1. RUNTIME PLATFORM INSTALL
   1. NEXUS
      1. Prerequisites

* Rhel OS with 1G RAM and 1 VCPU
* Internet access
* Filesystem directory with 10GB
* Linux user to be the owner of the install
* Port 8081 open for external access to the web console and for internal access from Fuse hosts
  + 1. Install details

Nexus repository is installed on host <IP> under /opt/nexus directory and belongs to the “nexus” user. The management interface can be reached on a browser at [http://10.40.11.24:8081](http://10.40.11.24:8081/). The login credentials are admin / ‘admin123’.

* + 1. Repository Configuration

We created a public group (named: **public**) which proxies the repositories that are used by Fuse by default (lised in the agent.properties file of the default profile). The Url of the group is ‘http://10.40.11.24:8081/repository/public/’.

The repositories originally configured in Fuse has been integrated into the Nexus configuration behind the “public” group.

|  |  |
| --- | --- |
| **Rpository Name** | **Remote URL** |
| fusesource | <http://repo.fusesource.com/nexus/content/groups/public> |
| fusesource-ea | <http://repo.fusesource.com/nexus/content/groups/ea> |
| fusesource-release | <http://repo.fusesource.com/nexus/content/repositories/releases> |
| jboss.org | <https://repositories.Jboss.org/nexus/content/repositories/public> |

The following repositories are hosted locally in Nexus and their purpose is to store the developed artifacts.

|  |  |
| --- | --- |
| **Repository Name** | **URL** |
| maven-release | http://10.40.11.24:8081/repository/maven-releases/ |
| maven-snapshot | http://10.40.11.24:8081/repository/maven-snapshots/ |

* 1. Jenkins
     1. Prerequisites
* Rhel OS with 2G RAM and 2 VCPU
* Filesystem directory with 5GB
* Linux user to be the owner of the install
* Port 8080 open for external access to the web console
* Firewall access toward Fuse (ports 22+8181) and the SCM (HTTPS/HTTPS)
  + 1. Install details

Jenkins is installed on host <IP> under /opt/jenkins directory and belongs to the “jenkins” user. The management interface can be reached on a browser at [http://](http://10.40.11.24:8081/)<IP>[:808](http://10.40.11.24:8081/)0. The login credentials are admin / ‘admin’.

* + 1. Install plugins

Git

Pipeline

* + 1. Job Configuration

Build (CI)

Deploy

Test (CD)

* + 1. Jenkins configuration

Tabs

DSL

* 1. FUSE/FABRIC
     1. Prerequisistes

The following prerequisites are needed to install Fuse on a rhel OS:

* Rhel OS with min 6G RAM and 2 VCPU
* Java JRE (1.7 or 1.8)
* a directory on the filesystem (10G recommended)
* a linux user and group to be the owner of the install
* the java JRE location defined as JAVA\_HOME in the user's profile file
* SSH connectivity for the install owner between the hosts (keys or password)
* Hosts reachable by their “hostname”, or all the “hostnames” defined in each /etc/hosts
* Firewall open between the hosts
* Firewall open with the other networks, at least for the port 8181, 9200 and 9000 and for the ports to be used by the applications

Also double check the supported configuration:

<https://access.redhat.com/articles/310603>

* + 1. Java 8

Java8 enables IPv6 by default. We needed to add the following line in the FUSE bin/setenv file to force the use of IPv4 addresses:

export KARAF\_OPTS=”-Djava.net.preferIPv4Stack=true”

* + 1. Environment description

Development

Jboss-fuse is installed on DEV in the directory **/opt/fuse/ and belongs to the fuseadmin/fuseadmin user & group.**

Before starting the Fabric, uncomment the “admin” user in **etc/user.properties.**

We edited the file **etc/system.properties** and changed karaf.name to fusedev for this instance. All other property files are configured already and predefined with the default values.

Production

Jboss-fuse is installed on PROD in the directory **/opt/fuse/ and belongs to the fuseadmin/fuseadmin user & group.**

The install has been done over “ nodes.

