use the remaining free space from the volume group where your root file system is located. Red Hat recommends that you do not choose this option, that is, you use an existing, specified volume group, or an additional block device.
- In the OpenShift *Installation and Configuration* guide, skip steps that you have already performed as part of this procedure, for example, steps 1, 2, 3 and 4 in section *2.3.3. Host Registration*, which describe the registration process.

Use the Red Hat Mobile Application Platform 4.x sizing tool or the Infrastructure Sizing Considerations for Installation of RHMAP MBaaS document to determine how many nodes to configure in your OpenShift cluster.

2.1. INSTALL THE RHMAP OPENSHIFT TEMPLATES

Get the templates by installing the RPM package **`rhmap-fh-openshift-templates`**:

1. Update subscription-manager information

```
subscription-manager refresh
```

2. Enable the *RHMAP 4.4 RPMs* repository in RHEL.

```
subscription-manager repos --enable=rhel-7-server-rhmap-4.4-rpms
```

3. Install **rhmap-fh-openshift-templates**

```
yum install rhmap-fh-openshift-templates
```

The Ansible scripts to install RHMAP are installed into **/opt/rhmap/4.4/rhmap-installer** directory.

The following templates are installed into the **/opt/rhmap/4.4/templates/core** directory:

- **fh-core-backend.json**
- **fh-core-frontend.json**
- **fh-core-infra.json**
- **fh-core-monitoring.json**
- **fh-nginx-proxy-template.json**
- **gitlab-migrate.json**

2.2. CONFIGURE ANSIBLE FOR INSTALLING RHMAP COMPONENTS.

To install Ansible version 2.2, see the Ansible Installation Guide.

Playbooks are Ansible's configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. The playbooks required to install RHMAP are included in the RPM you download. See http://docs.ansible.com/ansible/playbooks.html for more information.

Ansible requires ssh access to the nodes in your OpenShift cluster to perform the installation. To enable this, enter the nodes into an Ansible inventory file.

2.2.1. Setting Up an Inventory File.

The RPM file include several example inventory files that can be referenced as guidelines for your installation The inventory file acts as a configuration for your installation. See http://docs.ansible.com/ansible/intro_inventory.html for more information.

For a multi-node installation, consider the following template:

```
/opt/rhmap/4.4/rhmap-installer/inventories/templates/multi-node-example
```

There are a number of parameters that can be overridden. These include settings such as that required for an outbound HTTP Proxy, the OpenShift username and password to use, and the list of nodes you intend to use for RHMAP Core and MBaaS installation. These parameters represent the global configuration options when installing RHMAP. There are more specific parameters in the Core playbooks and MBaaS playbooks that are covered under the installation guides. Below is a list of the parameters you can set in this inventory file when installing RHMAP, their uses and their defaults.

Variable	Description	Required

Variable	Description	Required
ansible_ssh_user	The SSH user Ansible uses to install RHMAP. This user must allow SSH-based authentication without requiring a password	true
ansible_sudo	Allows Ansible to escalate privileges when required. For example when checking the required PVs exist.	true
target	The type of OpenShift are we targeting. The only value that is supported at this moment is **`enterprise`**	true
cluster_hostname	Cluster hostname. The base route for your OpenShift router.	true
domain_name	Customer subdomain name. For example **`rhmap`** which will become the base domain to access RHMAP: (rhmap.my.example.com)	true
oc_user	OpenShift User that runs commands	true
oc_password	OpenShift User password / token	true
kubeconfig	The path to the config file containing the client certificates for the **`system:admin`** user. Default value is **`/etc/origin/master/admin.kubeconfig`**	true
proxy_host	HTTP_PROXY_HOST value	false
proxy_port	HTTP_PROXY_PORT value	false
proxy_user	HTTP_PROXY_USER value (only needed if authentication is required, otherwise leave it blank)	false
proxy_pass	HTTP_PROXY_PASS value (only needed if authentication is required, otherwise leave it blank)	false

Variable	Description	Required
proxy_url	The syntax is: **http://<proxy-host>:<proxy-port>** or **http://<proxy_user>:<proxy_pass>@<proxy-host>:<proxy-port>**	false
url_to_check	URL to test prerequisite of outbound HTTP connection. Must be whitelisted if you use a HTTP Proxy server.	true
gluster_storage	This must be set to **true** of using Gluster FS for persistent storage. Default value is **false**.	false
gluster_metadata_name	The name which defines the Gluster cluster in the Persistent Volume definition.	false
gluster_endpoints	A comma separated list of IP values. Must be the actual IP addresses of a Gluster server, not FQDNs. For example, ["10.10.0.55","10.10.0.56"]	false
[master]	Note this is a host group, requires access to one or more OpenShift master nodes, for example: **my-domain-master1.example.com**.	true
[mbaas]	Note this is a host group. Each of the nodes you want to use to install the MBaaS, each node labeled as per the prerequisites, labelling is only required for the MBaaS. For example: **my-domain-mbaas1.example.com** **my-domain-mbaas2.example.com** **my-domain-mbaas3.example.com**	true

Variable	Description	Required
[core]	Note this is a host group. Each of the nodes you want to use to install the Core. **my-domain-node1.example.com** **my-domain-node2.example.com** **my-domain-node3.example.com**	true

2.3. SEEDING THE NODES WITH THE RHMAP IMAGES (OPTIONAL)

You can optionally download images in advance of the installation. Red Hat recommends downloading the images to improve the robustness of the installation process, especially on low speed networks.

Images can be seeded by running the **seed-images.yml** Playbook after setting the **project_type** variable. Valid values for **project_type** are **core** and **mbaas**. Pass the RHMAP version using the **rhmap_version** variable. An example command is shown below:

```
ansible-playbook -i my-inventory-file playbooks/seed-images.yml -e
"project_type=core" -e "rhmap_version=4.4"
```

CHAPTER 3. RHMAP INSTALLER PRE-REQUISITE CHECKS

3.1. COMMON PREREQUISITE CHECKS

Description	Parameter Name	Default Value	Fail/warning/recommended
Outbound Internet Connection	url_to_check	https://www.npmjs.com	fail only in strict_mode

3.2. CHECKS ON RHMAP CORE

Description	Parameter Name	Default Value	Fail/warning/recommended
Min number of CPUs	**min_required_vCPUS**	4	fail only in strict_mode
Min system memory per node (in MB)	**required_mem_mb_threshold**	7000	fail only in strict_mode
Min total free memory of all nodes (in KB)	**warning_kb_value**	4000000	warning
Number of PVs with 50 GB storage	**required_50_pv**	0	warning
Number of PVs with 25 GB storage	**required_25_pv**	2	warning
Number of PVs with 5 GB storage	**required_5_pv**	2	warning
Number of PVs with 1 GB storage	**required_1_pv**	1	warning

3.3. CHECKS ON 1 NODE MBAAS

Description	Parameter name	Default value	Fail/warning/recommended
Min number of CPUs	**min_required_vCPUS**	2	fail only in strict_mode
Min system memory per node (in MB)	**required_mem_mb_threshold**	7000	fail only in strict_mode

Description	Parameter name	Default value	Fail/warning/recommended
Min total free memory of all nodes (in KB)	`warning_kb_value`	4000000	warning
Number of PVs with 50 GB storage	`required_50_pv`	0	warning
Number of PVs with 25 GB storage	`required_25_pv`	1	warning
Number of PVs with 5 GB storage	`required_5_pv`	0	warning
Number of PVs with 1 GB storage	`required_1_pv`	1	warning

