Deployment Guide

Published: April 1, 2015
Version: 10.0.0.1

Table of Contents

1. Introduction

1.1. Overview

This document explains the intended deployment plan for any operational instance of OZONE (otherwise known as OZP).

Its intended audience includes:

- Systems/network administration and operations teams
- System developers who need to troubleshoot or identify usage scenarios for feature development

1.2. Scope

While this document can be used as a resource for deploying any operational instance of OZONE, it concentrates specifically on an operational system deployment. This includes systems intended for pre-production testing (e.g. Integration/Staging/QA systems).

1.2.1. Deployment Goals

- 99.99% Availability
- Minimal Performance Degradation under high load
- Efficient use of the utility resources (minimization of operational costs)
- Capable of handling > 25,000 users simultaneously

2. Deployment Requirements & Architecture

2.1. Hardware

The deployment described in this document uses the following hardware. Hardware specifications are per server/virtual machine:

1. Backend Apache Tomcat (JEE) Component — 1 Server/VM
   - 2 CPUs
   - 4GB RAM
   - 40GB Disk Space
2. MariaDB (RDBMS) Component — 1 Server/VM
   - 4 CPU
   - 16GB RAM
   - 60GB Disk
3. MongoDB (NoSQL) Component — 2 Servers/VMs
   - 4 CPU
   - 16GB RAM
   - 20GB Disk
4. NGINX (Static Web Content Distribution) Component — 1 Server/VM
   - 2 CPU
   - 2GB RAM
   - 60GB Disk
5. NGINX (Dynamic Web) Component — 1 Server/VM
   - 2 CPU
   - 4GB RAM
   - 70 GB Disk
6. HAProxy (Load Balancing) Component — 1 Server/VM
   - 2 CPU
   - 4GB RAM
   - 40GB Disk

2.2. Software Dependencies

The deployment described in this document uses the following set of (non-developed) software to create the stacks associated with each tier.

- OpenResty/NGINX 1.7.10
- Apache HTTPD 2.4
- Apache Tomcat 7.0
- HAProxy 1.5
- MariaDB 5.5
- MongoDB 2.6

- Java Development Kit 1.7

> JDK Note: The team prefers Oracle/Sun's JDK, but OpenJDK will function.

2.3. Networking

The OZONE deployment described in this document is deployed on a private VLAN that explicitly exposes various endpoints using Firewall and Network Address Translations (NAT).

Networking requirements:

- VLAN/private network
- Firewall with NAT Support

2.4. Security Considerations & Requirements

In the example deployment, all external-faced endpoints use SSL (TLSv1+). To provide this function, all external IP addresses and Hostnames require an associated SSL certificate from a Trusted Certificate Authority. *(Other deployments should follow their own policies for SSL/PKI Certificate use.)*

At the present time, the internal PKI service generates the certificates. Any external clients will be required to pre-load the internal Certificate Authority (OZP-Root-CA) in order to enable a transparent trusted connection (i.e. no "untrusted connection" messages).

2.5. Deployment Architecture

The OZONE operational system uses a multi-tiered, modular architecture composed of:

- Tier 0: Load Balancing
- Tier 1: Web Content Distribution
  - Module A: Static Web Content Distribution
  - Module B: Dynamic Web Content Distribution
- Tier 2: Middleware Analysis & Processing
- Tier 3: Data Persistence
  - Module A: Key-Value/Document Storage (NoSQL)
  - Module B: Relational Data

2.5.1. Platform Stack

2.5.1.1. General

OZONE uses an integration of multiple stacks instead of a single platform stack. All stacks consist of a common base: CentOS 6 x86-64.