On each node, before starting the Fabric, uncomment the “admin” user in **etc/user.properties.**

Also we edited the file **etc/system.properties** and changed karaf.name to fusedev for this instance. All other property files are configured already and predefined with the default values.

* + 1. Install Fuse Fabric on a single node

To create a single node Fabric:

>fabric:create –zookeeper-password admin

When the fabric is created, parameter's values are read from the properties files and loaded into the zookeeper registry. The files will not be used anymore upon that.

* + 1. Install Fuse Fabric over 3 nodes

In prod, the Fabric is a 3 members Ensemble cluster. We recommend using “fabric:join” for the containers that will be used as cluster members. The reasons are:

* for all the Ensemble members to be located under the same path on the filesystem
* to prevent the registries from being accidentally stopped from the web console

The procedure to create the Fabric cluster are:

FUSEPRD1>fabric:create –zookeeper-password admin

FUSEPRD2>fabric:join –zookeeper-password admin

FUSEPRD3>fabric:join –zookeeper-password admin

FUSEPRD1>fabric:ensemble-add prd2 prd3

Only an even number of member can be added/removed from the cluster at any time.

The Fabric registries (fuseprd1, fuseprd2, fuseprd3) must be stopped/started using the command line. However, it's recommended to operate all the other containers (child or SSH) from the Fabric itself (via the CLI or the web console).

* + 1. Containers and profiles

Child containers

> fabric:container-create-child fusedev <containerName>

**SSH containers**

To create SSH containers, the hosts to which the containers will be created must be accessible by SSH for the user used as the install owner (fuseadmin). This can be password-based or key-based and password-less. Also we recommend you create the same directory path on all hosts to contain the SSH containers (in our case /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers).

Example of creating a ssh container command:

fabric:container-create-ssh –user fuseadmin –password redhat --host 10.40.11.26 --path /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers <containerName>

The above command tells Fabric to create the ssh-container with a login user as **fuseadmin** and password of **redhat** and the host you want the container to be created on. The –path option specifies the filesystem path of the ssh-container. SSH keys can be used for SSH access and in such a case the –password option is not needed. The containers have to be created with the same user as the platform owner (fuseadmin).

After creating each ssh-container, verify that it is created by going to /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers on each PROD VM and see that pgw1, pgw2 and pgw3 (and smsgw1, smsgw2, smsgw3) directories are created under it. Those directories will have a “fabric8-karaf-1.2.0.redhat-621084" subdirectory that contains a lightweight Karaf runtime (with bin, etc, data, log and others directories beneath it).

Issue a “fabric:container-list” on karaf to see the status of new containers.

Profiles are assigned to containers using:

> fabric:container-add-profile <containerName> <profileName>

* + 1. Brokers

**Single instance broker**

Here is the procedure to create this broker:

* create a child container:

>fabric:container-create-child fusedev <containerName>

* create a broker profile

>fabric:mq-create <brokerName>

The name of this profile is “mq-broker-default.<brokerName>”.

The “mq-create” command creates a broker that has mq-base as its parent, thus using kahadb.

* add the "mq-broker-default.phccamq" profile to the activemq child container

>fabric:container-add-profile <containerName> <profileName>

**Master/Slave brokers with network replication**

In prod, there are 3 brokers configured in a master/slave topology (1 master + 2 slaves) using the network-based level-db replication as the persistence adapter.

Here is the procedure to create such a broker:

* First, 3 containers are created using the following commands

|  |
| --- |
| fabric:container-create-ssh --host <IP> --user fuseadmin --path /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers amq1 |
| fabric:container-create-ssh --host <IP> --user fuseadmin --path /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers amq2 |
| fabric:container-create-ssh --host <IP> --user fuseadmin --path /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers amq3 |

* A levelDB-based broker profile is created with

>fabric:mq-create –parent-profile replicated –replicas 3 <brokerName>

The above command will create a profile named "mq-broker-default.<brokerName>". You should see this profile among others if you issue the command: "fabric:profile-list".