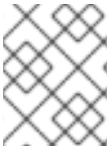
3.4. CHECKS ON 3 NODE MBAAS

Description	Parameter name	Default value	Fail/warning/recommended
Min number of CPUs	`min_required_vCPUS`	2	fail only in strict_mode
Min system memory per node (in MB)	`required_mem_mb_threshold`	7000	fail only in strict_mode
Min total free memory of all nodes (in KB)	`warning_kb_value`	4000000	warning
Number of PVs with 50 GB storage	`required_50_pv`	3	warning
Number of PVs with 25 GB storage	`required_25_pv`	0	warning
Number of PVs with 5 GB storage	`required_5_pv`	0	warning
Number of PVs with 1 GB storage	`required_1_pv`	1	warning

CHAPTER 4. USING DOCKER TO INSTALL RHMAP (OPTIONAL)

An alternative method for installation is to use the RHMAP installer docker image, which includes Ansible and the templates, so you do not need to install Ansible. To use the RHMAP installer docker image:

```
docker pull rhmap44/installer
```



NOTE

You still need to create an inventory file as described in Section 2.2.1, “Setting Up an Inventory File.”.

The following example shows how you to run the image with an interactive terminal and mounting the directory that contains the inventory file:

```
docker run -it \
       -v ~/.ssh/id_rsa:/opt/app-root/src/.ssh/id_rsa:Z \
       -v ${HOME}/Desktop/rhmap-installer/inventories:/opt/app-
root/src/inventories \
       -e ANSIBLE_PRIVATE_KEY_FILE=/opt/app-root/src/.ssh/id_rsa \
    rhmap44/installer bash
```

CHAPTER 5. PROVISIONING AN RHMAP 4.X CORE IN OPENSHIFT CONTAINER PLATFORM

Depending on your RHMAP subscription, you must either:

- Install the RHMAP Core on your own infrastructure.
- Access an instance of RHMAP Core hosted and managed by Red Hat.

5.1. PREREQUISITES

This guide assumes several prerequisites are met before the installation:

- All nodes in the cluster must be registered with the Red Hat Subscription Manager. See Chapter 2, *Preparing Infrastructure for Installation* for detailed steps.
- RHMAP RPMs are installed as described in Section 2.1, “Install the RHMAP OpenShift Templates”.
- Many Core components require direct outbound internet access to operate, make sure that all nodes have outbound internet access before installation. If you use a proxy for outbound internet access, note the proxy IP address and port, you will require both for configuration during the installation.
- Ansible version 2.2 is installed on a management node which has SSH access to the OpenShift cluster. See Section 2.2, “Configure Ansible for installing RHMAP components.” for more information.
- An existing OpenShift Container Platform installation, version 3.3, 3.4 or 3.5.
- A wildcard DNS entry must be configured for the OpenShift router IP address.
- A trusted wildcard certificate must be configured for the OpenShift router. See Using Wildcard Certificates in OpenShift documentation.
- Administrative access to the OpenShift cluster via the **`oc`** cli tool. This user must be able to:
  - Create a **`Project`**, and any resource typically found in a **`Project`** (e.g. **`DeploymentConfig`**, **`Service`**, **`Route`**)
  - Edit a **`Namespace`** definition
  - Add a **`Role`** to a **`User`**
  - Manage Nodes, specifically **`labels`**
- The rhmap-installer will run a number of pre-req checks which must pass before proceeding with the installation. See RHMAP Installer Pre-Requisite Checks for details.

For information on installation and management of an OpenShift cluster and its users, see the official OpenShift documentation.

5.2. INSTALLATION

The installation of a Core in OpenShift Container Platform results in all Core components running in Replication Controller backed Pods, with Persistent Volumes for Core data.

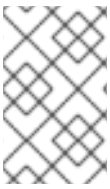
The installation consists of several phases. Before the installation, you must prepare your OpenShift cluster:

- Set up persistent storage — you need to create Persistent Volumes to cover the Persistent Volume requirements of the Core.
- Label the nodes — nodes can be labeled if the Core components are to run on specific nodes.

After the OpenShift cluster is properly configured:

- Install the Core
- Verify the installation

5.2.1. Setting Up Persistent Storage



NOTE

Ensure that the persistent volumes are configured according to the OpenShift documentation for configuring PersistentVolumes. If you are using NFS, see the Troubleshooting NFS Issues section for more information.

The Core requires a number of persistent volumes to exist before installation. As a minimum, make sure your OpenShift cluster has the following persistent volumes in an **`Available`** state, with at least the amount of free space listed below:

Component	Minimum recommended size (Default)
MongoDB	25Gi
Metrics Data Backup	5Gi
FH SCM	25Gi
GitLab Shell	5Gi
MySQL	5Gi
Nagios	1Gi

To change the default storage requirements of a component:

1. Update the persistent volume claims in the Core OpenShift templates as described in the Persistent Volume documentation.
   The following example JSON object definition shows how to create a 25GB persistent volume with **`ReadWriteOnce`** access mode:

   ```
   {
   ```

```
  "kind": "PersistentVolume",
  "apiVersion": "v1",
  "metadata": {
    "name": "examplePV"
  },
  "spec": {
    "capacity": {
      "storage": "25Gi"
    },
    "accessModes": [
      "ReadWriteOnce"
    ],
    "persistentVolumeReclaimPolicy": "Recycle",
    "nfs": {
      "path": "/path/to/examplePV",
      "server": "172.17.0.2"
    }
  }
}
```

NOTE

For more information on the types of Access Modes read the Persistent Volume Access Modes documentation.

2. Review the persistent volume reclaim policy as described in the Persistent Volume Reclaim Policy documentation to decide which policy suits your requirements. This policy affects how your data is handled if a persistent volume claim is removed.

5.2.1.1. Root Squash Recommendations

Both GlusterFS and NFS have a root squash option available, however they do not function in the same manner, here are some further details regarding this setting:

5.2.1.1.1. GlusterFS

Enabling root-squash prevents remote super-users from having super-user privileges on the storage system.

Recommended setting: Off

5.2.1.1.2. NFS

root_squash prevents remote super-users from changing other user's files on the storage system.

Recommended setting: On

5.2.1.2. Persistent Storage Recommendations

It is recommended by OpenShift to only use **`HostPath`** storage for single node testing. Instead of **`HostPath`** storage, use another driver such as GlusterFS or NFS when installing the Core. For more information on types of persistent volume read the Types of Persistent Volumes documentation.

For detailed information on persistent volumes and how to create them, see Persistent Storage in the OpenShift Container Platform documentation.

5.2.2. Applying Node Labels

You can skip this entire labeling section if your OpenShift cluster only has a single schedulable node. In such case, all Core components will run on that single node.

Red Hat recommends you deploy the Core components to dedicated nodes, separated from other applications (such as the RHMAP MBaaS and Cloud Apps). However this deployment structure is not required.