Figure 1: Platform Stack for Metrics App

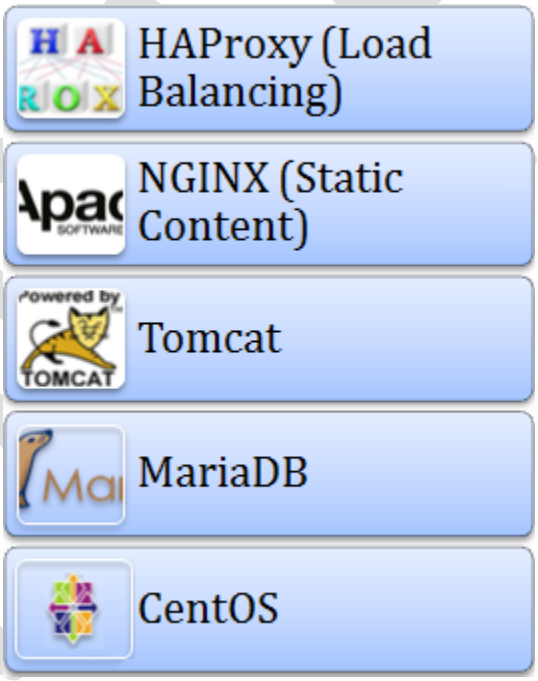

Figure 2: Platform Stack for Main App

2.5.2. Tier Platform Stack Software Mapping

The following section maps the software used in the platform stacks to the tiers described in section 2. If you already set up a dependency as part of the overall system dependencies (listed in 2.2), it will be italicized.

Tier 0: Load balancing

- CentOS

- *HAProxy*

Tier 1: Web Content Distribution

- Module A: Static Web Content Distribution
  - CentOS
  - *OpenResty (NGINX)*
- Module B: Dynamic Web Content Distribution
  - CentOS
  - *Apache HTTPD*

Tier 2: Middleware Analysis & Processing

- CentOS
- *Apache Tomcat*

Tier 3: Data Persistence

- CentOS
- *MariaDB*

3. Installation and Configuration Overview

After installing and configuring the necessary requirements, you must prepare OZONE. The remaining sections explain how to install and configure it. The process follows this order:

1. Deploy OZP Rest Backend
    a. Deploy MariaDB
    b. Deploy Elasticsearch Cluster
    c. Deploy OZP Rest Web Application
2. Deploy Metrics Service
    a. Deploy MongoDB Cluster
    b. Deploy NGINX Server with PHP
    c. Deploy Metrics application (PHP-based)
3. Deploy Client Application components (HUD, Webtop, Center and IWC)
    a. Deploy NGINX Server
    b. Deploy pre-built IWC distribution
    c. Deploy pre-built HUD distribution
    d. Deploy pre-built Center distribution
    e. Deploy pre-built Webtop distribution

4. OZP REST Backend Deployment

4.1. Deploy MariaDB

4.1.1. Software Versions

The deployment in this example uses this specific version of MariaDB:

- MariaDB 5.5.42

4.1.2. Prerequisites

- Available large disk storage area (>10GB)
- YUM repo with Elasticsearch RPM
    - If you have Internet access, install the yum repo per the instructions at https://mariadb.com/kb/en/mariadb/yum/
    - For private networks, the following MariaDB RPMs should be downloaded, and loaded into a custom YUM repo
        - MariaDB-5.5.42-centos6-x86_64-client.rpm
        - MariaDB-5.5.42-centos6-x86_64-common.rpm
        - MariaDB-5.5.42-centos6-x86_64-compat.rpm
        - MariaDB-5.5.42-centos6-x86_64-server.rpm
        - MariaDB-5.5.42-centos6-x86_64-shared.rpm
- Root DB user password

ASSUMPTION
/data is already created with 755 root:root permissions, on a partition with >10GB free space

4.1.3. Overview

1. Install Package from YUM repo
2. Create Data Storage Folder
3. Configure MariaDB
4. Configure Firewall to allow ingress
5. Configure and Start Service
6. Initialize Database Server
7. Verify MariaDB

