WEBADM HIGH AVAILABILITY GUIDE

WebADM High Availability Guide

HA Cluster

1. Product Documentation

This document is a deployment guide for RCDevs WebADM in high availability (or cluster) mode. The reader should notice that this document is not a guide for installing WebADM applications (Web Services and WebApps).

2. Product Overview

WebADM is a powerful Web-based LDAP administration software designed for professionals to manage LDAP Organization resources such as Domain Users and Groups. It is the configuration interface and application container for RCDevs Web Services and WebApps such as OpenOTP. WebADM requires an LDAP directory as back-end user store and a SQL database for logs and end-user message customizations. WebADM is compatible with Novell eDirectory, OpenLDAP, RCDevs Directory Server and Microsoft ActiveDirectory 2003/2008.

3. System Requirements

The current version of WebADM runs on Linux 32bit or 64bit operating systems with GLIBC >= 2.5. The installation package contains all the required dependencies allowing WebADM to run on any Linux-based system without other requirements. WebADM only needs an LDAP backend (Novell eDirectory, OpenLDAP, RCDevs Directory Server, Microsoft ActiveDirectory, or Oracle Directory) and a SQL backend (MySQL, PostgreSQL, Oracle or Microsoft SQL).

For running WebADM and its applications, as well as the OpenOTP Radius Bridge server and RCDevs Directory Server, your system should fit the following requirements:

- A dedicated server computer or Virtual machine with Linux GLIBC >= 2.5 (RedHat, Centos, SUSE, Debian, Ubuntu).
- 2 GHz processor (multi-core / multi-thread processor is highly recommended). Both 32 and 64-bit chips are supported provided that 32 libraries are present.
- 2GB RAM memory.
- 2GB disk space for installation files.
- Network access with DNS and NTP integration.
- A local or remote LDAP directory server (RCDevs Directory Server, OpenLDAP, Novell eDirectory or Microsoft ActiveDirectory >= 2003). WebADM for ActiveDirectory 2003 has some limitations which do not exist with ActiveDirectory 2008. Always prefer using ActiveDirectory 2008 with WebADM.
- A local or remote SQL database server (MySQL, PostgreSQL). Oracle and MS SQL Server support are included but setup might require manual table creation.
- Outbound Internet access for checking versions, connecting SMS gateways and sending emails.
- A local mail transfer agent (Sendmail or Postfix).
- Firewall open ports: 80, 443, 8080, 8443, 1812. Some other ports are required for cluster node communications as described

later.

4. High Availability Mechanisms

⚠ Warning

Starting from WebADM version 1.4.2, any high availability and clustering feature require an RCDevs Enterprise license. Without a valid license file, the HA and cluster features are automatically disabled.

WebADM supports several high-availability mechanisms for internal and external service failover and for the whole system redundancy. It supports connecting several external data sources such as LDAP directories and SQL databases at the same time and does automatic failover. WebADM connects by default the first declared service (LDAP / SQL / Session Manager / Proxy) and transparently switches to a secondary service in case of primary service failure.

For systems requiring high-availability and near-zero downtime, WebADM supports cluster setup. In cluster mode, the whole system and services can be deployed on two or more servers for ensuring global redundancy, failover and even load-balancing functionalities.

4.1 Connecting Redundant External Services

To enable more than one connection to external services, you just need to configure the external services' connections in the /opt/webadm/conf/servers.xml configuration file. WebADM will automatically check for service responsiveness in the order the services are specified. It will also connect the first declared service in priority but if this service goes down, it will try to connect the next responsive service. When connected to a non-primary service, WebADM will re-check if the primary service has recovered every minute. If at one moment, the service goes up again, WebADM will reconnect its primary service immediately.

The external service switching works for any server connection defined in the `/opt/webadm/conf/servers.xml` file. Failover is done transparently by WebADM and your client systems and end-users won't be affected by the automatic external service switching.

Note

The WebADM session manager and PKI server are specified in the servers.xml file but are local WebADM services (part of the WebADM software).

4.1.1 Connecting Two LDAP Servers

In this example, WebADM uses "LDAP Server 1" by default and switches to "LDAP Server 2" in case "LDAP Server 1" goes down.

```
<LdapServer name="LDAP Server 1"
 host="server1"
 port="389"
 encryption="TLS" />

<LdapServer name="LDAP Server 2"
 host="server2"
 port="389"
 encryption="TLS" />
```

It is mandatory that the two LDAP servers use replication. This is automatic with Active Directory when using two domain controllers in the same domain or with Novell eDirectory when LDAP partition replication is set up. RCDevs Directory Server an OpenLDAP require LDAP replication configuration. Please refer to the OpenLDAP documentation for OpenLDAP replication.

Remark

Local LDAP connection does not need a security transport layer. Yet, remote LDAP connections should use SSL or TLS if there is a risk of network packet sniffing between the servers.

The LDAP server (Novell eDirectory, OpenLDAP or RCDevs directory server) can be installed and run on one or several of the cluster nodes. They can be deployed on another dedicated server too.