To use an example, if you have two nodes where you would like the Core components to be deployed to, these two nodes should have a specific label e.g. **`type=core`**. You can check what **`type`** labels are applied to all nodes with the following command:

```
oc get nodes -L type
```

```
NAME          STATUS                     AGE       TYPE
ose-master    Ready,SchedulingDisabled   27d       master
infra-1       Ready                      27d       infra
infra-2       Ready                      27d       infra
app-1         Ready                      27d       compute
app-2         Ready                      27d       compute
core-1        Ready                      27d
core-2        Ready                      27d
mbaas-1       Ready                      27d       mbaas
mbaas-2       Ready                      27d       mbaas
mbaas-3       Ready                      27d       mbaas
```

To add a **`type`** label to the **`core-1`** and **`core-2`** nodes, use the following command:

```
oc label node core-1 type=core
oc label node core-2 type=core
```

5.2.3. Installing the Core Using Ansible

5.2.3.1. Setting Variables

The variables required for installation of RHMAP Core are set in **`/opt/rhmap/4.4/rhmap-installer/roles/deploy-core/defaults/main.yml`**. This file will allow you to configure the RHMAP Core project for your own environment.

5.2.3.2. Configure Monitoring Components

Setup the monitoring parameters with SMTP server details. This is required to enable email alerting via Nagios when a monitoring check fails. If you don't require email alerting or want to set it up at a later time, the sample values can be used.

```
monitoring:
  smtp_server: "localhost"
  smtp_username: "username"
```

```
  smtp_password: "password"
  smtp_from_address: "nagios@example.com"
  rhmap_admin_email: "root@localhost"
```

5.2.3.3. Configure Front End Components

- SMTP server parameters

The platform sends emails for user account activation, password recovery, form submissions, and other events. Set the following variables as appropriate for your environment:

```
frontend:
  smtp_server: "localhost"
  smtp_username: "username"
  smtp_password: "password"
  smtp_port: "25"
  smtp_auth: "false"
  smtp_tls: "false"
  email_replyto: "noreply@localhost"
```

- Git External Protocol

The default protocol is **`https`**. Change this to **`http`** if you use a self-signed certificate.

```
frontend:
  git_external_protocol: "https"
```

- Build Farm Configuration

To determine the values for the **`builder_android_serviced_host`** and **`builder_iphone_serviced_host`** variables, contact Red Hat Support asking for the RHMAP Build Farm URLs that are appropriate for your region.

```
frontend:
  builder_android_service_host: "https://androidbuild.feedhenry.com"
  builder_ios_service_host: "https://iosbuild.feedhenry.com"
```

5.2.4. Running the Playbook to deploy RHMAP Core

To deploy the Core run the following command from **`/opt/rhmap/4.4/rhmap-installer`**, referencing your own inventory file:

```
ansible-playbook -i my-inventory-file playbooks/core.yml
```

The installer will run through all the tasks required to create the RHMAP Core project. It may take some time for all the Pods to start. Once all Pods are running correctly, you should see output similar to the following:

```
NAME                    READY     STATUS      RESTARTS   AGE
fh-aaa-1-ey0kd          1/1       Running     0          3h
fh-appstore-1-ok76a     1/1       Running     0          6m
fh-messaging-1-isn9f    1/1       Running     0          3h
fh-metrics-1-cnfxm      1/1       Running     0          3h
```

```
fh-ngui-1-mosqj         1/1       Running     0          6m
fh-scm-1-c9lhd          1/1       Running     0          3h
fh-supercore-1-mqgph    1/1       Running     0          3h
gitlab-shell-1-wppga    2/2       Running     0          3h
memcached-1-vvt7c       1/1       Running     0          4h
millicore-1-pkpwv       3/3       Running     0          6m
mongodb-1-1-fnf7z       1/1       Running     0          4h
mysql-1-iskrf           1/1       Running     0          4h
nagios-1-mtg31          1/1       Running     0          5h
redis-1-wwxzw           1/1       Running     0          4h
ups-1-mdnjt             1/1       Running     0          4m
```

Once all Pods are running the Ansible installer will run a Nagios checks against the RHMAP Core project to ensure it is healthy.

You can also access and view the Nagios dashboard at this point. The status of these checks can be useful if something has gone wrong during installation and needs troubleshooting.

To access Nagios, follow the Accessing the Nagios Dashboard section in the Operations Guide.

See the Troubleshooting guide if any Pods are not in the correct state or the installation has failed prior to this.

5.2.5. Verifying The Installation

1. Log in to the Studio
   To retrieve the **`URL`** for the Core Studio, use the following command:

   ```
   oc get route rhmap --template "https://{{.spec.host}}"
   ```

   The Admin username and password are set in the **`millicore DeploymentConfig`**. To view them use this command:

   ```
   oc env dc/millicore --list| grep FH_ADMIN
   ```

   ```
   FH_ADMIN_USER_PASSWORD=password
   FH_ADMIN_USER_NAME=rhmap-admin@example.com
   ```

   See the Troubleshooting guide if you are unable to login to the Studio.

5.3. POST-INSTALLATION STEPS

- Adjusting System Resource Usage of the Core - we **strongly recommend** that you adjust the system resource usage of Core components as appropriate for your production environment.
- Optional: Set up centralized logging - see the Enabling Centralized Logging section for imformation about how to deploy a centralized logging solution based on ElasticSearch, Fluentd, and Kibana.

5.3.1. Accessing the Hosted Core

After Red Hat receives the purchase order for an RHMAP 4.4 subscription with a hosted Core, a member of the sales team internally requests a new RHMAP domain for accessing an instance of the RHMAP Core hosted by Red Hat.

Once the domain is created, a representative of the Red Hat Customer Enablement team will instruct you how to install the MBaaS.

The following steps for getting access to RHMAP Core are performed by a representative of the Red Hat Customer Enablement team:

1. Creating a domain.
   The domain, such as **`customername.redhatmobile.com`**, hosts the RHMAP Core for a single customer.

2. Creating an administrator account.
   An RHMAP administrator account is created in the domain, and the customer's technical contact receives an activation e-mail which allows access to the domain using the new account.

CHAPTER 6. PROVISIONING AN RHMAP 4.X MBAAS IN OPENSHIFT CONTAINER PLATFORM

6.1. OVERVIEW

An OpenShift Container Platform cluster can serve as an MBaaS target and host your Cloud Apps and Cloud Services. This guide provides detailed steps to deploy the RHMAP 4.x MBaaS on an OpenShift Container Platform cluster.

6.2. PREREQUISITES

This guide assumes several prerequisites are met before the installation:

- Ansible version 2.2 is installed on a management node which has SSH access to the OpenShift cluster. See Section 2.2, “Configure Ansible for installing RHMAP components.” for more information.
- All nodes in the cluster must be registered with the Red Hat Subscription Manager. See Chapter 2, *Preparing Infrastructure for Installation* for detailed steps.
- The MBaaS requires outbound internet access to perform npm installations, make sure that all relevant nodes have outbound internet access before installation.
- An existing OpenShift Container Platform installation, version 3.3, 3.4 or 3.5.
- The OpenShift Container Platform master and router must be accessible from the RHMAP Core.
- A wildcard DNS entry must be configured for the OpenShift Container Platform router IP address.
- A trusted wildcard certificate must be configured for the OpenShift Container Platform router. See Using Wildcard Certificates in OpenShift Container Platform documentation.
- Image streams and images in the **`openshift`** namespace must be updated to the latest version. Refer to sections Updating the Default Image Streams and Templates and Importing the Latest Images in the OpenShift Container Platform Installation and Configuration guide.
- You must have administrative access to the OpenShift cluster using the **oc** CLI tool, enabling you to:
  - Create a *project*, and any resource typically found in a project (for example, *deployment configuration*, *service*, *route*).
  - Edit a *namespace* definition.
  - Create a *security context constraint*.
  - Manage nodes, specifically *labels*.
- The rhmap-installer will run a number of pre-req checks which must pass before proceeding with the installation. See RHMAP Installer Pre-Requisite Checks for details.