This profile has “mq-replicated” instead of “mq-base” as its parent profile. The mq-replicated profile is a broker profile preconfigured for the use of replicated level-db as the persistence adapter. Check the profile's properties file to make sure the profile is using the “broker.xml” file and not the “ssl-broker.xml” one.

The profile has to be assigned to at least 2 containers.

> fabric:container-add-profile amq1 mq-broker-default.<brokerName>

> fabric:container-add-profile amq2 mq-broker-default.<brokerName>

> fabric:container-add-profile amq3 mq-broker-default.phccamq

Note

The “replicas” parameter indicates the size of the replication cluster. The leveldb cluster works like zookeeper, which means that at least half of the number of instances must be alive for the service to work correctly. In our case, with a size of 3, this means that at least 2 amq brokers must be alive for the broker service to be available.

This is the exact same behaviour as for the Fabric, which will fall down in read-only mode if less than 2 of the 3 registries (fuseprd1, fuseprd2, fuseprd3) are available. Though 3 instances is the minimum required for HA, we recommend using a cluster of 5 members in **production** when possible.

* + 1. Nexus configuration

The linking to Nexus from Fuse console is done by configuring the Fabric agent's **org.ops4j.pax.url.mvn.repositories** property as shown below: The list usually configured has been replaced by 3 new entries.

|  |  |
| --- | --- |
| Fabric profile to link the Fuse to nexus is located under “Wiki” in the following location:  default/io.fabric8.agent.properties | org.ops4j.pax.url,mvn.repositories= \  file:${runtime.home}/${karaf.default.repository}@snapshot@id=karaf-default, \  file:${runtime.data}/maven/upload@snapshots@id=fabric-upload, \  <http://10.40.11.24:8081/repository/maven-snapshots/@id=snapshots>, \  <http://10.40.11.24:8081/repository/maven-releases/@id=releases>. \  <http://10.40.11.24:8081/repository/public/@id=public> |

* + 1. Fabric Insight configuration

Fabric Insight is an embedded ELK (ElasticSearch – LogStash – Kibana) implementation that provides data aggregation within the Fabric.

The profiles represented each of the 3 components are provided out-of-the-box within Fuse.

Here's the procedure to enable data aggregation with Insight

* Create an ssh-container

> fabric:container-create-ssh --host <IP>--user fuseadmin --path /opt/fuse/jboss-fuse-6.2.1.redhat-084/sshcontainers datastore

* Add “insight-elacticsearch.datastore” to this container

> fabric:container-add-profile datastore insight-elasticsearch.datastore

* Add “insight-console” to all registries (fuseprd1,fuseprd2,fuseprd3)

|  |
| --- |
| fabric:container-add-profile fuseprd1 insight-console  fabric:container-add-profile fuseprd2 insight-console  fabric:container-add-profile fuseprd3 insight-console |

* Add "insight-logs.elasticsearch" and "insight-camel" profiles to all ssh-containers hosting applications

|  |
| --- |
| fabric:container-add-profile pgw1 insight-logs.elasticsearch  fabric:container-add-profile pgw2 insight-logs.elasticsearch  fabric:container-add-profile pgw3 insight-logs.elasticsearch |
| fabric:container-add-profile pgw1 insight-camel  fabric:container-add-profile pgw2 insight-camel  fabric:container-add-profile pgw3 insight-camel |

* + 1. Summary table

The following table shows installed containers and profiles

|  |  |  |
| --- | --- | --- |
| ENVIRONMENT / Host | Container | Profiles |
| DEV  <IP> <HOSTNAME> | fuseprd1 | This container is the fabric registry.  It hosts  - zookeeper (**fabric-ensemle-000-1)**  - **insight-console** plugin |
|  | pgw1 | SSH container dedicated to applications   * phcc-pgw * insight-logs.ealsticsearch * insight-camel   The last 2 profiles enable the container to send all the logs, camel events to the Fabric Console Insight to be viewed and analyzed. |
|  | smsgw1 | SSH container dedicated to applications   * phcc-sms * phcc-saaj * insight-logs.ealsticsearch * insight-camel   The last 2 profiles enable the container to send all the logs, camel events to the Fabric Console Insight to be viewed and analyzed. |
|  | amq1 | SSH container dedicated to broker  - mq-broker.default.<brokerName> |

* 1. MYSQL
     1. Prerequisites
* Rhel OS with 4G RAM and 2 VCPU
* Filesystem directory with 20GB
* Linux user to be the owner of the install, owning the above FS directory
* Port 3006 open for external access to the Workbench and for internal access from Fuse hosts
  + 1. Install

MySQL v5.9 is installed on host <IP> with and belongs to the “mysql” user.