4.1.4. Procedure

1. Install Package from YUM repo:
   ```
   yum install MariaDB-server
   ```
2. Create Data Storage Folder:
   ```
   install -o mysql -g mysql -m 775 -d /data/dbs
   ```

3. Configure MariaDB:
   - Update /etc/my.cnf with the following entries

     ```
     [server]
     datadir=/data/dbs
     ```
4. Configure Firewall to allow ingress to node
   - Add the following line to `/etc/sysconfig/iptables` before any REJECT rules

     ```
     A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT
     ```
   - Reload the rules

     ```
     service iptables reload
     ```
5. Configure and start the service

   ```
   chkconfig mysql on
   service mysql start
   ```

   NOTE: If you receive a failure when starting, you will need to check permissions on the data directory (`/data/dbs`) to ensure the `mysql` user can read and write to it.
6. Initialize Database Server

   ```
   sudo -l -u mysql mysql_install_db --datadir=/data/dbs
   ```

4.2. Deploy Elasticsearch Clustering

Prerequisites

- Java JDK Installed on system
  - Oracle JDK is recommended (OpenJDK 1.7+ is sufficient)
- Available Large disk storage area (>10GB)
- YUM repo with Elasticsearch RPM
  - If you have Internet access, the main Elasticsearch repo can be used per these instructions: http://www.elastic.co/guide/en/Elasticsearch/reference/1.4/setup-repositories.html#_yum
  - For private networks, the Elasticsearch RPM should be downloaded, and loaded into a custom YUM repo

ASSUMPTION

The `/data` partition is pre-created on a disk with >10G available.

4.2.1. Instructions

1. Install the YUM package from its repo:

   ```
   yum install Elasticsearch
   ```
2. Create the Data Storage Folder:

   ```
   install -o Elasticsearch -g Elasticsearch -m 775 /data/es
   ```
3. Configure Elasticsearch:

Search for the associated keys (text before the ':'). If it is commented out of the code, add it back into the active code. Update the section with the following values, substituting locally relevant values for anything between angle brackets:

```
cluster.name: ozp-<dev|int|prod>-search
index.number_of_replicas: 2
path.data: /data/es
path.work: /var/lib/Elasticsearch
path.logs: /var/log/Elasticsearch
bootstrap.mlockall: true
discovery.zen.ping.multicast.enabled: false
discovery.zen.ping.unicast.hosts: ["<es-node1", "<es-node2>"]
```

Note: List all nodes here, including the one you are configuring

4. Configure the Firewall to allow ingress to node:
   a. Add the following line to `/etc/sysconfig/iptables` before any REJECT rules:

```
A INPUT -m state --state NEW -m tcp -p tcp --dport 9200:9400 -j
ACCEPT
```

   b. Reload the rules:

```
service iptables reload
```

5. Configure and start service:

```
chkconfig Elasticsearch on
service Elasticsearch start
```

6. Verify Elasticsearch service

```
curl http://localhost:9200/_cluster/health
```

4.3. Install OZP-REST Web Application

After installing MariaDB and Elasticsearch, install OZP-REST. Prerequisites and assumptions are listed before the instructions.

Prerequisites

Before you begin the OZP-REST install, the following things must be configured:

- Customized OZP-REST WAR see 4.3.1: Customize the OZP-REST WAR.
- Establish PKI certificates for SSL see 2.4: Security Considerations & Requirements.
- Deploy the Elasticsearch cluster and confirm that it is accessible from the ozp-rest host.
- MariaDB setup with
  - An empty database `ozp` .
  - An `ozpuser` user that has full privileges on the `ozp` database
  - The `ozpuser` user has access from OZP-Rest host(s)

    - Connection information for `ozpuser`
- Access to MariaDB client `mysql` (preferably from a separate host).