For information on installation and management of an OpenShift Container Platform cluster and its users, see the official OpenShift documentation.

6.3. INSTALLATION

The installation of an three-node MBaaS in OpenShift Container Platform results in a resilient three-node cluster:

- MBaaS components are spread across all three nodes.
- MongoDB replica set is spread over three nodes.
- MongoDB data is backed by persistent volumes.
- A Nagios service with health checks and alerts is set up for all MBaaS components.

The installation consists of several phases. Before the installation, you must prepare your OpenShift Container Platform cluster:

- Set up persistent storage - you need to create Persistent Volumes with specific parameters in OpenShift Container Platform.
- Label the nodes - nodes need to be labeled in a specific way, to match the node selectors expected by the OpenShift Container Platform template of the MBaaS.
- Network Configuration - configuring the SDN network plugin used in OpenShift Container Platform so that Cloud Apps can communicate with MongoDB in the MBaaS.

After the OpenShift Container Platform cluster is properly configured:

- Install the MBaaS from a template
- Verify the installation

6.3.1. Before The Installation

The installation procedure poses certain requirements on your OpenShift Container Platform cluster in order to guarantee fault tolerance and stability.

6.3.1.1. Network Configuration

Cloud Apps in an MBaaS communicate directly with a MongoDB replica set. In order for this to work, the OpenShift Container Platform SDN must be configured to use the **`ovs-subnet`** SDN plugin. For more detailed information on configuring this, see Migrating Between SDN Plug-ins in the OpenShift documentation.

6.3.1.1.1. Making Project Networks Global

If you cannot use the **`ovs-subnet`** SDN plugin, you must make the network of the MBaaS project global after installation. For example, if you use the **`ovs-multitenant`** SDN plugin, projects must be configured as global. The following command is an example of how to make a project global:

```
oadm pod-network make-projects-global live-mbaas
```

To determine if projects are global, use the following command:

```
oc get netnamespaces
```

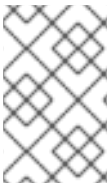
In the output, any projects that are configured global have namespaces with a value of "0"

NOTE

If a project network is configured as global, you cannot reconfigure it to reduce network accessibility.

For further information on how to make projects global, see Making Project Networks Global in the OpenShift Container Platform documentation.

6.3.1.2. Persistent Storage Setup



NOTE

Ensure that the persistent volumes are configured according to the OpenShift documentation for configuring PersistentVolumes. If you are using NFS, see the Troubleshooting NFS Issues section for more information.

Some components of the MBaaS require persistent storage. For example, MongoDB for storing databases, and Nagios for storing historical monitoring data.

As a minimum, make sure your OpenShift Container Platform cluster has the following persistent volumes in an **`Available`** state, with at least the amount of free space listed below:

- Three **50 GB** persistent volumes, one for each MongoDB replica
- One **1 GB** persistent volume for Nagios

For detailed information on PersistentVolumes and how to create them, see Persistent Storage in the OpenShift Container Platform documentation.

6.3.1.3. Apply Node Labels for MBaaS

By applying labels to OpenShift Container Platform nodes, you can control which nodes the MBaaS components, MongoDB replicas and Cloud Apps will be deployed to.

This section describes the considerations for:

- Section 6.3.1.3.1, “Labelling for MBaaS components”
- Section 6.3.1.3.2, “Labelling for MongoDB replicas”

Cloud apps get deployed to nodes labeled with the default **`nodeSelector`**, which is usually set to **`type=compute`** (defined in the OpenShift Container Platform master configuration).

You can skip this entire labeling section if your OpenShift Container Platform cluster only has a single schedulable node. In such case, all MBaaS components, MongoDB replicas, and Cloud Apps will necessarily run on that single node.

6.3.1.3.1. Labelling for MBaaS components

Red Hat recommends that MBaaS components are deployed to dedicated nodes and that these nodes are separated from other applications, for example, RHMAP Cloud Apps.

Refer to Infrastructure Sizing Considerations for Installation of RHMAP MBaaS for the recommended number of MBaaS nodes and Cloud App nodes for your configuration.

For example, if you have 12 nodes, the recommendation is:

- Dedicate three nodes to MBaaS and MongoDB.
- Dedicate three nodes to Cloud Apps.

To achieve this, apply a label, such as **`type=mbaas`** to the three dedicated MBaaS nodes.

```
oc label node mbaas-1 type=mbaas
oc label node mbaas-2 type=mbaas
oc label node mbaas-3 type=mbaas
```

Then, when creating the MBaaS project, as described later in Section 6.3.2, “Installing the MBaaS”, set this label as the **`nodeSelector`**.

You can check what **`type`** labels are applied to all nodes with the following command:

```
oc get nodes -L type
```

```
NAME          STATUS                        AGE       TYPE
ose-master    Ready,SchedulingDisabled      27d       master
infra-1       Ready                         27d       infra
infra-2       Ready                         27d       infra
app-1         Ready                         27d       compute
app-2         Ready                         27d       compute
app-3         Ready                         27d       compute
mbaas-1       Ready                         27d       mbaas
mbaas-2       Ready                         27d       mbaas
mbaas-3       Ready                         27d       mbaas
```

In this example, the deployment would be as follows:

- Cloud apps get deployed to the three dedicated Cloud App nodes **`app-1`**, **`app-2`**, and **`app-3`**.
- The MBaaS components get deployed to the three dedicated MBaaS nodes **`mbaas-1`**, **`mbaas-2`**, and **`mbaas-3`** (if the **`nodeSelector`** is also set on the MBaaS Project).

6.3.1.3.2. Labelling for MongoDB replicas

In the production MBaaS template, the MongoDB replicas are spread over three MBaaS nodes. If you have more than three MBaaS nodes, any three of them can host the MongoDB replicas.

To apply the required labels (assuming the three nodes are named **`mbaas-1`**, **`mbaas-2`**, and **`mbaas-3`**):

```
oc label node mbaas-1 mbaas_id=mbaas1
oc label node mbaas-2 mbaas_id=mbaas2
oc label node mbaas-3 mbaas_id=mbaas3
```

You can verify the labels were applied correctly by running this command:

```
oc get nodes -L mbaas_id
```

```
NAME          STATUS                     AGE       MBAAS_ID
10.10.0.102   Ready                      27d       <none>
10.10.0.117   Ready                      27d       <none>
10.10.0.141   Ready                      27d       <none>
10.10.0.157   Ready                      27d       mbaas3
10.10.0.19    Ready,SchedulingDisabled   27d       <none>
10.10.0.28    Ready                      27d       mbaas1
10.10.0.33    Ready                      27d       <none>
10.10.0.4     Ready                      27d       <none>
10.10.0.99    Ready                      27d       mbaas2
```

See Updating Labels on Nodes in the OpenShift Container Platform documentation for more information on how to apply labels to nodes.

6.3.1.3.2.1. Why are MongoDB replicas spread over multiple nodes?

Each MongoDB replica is scheduled to a different node to support failover.

For example, if an OpenShift Container Platform node failed, data would be completely inaccessible if all three MongoDB replicas were scheduled on this failing node. Setting a different **`nodeSelector`** for each MongoDB **`DeploymentConfig`**, and having a corresponding OpenShift Container Platform node in the cluster matching this label will ensure the MongoDB Pods get scheduled to different nodes.