Download and install the following rpm packages

* MySQL-server-5.6.31-1.el7.x86\_64.rpm
* MySQL-client-5.6.31-1.linux\_glibc2.5.x86\_64.rpm
* Perl-Data-Dumper-2.145-3.el7.x86\_64.rpm

The runtime for MYSQL is then initiated by the **‘/usr/bin/mysqld\_install\_db’** script that must be run by the ‘mysql’ user.

MYSQL startup options are specified in ‘/etc/my.cnf’ file. The main properties are the access port ‘port = 3306’, ‘bind-address = 10.40.11.26’, socket location...

* + 1. Configuration

We created a start script in **/usr/lib/systemd/system/** with name **mysql.service** to allow MySQL to be automatically started as a service. The script specifies start and stop, User, Group and access-port (3306) for MYSQL server. Specifying ‘- - bind-address=10.40.11.26’ allows users to connect to MYSQL on this host using IPV4 protocol ( IPV6 is enabled by default).

MYSQL super-admin user/password is: root/mysql. One can change this password using following command: **‘mysqladmin –u root password ‘newpwd’’**. Login to mysql is done using: **‘mysql –u root –p’;** that is provided by the “client” RPM.

We created a new user (admin/admin) to be the owner of the “fusedb”. So this user has been granted full permission on all objects belonging to this database.

* + 1. **MySQL users and passwords**

A Mysql user is defined by 2 parts: [username@domain](mailto:username@domain), where the domain represents the URL from which the connection is established.

This means that [admin@10.40.11.24](mailto:admin@10.40.11.24) and [admin@10.40.11.25](mailto:admin@10.40.11.25) are seen as 2 different users with their own set of permissions. Each of those users must be set with an appropriate password:

> set password for 'admin'@'10.40.11.24' = PASSWORD('admin');

Also we need to grant permissions to the admin user:

Commands to create new users:

- CREATE USER 'admin'@'ictapp.phc.gov.qa' IDENTIFIED BY 'admin';

- SET PASSWORD FOR 'admin'@'ictapp.phc.gov.qa' = PASSWORD('admin');

- GRANT ALL ON fusedb.\* TO 'admin'@'ictapp.phc.gov.qa';

* + 1. Database tables

The following table is created in Mysql server for the paymentGW application:

CREATE TABLE pgw  
(  
   tid INT UNSIGNED NOT NULL AUTO\_INCREMENT,  
   PRIMARY KEY (tid),  
   date DATETIME NOT NULL,  
   user VARCHAR(128) NOT NULL,  
   amount INT NOT NULL,  
   status VARCHAR(16) NOT NULL,  
   receiptid VARCHAR(32) NOT NULL,  
   hash VARCHAR(64) NOT NULL  
);

The following tables are created in Mysql server for the SMS GW application:

CREATE TABLE sms  
(  
   id INT UNSIGNED NOT NULL AUTO\_INCREMENT,  
   PRIMARY KEY (id),  
   sentdate DATETIME NOT NULL,  
  groupid VARCHAR(128),

phoneno VARCHAR(16) NOT NULL,  
  status VARCHAR(16)  
);

CREATE INDEX groupid\_idx ON sms (groupid);

CREATE TABLE msg  
(

  groupid VARCHAR(128) NOT NULL,

message VARHCAR(5000) NOT NULL,

INDEX (groupid),

FOREIGN KEY (groupid) REFERENCES sms(groupid),

ON UPDATE CASCADE ON DELETE RESTRICT  
);

doesn't have any Open Source API Management product at the moment.