Resources

- Database schema included in https://jenkins.di2e.net/view/OZP/job/build-ozp-rest-master/lastSuccessfulBuild/artifact/rest-backend-0.7.4.tar.gz

4.3.1. Customize the OZP-REST WAR

The software provides the OZP Rest WAR as a FOSS, general purpose web application. For specific deployments, the web app may require additional customizations to comply with the enterprise infrastructure policies.

Resource

- Source Code Repository: https://github.com/ozone-development/ozp-security
- Built Artifact (DI2E Account Required/No VPN): https://jenkins.di2e.net/view/OZP/job/OZP-Security-Plugin/lastSuccessfulBuild/artifact/target/ozone-security-4.2.jar

Prerequisites

Install software:

- Unzip (`yum install -y unzip`)
- Java 1.7+ (OpenJDK is fine)

4.3.1.1. Instructions: Customizing the OZP-REST WAR

1. Explode the WAR file to a temporary directory:

```
mkdir /tmp/extract/ozp-rest
cd /tmp/extract/ozp-rest
unzip <path-to-ozp-rest-war>
```

2. Replace the security plugin with the OZP IAA Plugin:

```
rm -f WEB-INF/lib/ozone-security*.jar
wget -O WEB-INF/lib/ozone-security-4.2.jar
https://jenkins.di2e.net/view/OZP/job/OZP-Security-
Plugin/lastSuccessfulBuild/artifact/target/ozone-security-4.2.jar
```

3. Repack the WAR:

```
jar cf /tmp/extract/ozp-rest-aml.war *
```

4. Upload it to the CM repository.

4.3.2. Install Apache Tomcat

4.3.2.1. Software Versions

The deployment in this example uses this specific version of Apache Tomcat:

- Apache Tomcat 7.0.33

4.3.2.2. Prerequisites

- YUM repo with Apache Tomcat RPM
  - Use the standard (pre-loaded) and EPEL repositories (`yum install -y epel-release`)
  - PKI/SSL Certificates Keystore and Trust Store ready and available

ASSUMPTION

Keystore is stored in `/etc/pki/tls/keystores/<fqdn>.jks` and is password protected

where `<fqdn>` is the fully-qualified domain name of the local host (e.g. appsvr01.amlng.ab2e.net).

Trust Store is stored at `/etc/pki/CA/ozp-trusts.jks`. It is password protected.

4.3.2.3. Outcome

After completing this task, Apache Tomcat will run on the local system using ports 8443 and 8009. Connections on port 8443 will be encrypted using SSL, and 8009 connections will use the Apache AJP communications protocol.

However, web apps will not start or run.

4.3.2.4. Overview

1. Install Package from YUM repo
2. Configure Tomcat Connector (`/usr/share/tomcat/conf/server.xml`)
3. Configure JVM options for SSL and Heap Memory (`/etc/sysconfig/tomcat`)
4. Configure Firewall to allow ingress
5. Configure and Start Service

4.3.2.5. Procedure

1. Install Package from YUM repo:
   `yum install -y tomcat`
2. Configure the Tomcat Connector in `/usr/share/tomcat/conf/server.xml/`:
   Under the `<Service name="Catalina">` section, add the following entry:

```
<Connector
        protocol="HTTP/1.1"
        port="8443" maxThreads="200"
        scheme="https" secure="true" SSLEnabled="true"
        keystoreFile="/etc/pki/tls/keystores/<fqdn>.jks"
        keystorePass="<keystore-password>"
        truststoreFile="/etc/pki/CA/ozp-trusts.jks"
        truststorePass="<truststore-password>"
```

```
clientAuth="want" sslProtocol="TLS"/>
```

NOTE

We need to set `clientAuth` to "`want`" instead of "`true`" due to the use of the HTTP OPTIONS request, which, by the official standard, will *not* pass any kind of authentication information; it is expected to be able to anonymously request OPTIONS