In the production MBaaS template, there is a different **`nodeSelector`** for each MongoDB **`DeploymentConfig`**:

- **`mbaas_id=mbaas1`** for **`mongodb-1`**
- **`mbaas_id=mbaas2`** for **`mongodb-2`**
- **`mbaas_id=mbaas3`** for **`mongodb-3`**

6.3.2. Installing the MBaaS

6.3.2.1. Setting Variables

The variables required for installation of RHMAP MBaaS are set in the following file:

```
/opt/rhmap/4.4/rhmap-installer/roles/deploy-mbaas/defaults/main.yml
```

Set up the monitoring parameters with SMTP server details, which are required to enable email alerting from Nagios. If you do not require email alerting or want to set it up at a later time, the sample values can be used.

```
monitoring:
  smtp_server: "localhost"
  smtp_username: "username"
  smtp_password: "password"
  smtp_from_address: "nagios@example.com"
  rhmap_admin_email: "root@localhost"
```

6.3.2.2. Run the Playbook

To provision a 1-node MBaaS, enter:

```
ansible-playbook -i my-inventory-file playbooks/1-node-mbaas.yml
```

To provision a 3-node MBaaS, enter:

```
ansible-playbook -i my-inventory-file playbooks/3-node-mbaas.yml
```

6.3.3. Verifying The Installation

1. Ping the health endpoint.
   If all services are created, all Pods are running, and the route is exposed, the MBaaS health endpoint can be queried as follows:

   ```
   curl `oc get route mbaas --template "
   {{.spec.host}}"`/sys/info/health
   ```

   The endpoint responds with health information about the various MBaaS components and their dependencies. If there are no errors reported, the MBaaS is ready to be configured for use in the Studio. Successful output will resemble the following:

   ```
   {
     "status": "ok",
     "summary": "No issues to report. All tests passed without error",
     "details": [
       {
         "description": "Check Mongodb connection",
         "test_status": "ok",
         "result": {
           "id": "mongodb",
           "status": "OK",
           "error": null
         },
         "runtime": 33
       },
       {
         "description": "Check fh-messaging running",
         "test_status": "ok",
         "result": {
           "id": "fh-messaging",
           "status": "OK",
           "error": null
         },
         "runtime": 64
       },
       {
         "description": "Check fh-metrics running",
         "test_status": "ok",
         "result": {
           "id": "fh-metrics",
           "status": "OK",
           "error": null
         },
         "runtime": 201
   ```

```
    },
    {
      "description": "Check fh-statsd running",
      "test_status": "ok",
      "result": {
        "id": "fh-statsd",
        "status": "OK",
        "error": null
      },
      "runtime": 7020
    }
  ]
}
```


2. Verify that all Nagios checks are passing.
   Log in to the Nagios dashboard of the MBaaS by following the steps in the Accessing the Nagios Dashboard section in the Operations Guide.

   After logging in to the Nagios Dashboard, all checks under the left-hand-side **Services** menu should be indicated as **OK**.

See the Troubleshooting guide if any of the checks are not in an **OK** state.

After verifying that the MBaaS is installed correctly, you must create an MBaaS target for the new MBaaS in the Studio.

6.4. CREATING AN MBAAS TARGET

1. In the Studio, navigate to the *Admin > MBaaS Targets* section. Click *Create MBaaS Target*.

2. Enter the following information

   - **MBaaS Id** - a unique ID for the MBaaS, for example: **`live-mbaas`**.

   - **OpenShift Master URL** - the URL of the OpenShift Container Platform master, for example, **`https://master.openshift.example.com:8443`**.

   - **OpenShift Router DNS** - a wildcard DNS entry of the OpenShift Container Platform router, for example, **`*.cloudapps.example.com`**.

   - **MBaaS Service Key**
     Equivalent to the value of the **`FHMBAAS_KEY`** environment variable, which is automatically generated during installation. To find out this value, run the following command:

     ```
     oc env dc/fh-mbaas --list | grep FHMBAAS_KEY
     ```

     Alternatively, you can find the value in the OpenShift Container Platform Console, in the *Deployment config* of the **`fh-mbaas`**, in the *Environment* section.

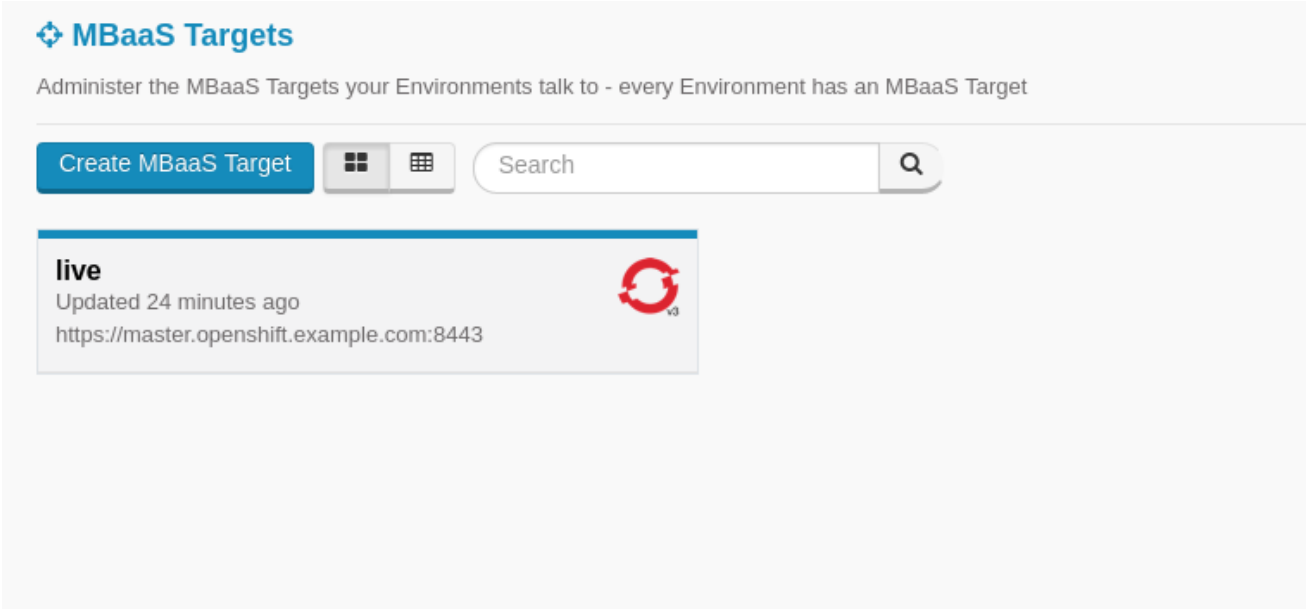
   - **MBaaS URL**
     A URL of the route exposed for the **`fh-mbaas-service`**, including the *https* protocol prefix. This can be retrieved from the OpenShift Container Platform web console, or by running the following command:

```
echo "https://"$(oc get route/mbaas -o template --template
{{.spec.host}})
```

- **MBaaS Project URL** - (Optional) URL where the OpenShift Container Platform MBaaS project is available e.g. https://mbaas-mymbaas.openshift.example.com:8443/console/project/my-mbaas/overview.
- **Nagios URL** - (Optional) Exposed route where Nagios is running in OpenShift Container Platform e.g. https://nagios-my-mbaas.openshift.example.com.

3. Click *Save MBaaS* and you will be directed to the MBaaS Status screen. The status should be reported back in less than a minute.

Once the process of creating the MBaaS has successfully completed, you can see the new MBaaS in the list of MBaaS targets.



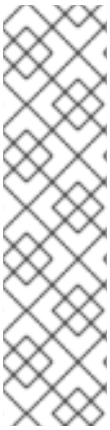
In your OpenShift Container Platform account, you can see the MBaaS represented by a project.

6.5. AFTER INSTALLATION

- Create an Environment - you must create at least one environment for the MBaaS to be usable by Cloud Apps and Cloud Services

- Adjusting System Resource Usage of the MBaaS and Cloud Apps - Red Hat recommends that you adjust the system resource usage of MBaaS components as appropriate for your production environment

CHAPTER 7. PROVISIONING AN RHMAP BUILD FARM



NOTE

This feature is a technical preview. The hosted Build Farm remains the default configuration. Red Hat recommends that you:

1. Install RHMAP and verify your installation with the hosted {BuildFarmName}.
2. Install {BuildFarmName} without changing the {BuildFarmName} configuration.
3. After verifying the {BuildFarmName} installation, run the {BuildFarmName} installation again with the option to change the configuration to point to the self-managed {BuildFarmName}.

For more information, see:

- Mobile Developer Guide for information required by mobile developers to build apps.
- Administrator's Guide for information on administering the {BuildFarmName}, including specific information about Configuring a macOS Server for Customization.

7.1. OVERVIEW OF BUILD FARM INSTALLATION

By default, RHMAP is configured to use the build farm hosted by Red Hat to create binary files for iOS and Android Client Apps as described in Section 5.2.3.3, "Configure Front End Components". You can provision a self-managed build farm, this guide provides detailed steps to deploy the RHMAP build farm on an OpenShift Container Platform and a MacOS server. After provisioning, the build farm consists of:

- An OpenShift project which includes:
  - A Jenkins server pod which runs all the build jobs for creating Client App binaries.
  - One or more pods with all the required tools to build Android apk files, dynamically provisioned as required to create the Android binaries.
  - A nagios server for monitoring the {BuildFarmName} components
- An optional macOS Server with all the required tools to build iOS binaries

The {BuildFarmName} installation is completed by running an Ansible playbook which installs the components, including the tools required to build iOS apps on macOS servers.

NOTE

The Client App templates included in this release contain Jenkins configuration files (`Jenkinsfile`) that enable you to use the self-managed build farm.

7.2. PREREQUISITES FOR {BUILDFARMNAME}

This guide assumes several prerequisites are met before the installation:

- A working RHMAP 4.6 Self-Managed Core and MBaaS installation

- An optional macOS server if you need to build iOS Apps. You also require a user with sudo permissions and SSH access. See macOS Prerequisites for further details.
- Ansible version 2.2 is installed on a management node which has SSH access to the OpenShift cluster nodes. See Section 2.2, “Configure Ansible for installing RHMAP components.” for more information.
- Java is required for some configuration of Jenkins during the installation. The Ansible playbook installs Java if required.
- All nodes in the cluster must be registered with the Red Hat Subscription Manager. See Chapter 2, *Preparing Infrastructure for Installation* for detailed steps.
- The build farm requires outbound internet access to perform npm installations, make sure that all relevant nodes have outbound internet access before installation.
- As part of the technical preview, the Ansible playbook retrieves the Build Farm container images from https://hub.docker.com/. Make sure you have access to this site before attempting installation.
- An existing OpenShift Container Platform installation, version 3.3, 3.4 or 3.5 (A mininum of 2 cores and 6G RAM and 1G per concurrent build).
- The OpenShift Container Platform master and router must be accessible from the RHMAP Core.
- A wildcard DNS entry must be configured for the OpenShift Container Platform router IP address.
- A trusted wildcard certificate must be configured for the OpenShift Container Platform router. See Using Wildcard Certificates in OpenShift Container Platform documentation.
- Image streams and images in the **`openshift`** namespace must be updated to the latest version. Refer to sections Updating the Default Image Streams and Templates and Importing the Latest Images in the OpenShift Container Platform Installation and Configuration guide.
- You must have administrative access to the OpenShift cluster using the **`oc`** CLI tool, enabling you to:
  - Create a *project*, and any resource typically found in a project (for example, *deployment configuration*, *service*, *route*).
  - Edit a *namespace* definition.
  - Create a *security context constraint*.
  - Manage nodes, specifically *labels*.

For information on installation and management of an OpenShift Container Platform cluster and its users, see the official OpenShift documentation.

- Section 7.3, “macOS Specific Prerequisites”
- Set up persistent storage - you need to create Persistent Volumes with specific parameters in OpenShift Container Platform.
- Install the build farm from a template
- Verify the installation

7.3. MACOS SPECIFIC PREREQUISITES

Red Hat recommends a single MacInCloud installation with the following specification:

- 4GB RAM
- 2 Cores
- macOS 10.12 Sierra
- 30GB minimum disk space and additional 20GB per XCode version

7.3.1. macOS and macOS Ansible Scripts Configuration

To configure your macOS server:

1. Section 7.3.1.1, “Create a SSH User”
2. Section 7.3.1.2, “Configure Remote Login”

7.3.1.1. Create a SSH User

Ansible requires SSH access to the macOS server in order to correctly provision the machine to support iOS builds. To enable this, it is required that a user with sudo permissions and SSH access is created on the target machine. An example bash script is provided to achieve this.

WARNING

The provided scripts helps with creating an OSX user. Do not use these scripts without reviewing their contents as it will allow access to your machine with a default password. Change the default **`PASSWORD`** and ensure **`USER_ID`** is not assigned.

By default, this script will create a user named **`jenkins`** with a **`UID`** of **`550`** and add the user to the the **`sudoers`** file. The users **`PASSWORD`** variable is set by default to **`Password1`**. To enable iOS builds, this user is added to the **`admin`** group.

If using the sample script (contents provided below) you should read it carefully, understand its implications and edit it as required. The script can be downloaded for convenience and run using the following commands:

```
curl -O https://raw.githubusercontent.com/aerogear/digger-jenkins/FH-
v4.6/admin/create-osx-user.sh
sudo bash create-osx-user.sh
```

```
#!/usr/bin/env bash

USERNAME="jenkins"
PASSWORD="Password1"
REAL_NAME="Jenkins Agent"
GROUP_NAME="staff"
```

```
# the first user's id is 500, second is 501 ...
# picking a big number to be on the safe side.
# You can run this one to list UIDs
#   dscl . -list /Users UniqueID
USER_ID=550

# GID 20 is `staff`
GROUP_ID=20

############### end of parameters

. /etc/rc.common
dscl . create /Users/${USERNAME}
dscl . create /Users/${USERNAME} RealName ${REAL_NAME}
dscl . passwd /Users/${USERNAME} ${PASSWORD}

dscl . create /Users/${USERNAME} UniqueID ${USER_ID}
dscl . create /Users/${USERNAME} PrimaryGroupID ${GROUP_ID}
dscl . create /Users/${USERNAME} UserShell /bin/bash
dscl . create /Users/${USERNAME} NFSHomeDirectory /Users/${USERNAME}
dseditgroup -o edit -a ${USERNAME} -t user admin
cp -R /System/Library/User\ Template/English.lproj /Users/${USERNAME}
chown -R ${USERNAME}:${GROUP_NAME} /Users/${USERNAME}

echo "${USERNAME}  ALL=(ALL:ALL) ALL" >> /etc/sudoers

echo "Done creating OSX user - you may need to restart the osx server to
apply all changes for the user ${USERNAME}"
```

IMPORTANT

Restart the server to implement the changes made above.