3. Configure the JVM options for SSL and Heap Memory by adding the following entries to the `/etc/sysconfig/tomcat` file:

```
export JAVA_OPTS="${JAVA_OPTS} -Xms2g -Xmx2g -XX:MaxPermSize=512m"

      export JAVA_OPTS="${JAVA_OPTS} -
Djavax.net.ssl.keyStore=/etc/pki/tls/keystores/$(hostname --fqdn).jks"

      export JAVA_OPTS="${JAVA_OPTS} -
Djavax.net.ssl.keyStorePassword='"$(echo <keystore-password>)"'"

      export JAVA_OPTS="${JAVA_OPTS} -
Djavax.net.ssl.trustStore=/etc/pki/CA/ozp-trusts.jks"

      export JAVA_OPTS="${JAVA_OPTS} -
Djavax.net.ssl.trustStorePassword=<truststore-password>"
```

4. Configure the firewall to allow ingress to the SSL connector port and the AJP connector port by adding the following line to `/etc/sysconfig/iptables` before any REJECT rules:

```
      A INPUT -m state --state NEW -m tcp -p tcp --dport 8443 -j ACCEPT

      A INPUT -m state --state NEW -m tcp -p tcp --dport 8009 -j ACCEPT
```

5. Reload the rules:
   `service iptables reload`
6. Configure and start the service
   `chkconfig tomcat on`
   `service tomcat start`

4.3.3. Install OZP-REST Deployment Instructions

Note: Perform all the steps at the root level. For CentOS 6, the <tomcat-root> directory is normally located at `/usr/share/tomcat`.

1. Install Apache Tomcat

```
Yum install -y tomcat
```

2. Install OZP server-level configuration files

```
<tomcat-root>/lib/SecurityConfig.properties

<tomcat-root>/lib/SecurityContext.xml

<tomcat-root>/lib/MarketplaceConfig.groovy
```

3. Configure SSL on Tomcat

a. Configure the JVM to use the PKI certificate and make it accessible to the OZP REST Web app
b. Configure the HTTPS connector in `server.xml`.

4. Load Schema into MariaDB

```
mysql -u ozpuser -h <mariadb-host> -D ozp < <DB-Schema-File>
```

5. Load Initial Data into MariaDB

```
mysql -u ozpuser -h <mariadb-host> -D ozp < <DB-Data-File>
```

6. Install (deploy) the customized OZP-REST WAR
   a. From a Web location

```
wget -O <tomcat-root>/webapps/ozp-rest.war http://infra-
adm01/cfgmgmt/apps/ozp-rest/war-latestchown tomcat:tomcat <tomcat-
root>/webapps/ozp-rest.war
```

   b. From a local download of the WAR

```
install -o tomcat -g tomcat -m 644 <local-path>/ozp-rest.war
<tomcat-root>/webapps/ozp-rest.war
```

7. Start Tomcat and enable auto-start

```
chkconfig tomcat on

service tomcat start
```

8. Check for errors and attempt to resolve them.
   a. Review the following logs in <tomcat-root>/logs/
      i. `catalina.out`
      ii. `marketplace.log`
      iii. `stacktrace.log`
      iv. `localhost.<date>.log`

5. Install OZP-Metrics

5.1. Deploy NGINX Server with PHP

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

5.1.1. Deployment Instructions

1. Install OpenResty

5.2. Deploy Metrics application (PHP-based)

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

5.2.1. Deployment Instructions

6. Deploy Client Application Components (HUD, Webtop, Center and IWC)

6.1. Deploy NGINX Server

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

6.1.1. NGINX Deployment Instructions

6.2. Deploy pre-built IWC distribution

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

6.2.1. IWC Deployment Instructions

6.3. Deploy pre-built HUD distribution

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

6.3.1. HUD Deployment Instructions

6.4. Deploy pre-built Center distribution

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

6.4.1. Center Deployment Instructions

6.5. Deploy pre-built Webtop distribution

Prerequisites

asdfa

Assumptions

aldk

Resources

Ladjk

6.5.1. Webtop Deployment Instructions