7.3.1.2. Configure Remote Login

You must enable the Ansible process running remotely to execute commands on the macOS server via SSH.

A sample script (contents provided below) is provided to achieve this. The **USERNAME** variable must match the user that was created in the previous step. It can be downloaded and executed by running the following:

```
curl -O https://raw.githubusercontent.com/aerogear/digger-jenkins/FH-
v4.6/admin/enable-osx-remote-login.sh
sudo bash enable-osx-remote-login.sh
```

```
#!/usr/bin/env bash

# This script helps with enabling SSH for given OSX user.
# This script is not meant to be run in an automation. Run it manually.

USERNAME="jenkins"

# com.apple.access_ssh is a special group name on OSX.
```

```
# any user part of that group can have SSH connections in.
OSX_SSH_GROUP_NAME="com.apple.access_ssh"

systemsetup -setremotelogin on
# in order to check what groups are are there:
#   dscl . list /Groups PrimaryGroupID
# create a group for limiting SSH access
dseditgroup -o create -q ${OSX_SSH_GROUP_NAME}
# add user into this group
dseditgroup -o edit -a ${USERNAME} -t user ${OSX_SSH_GROUP_NAME}
# now, following should work  ---> ssh username@localhost
```

This allows the **USERNAME** created previously to access the server via SSH.

7.3.2. Connecting Using Ansible

A number of tasks in the **provision-osx** Ansible role used in the installer require privilege escalation. This allows us to run commands as **sudo** user

The user created in the previous steps should be provided as a host variable to Ansible via **ansible_ssh_user**

Set **ansible_sudo_pass** variable or pass the **--ask-sudo-pass** flag when running the Ansible playbook to enable the installer to work with root permissions. This value should match the password set when creating the user.

7.3.3. Configuring Network access

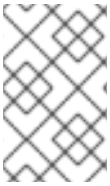
The macOS server requires a network access to the following hosts. If you are using a proxy, configure it to allow access to these hosts.

Table 7.1. Required resources

Hostname	Provides
https://github.com	**Homebrew packages** and **Cocoapods**
https://raw.githubusercontent.com	**NVM - Node Version Manager**
http://developer.apple.com	**Apple Certificates** and **Xcode**
https://npmjs.org	**NPM packages**

Access to other external internet resources is required if you install other packages as described in the Administrator's Guide.

7.4. PERSISTENT STORAGE SETUP



NOTE

Ensure that the persistent volumes are configured according to the OpenShift documentation for configuring PersistentVolumes. If you are using NFS, see the Troubleshooting NFS Issues section for more information.

Some components of the Build Farm require persistent storage. For example, Nagios for storing historical monitoring data.

As a minimum, make sure your OpenShift Container Platform cluster has the following persistent volumes in an **`Available`** state, with at least the amount of free space listed below:

- 40GB for Jenkins
- 10GB for Android-sdk
- 1GB for Nagios

For detailed information on PersistentVolumes and how to create them, see Persistent Storage in the OpenShift Container Platform documentation.

7.5. INSTALLING {BUILDFARMNAME}

1. Navigate to the **`/opt/rhmap/4.6/rhmap-installer/`**
2. Create or update the inventory file

NOTE

Red Hat recommends that you review each variable values to avoid errors during installation.

3. Run a variation of the following command and follow the instructions displayed:

```
ansible-playbook -i <inventory-file> buildfarm.yml [--skip-tags=
<exclude-option>] [-e "core_project_name=<core-project-name>"]
```

The **`-e "core_project_name=<core-project-name>"`** option allows you configure RHMAP to use the self-managed Build Farm.

The **`--skip-tags=<exclude-option>`** allows you skip tasks. For example, to install the buildfarm for Android Applications only:

```
ansible-playbook -i your-inventory-copy buildfarm.yml --skip-
tags=provision-osx -e "core_project_name=<core-project-name>"
```

In the following example, you install {BuildFarmName} for Android and iOS Applications, but do not configure RHMAP to use the self-managed {BuildFarmName}:

```
ansible-playbook -i your-inventory-copy buildfarm.yml -e
"core_project_name=<core-project-name>"
```

After this installation is complete, you can run the following command to configure RHMAP to use the self-managed {BuildFarmName}:

```
ansible-playbook -i your-inventory-copy buildfarm.yml -e
"core_project_name=<core-project-name" --tags=configure-millicore
```

1. Complete the on-screen instructions.
   There are three points during the installation when the installer stops and manual steps required:

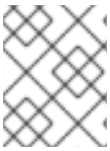
   a. Accept the Android SDK Licence.

   b. Copy the server's public ssh key to Jenkins. The server is where ansible task are running.

   c. Accept the Oracle Licence when installing Java on macOS server
   All of these steps are required for the installation to complete successfully. At each step the Ansible Installer pauses and prompts with the required actions.

During the installation the Ansible playbook performs the following tasks:

- Verify and install (if required) the minimum Java version
- Create a project in OpenShift
- Install Jenkins
- Install Android SDK to a Persistent Volume
- Configure Jenkins
- Setup macOS Server(s) (optional depending on command used)
- Configure RHMAP Core to target the Self-Managed Buildfarm
- Install Nagios
- Display URL and credentials for Jenkins and Nagios



NOTE

These tasks are idempotent. You can run them multiple times and produce the same results.

Once the installation is complete, verify the installation following the steps in Verify Installation Section

7.6. SETTING UP AN INVENTORY FILE FOR BUILD FARM

Installing {BuildFarmName} requires additional inventory file entries compared to the installation of Core and MBaaS. Use the information in this section to either edit your existing inventory file or create a new inventory file as described in the Section 2.2.1, "Setting Up an Inventory File." section.

7.6.1. General Inventory File Variables

Variable name	Description	Default value	Required
jenkins_route_protocol	Route protocol used to contact Jenkins	https	No

7.6.2. Android Inventory File Variables

Variable name	Description	Default value	Required
android_sdk_home	Location of Android SDK that is installed in the Android SDK container.	/opt/android-sdk-linux	No

7.6.3. {BuildFarmName} Configuration Inventory File Variables

Variable name	Description	Default value	Required
project_name	Name of the OpenShift project.	digger	No
concurrent_android_builds	Number of max number of concurrent Android builds that are allowed.	5	No

7.6.4. Jenkins Inventory File Variables

Variable name	Description	Default value	Required
project_name	Name of the OpenShift project.	digger	No
enable_oauth	Enable OAuth OpenShift integration. If false, a static account is initialized.	false	No
master_memory_limit	Maximum amount of memory for Jenkins master container.	3Gi	No
master_volume_capacity	Space available for data.	40Gi	No

7.6.5. Nagios Inventory File Variables

These variables are used in Nagios when sending alert emails.

Variable name	Description	Default value	Required
smtp_server	SMTP server to send alert emails.	localhost	No

Variable name	Description	Default value	Required
smtp_username	SMTP username.	username	No
smtp_password	Password of the SMTP user.	password	No
smtp_from_address	SMTP from address.	admin@example.com	No
rhmap_admin_email	Destination address of alert emails.	root@localhost	No
jenkins_user	Jenkins user associated with nagios checks	admin	No
jenkins_pass	Password for jenkins_user	password	No

7.6.6. Java Inventory File Variables

Variables below are to configure the JDK to install on the remote that runs Jenkins CLI commands.

Variable name	Description	Default value	Required
repo	Repository to install JDK.	rhel-7-server-optional-rpms	No
java_version	JDK version to install.	1.8.0	No

7.6.7. Login Inventory File Variables

Variables below are used to login to the OpenShift Cluster.

Variable name	Description	Default value	Required
oc_user	OpenShift user to login to OpenShift.		Yes
oc_password	OpenShift user password.		Yes
login_url	Url used to log in.	https://localhost:8443	No. * Except if running the playbook locally against a remote server and using **`ansible_connection=local`** Otherwise allow to default.

Variable name	Description	Default value	Required

7.6.8. Provisioning macOS Inventory File Variables

Variables below are used while provisioning an OSX node.

Variable name	Description	Default value	Required
ansible_become_pass	Sudo password for carrying out root priveledged actions on a macOS server		Yes/No if passing the value via the command line
remote_tmp_dir	What directory to use when creating some temporary files.	/tmp	No
node_versions	A list of Node versions to install.	6	No
xcode_install_version	The version of the xcode-install tool to install on the node.	2.2.1	No
gem_packages		name: public_suffix, version: 2.0.5:name: xcode-install: <xcode_install_version>	No
cocoapods_version	The version of the Cocoapods gem to install.	1.1.1	No
npm_packages	A list of global NPM packages to install. Format: **`{ name: <PACKAGE_NAME>, version: <PACKAGE_VERSION> }`**.	name: cordova, version: 7.0.1	No

Variable name	Description	Default value	Required
homebrew_packages	The packages to install using Homebrew. Format: `{ name: <PACKAGE_NAME> }`.	gpg, grep, jq	No
homebrew_version	The version of Homebrew to install (git tag).	1.3.1	No
homebrew_repo	The git repo where Homebrew resides (defaults to GitHub repo).	https://github.com/Homebrew/brew	No
homebrew_prefix	The parent directory of the directory where Homebrew resides.	/usr/local	No
homebrew_install_path	Where Homebrew will be installed.	<homebrew_prefix>/Homebrew	No
homebrew_brew_bin_path	Where `brew` will be installed.	/usr/local/bin	No
homebrew_paths		<homebrew_install_path>, <homebrew_brew_bin_path>, <homebrew_var_path>,/usr/local/Cellar,/usr/local/opt,/usr/local/share,/usr/local/etc,/usr/local/include	No
homebrew_taps	A list of taps to add.	homebrew/core, caskroom/cask	No
xcode_install_user	Apple Developer Account username. If this is not set then Xcode will not be installed.		Yes (if xcode is required)
xcode_install_password	Apple Developer Account password. If this is not set then Xcode will not be installed.		Yes (if xcode is required)

Variable name	Description	Default value	Required
xcode_install_session_token	Apple Developer Account auth cookie from **`fastlane spaceauth`** command (For accounts with 2FA enabled).		Yes (if xcode is required)
xcode_versions	A list of Xcode versions to install. These may take over 30 minutes each to install.	'8.3.3'	No
xcode_default_version	Teh default version of xcode to be used	<xcode_version>[0]	No
apple_wwdr_cert_url	Apple WWDR certificate URL. Defaults to Apple's official URL.	http://developer.apple.com/certificationauthority/AppleWWDRCA.cer	No
apple_wwdr_cert_file_name	Output file name of the downloaded file.	AppleWWDRCA.cer	No
buildfarm_node_port	The port to connect to the macOS node on.	22	No
buildfarm_node_root_dir	Path to Jenkins root folder.	/Users/jenkins	No
buildfarm_credential_id	Identifier for the Jenkins credential object.	macOS_buildfarm_cred	No
buildfarm_credential_description	Description of the Jenkins credential object.	Shared credential for the macOS nodes in the buildfarm.	No
buildfarm_node_name	Name of the slave/node in Jenkins.	macOS (<node_host_address>)	No
buildfarm_node_labels	List of labels assigned to the macOS node.	ios	No
buildfarm_user_id	Jenkins user ID.	admin	No

Variable name	Description	Default value	Required
buildfarm_node_executors	Number of executors (Jenkins configuration) on the macOS node. There is currently no build isolation with the macOS node meaning there is no guaranteed support for concurrent builds. This value should not be changed unless you are certain all apps will be built with the same signature credentials.	1	No
buildfarm_node_mode	How the macOS node should be utilised. The following options are available:	NORMAL	No. Can be set to EXCLUSIVE to set that only build jobs with labels matching this node will use this node.
buildfarm_node_description	Description of the macOS node in Jenkins.	macOS node for the buildfarm	No
project_name	The name of the {BuildFarmName} Project in OpenShift	digger	No
proxy_host	Proxy url/base hostname to be used.		No/Yes if the macOS server only has outbound internet access via proxy
proxy_port	Proxy port to be used.		No/Yes if the macOS server only has outbound internet access via proxy
proxy_device	The proxy network device to use the proxy config from the list of devices.	Ethernet	No
proxy_ctx	A list of proxies to be set.	webproxy, securewebproxy	No

Variable name	Description	Default value	Required
buildfarm_lang_env_var	Value of **LANG** environment variable to set on the macOS node. CocoaPods require this to **en_US.UTF-8**.	en_US.UTF-8	No
buildfarm_path_env_var	**$PATH** environment variable to use in the macOS node.	$PATH:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin	No
credential_private_key	Private key stored in Jenkins and used to SSH into the macOS node. If this is not set then a key pair will be generated.		No
credential_public_key	Public key of the pair. If this is not set then a key pair will be generated.		No
credential_passphrase	Passphrase of the private key. This is stored in Jenkins and used to SSH into the macOS node. If this is not set the private key will not be password protected.		No

7.7. VERIFYING THE {BUILDFARMNAME} INSTALLATION

To monitor the status of the {BuildFarmName} components, the {BuildFarmName} installer will install Nagios and several Nagios checks. At the end of the installation Ansible will trigger each of the checks and output the status of these checks. To navigate to Nagios and manually verify that all checks are passing, follow the steps below.

To verify the installation

1. Log in to Nagios using the url and credentials outputted at the end of the installation.
2. Navigate to the Services Link
3. Verify that all Nagios checks are passing.

The environment specific checks carried out by Nagios are:

- Container CPU Usage

- Container Memory Usage
- Container Resource Limits
- Pod Disk Storage

The {BuildFarmName} specific checks carried out by Nagios are:

- Availability of the Jenkins container (pod)
- Status of the network link between Jenkins and the macOS servers
- Status of the Android SDK PersistentVolumeClaim

After verifying that the {BuildFarmName} is installed correctly, you can use it to build applications as described in Using the Build Farm If you want to revert to using the hosted build farm, follow the instructions in the Using the RHMAP Hosted Build Farm section.

CHAPTER 8. POST-INSTALLATION TASKS

After installing the Core and the MBaaS, you can enable several features to access all functionality of the RHMAP cluster:

- Set up centralized logging.
  - Enabling Centralized Logging
- Set up monitoring.
  - Monitoring RHMAP with Cockpit
  - Monitoring RHMAP with Nagios

Core

- Set up email configuration for the Core.
  - Modifying SMTP Server Setup in the Core
- Connect the Build Farm.
  - Modifying Build Farm Configuration

MBaaS

- Enable the MBaaS and Cloud Apps to make use of all available system resources.
  - Adjusting System Resource Usage of the MBaaS and Cloud Apps
- Set up email configuration for the MBaaS.
  - Setting Up SMTP for Cloud App Alerts