4.1.2 Connecting Two SQL Servers

The following example illustrates two redundant SQL servers.

```
<SqlServer name="SQL Server 1"
 type="MySQL"
 host="server1"
 user="webadm"
 password="rwebadm"
 database="webadm" />

<SqlServer name="SQL Server 2"
 type="MySQL"
 host="server2"
 user="webadm"
 password="rwebadm"
 database="webadm" />
```

It is preferred that SQL databases use replication but this is not a requirement. It’s a requirement if you use Hardware Tokens with WebADM because Token inventory is stored in the SQL.

4.2 Installing WebADM In Cluster-Mode

All the components in WebADM have been designed to support clustering. In this case, the WebADM components (i.e. the WebADM and Radius Bridge software) are deployed on several server computers to provide redundancy, failover or load-balancing.

4.2.1 WebADM Internal Components

A WebADM server includes several internal components. These components are local TCP/IP network services (just like the external services) started by the WebADM startup script and part of the base installation. They must be correctly configured for working in cluster mode.

The HTTP and SOAP server

The internal Web server provides the SOAP-based web services on port HTTP 8080 and HTTPS 8443. And it provides the Admin Portal and end-user WebApps on HTTPS port 443. SSL server certificates are automatically generated during the initial setup by an internal self-signed certificate authority (CA).

In cluster mode, all the services running over SSL/TLS must have certificates issued by one central certificate authority. And only one cluster node will play the role of the certificate authority. It is a requirement that all the HTTPS services which provide authentication based on client certificates, trust the client certificates issued centralized CA.

The session manager

This component handles all the user sessions initiated by web services such as OpenOTP and the WebApps. Even if multiple session managers can be specified on each node for failover purposes, in cluster mode, only one session manager should be used for all the cluster nodes at one moment. This is required for the cluster session sharing system to ensures clients requests will be handled correctly whatever node is used and to ensure user data integrity remains consistent. The session manager is used by the cluster nodes to communicated internal information too, such as configuration updates.

Note

With WebADM >= 1.2.6-1, the session manager supports automatic synchronous replication. Session data are replicated in real-time between the two first session servers in your configuration. Failover to the secondary node does also not break running sessions.

Web services’ sessions are also shared for the whole cluster so that internal user working data and user locks remain coherent over your cluster service nodes. The WebADM WebApps use the session manager to handle user login sessions too. This has the big advantage that user browser requests can come randomly to any HTTP service node without impacting the system or the client. This is very handy for working with round-robin load-balancers in front of the service nodes.

The PKI server

One node is assigned the certificate authority role. It will run the WebADM Rsignd service which provides certificate signing for the local node and for your other cluster node. The PKI is required during the setup of your cluster nodes for generating SSL server certificates and configuring local CA trusts. It is used by the Admin Portal and the WebApps for issuing and renewing administrator and WebApp user certificates too.

5. Cluster Setup

In this section, we will describe how to set up a cluster configuration for WebADM and Radius Bridge. The cluster will provide redundant web services (ex. OpenOTP), WebApps and RADIUS authentication services.

5.1 Installing The First Node

The first node of your cluster is a standard WebADM installation and there is nothing specific to be configured on your first WebADM system. Yet, some firewall ports will have to be opened for allowing the others nodes to communicate with the internal services such as the session manager and PKI server.

The setup of the primary node is started with the command `/opt/webadm/bin/setup`. The setup will initiate the CA, create local service certificates, setup permissions, etc… The configuration of the `servers.xml` file will contain the following information:

LDAP Servers:

- LDAP 1
- LDAP 2

SQL Servers:

- SQL 1

Session Manager:

- Localhost
- <secondary server>

PKI Server:

- Localhost

In this example, we connect two LDAP servers for redundancy and only one SQL server. The Radius Bridge is installed on the same server running WebADM and uses the following OpenOTP URL in the `/opt/radiusd/conf/openotp.conf`: http://localhost:8080/openotp/

5.2 Installing A Secondary Node

The node is installed with the self-installer packages like with the primary node but the setup script must be run using the `slave` parameter with the command: `/opt/webadm/bin/setup slave`.

The setup can be re-run on an existing installation. You can also install a second VMWare appliance and re-run its WebADM setup script after installation for adding the node to your cluster.

The secondary node should use the same configuration files as the primary node. You can copy the `/opt/webadm/conf/webadm.conf` file from the primary node. Special attention should be given to the LDAP encryption key which must be the same on all your cluster nodes.

The `/opt/webadm/conf/servers.xml` should use the same LDAP / SQL servers and in the same order. The session management services will be running on both servers but only one of them must be used at a time by both servers. The two session managers can also be specified in the `servers.xml` files but in the same order. It is possible to use the local session manager on both servers when both WebADM servers are used in failover only and are never used at the same time.

The PKI server will not be set up nor run on the secondary server. The server will use the primary server PKI. During the setup in slave mode, the script will ask for the IP address, port number and secret of the primary server PKI. It will communicate with the remote PKI to initialize its SSL certificates ad CA trusts.

Proceed with the following steps for your secondary node installation:

1. On the primary server, allow client PKI connections to the Rsignd PKI server. This is done by adding a client configuration block for the secondary server in the `/opt/webadm/conf/rsignd.conf` file:

```
client {
 hostname 127.0.0.1
 secret secret
 services getcacert signcsr
}

client {
 hostname <secondary node IP>
 secret secret
 services getcacert signcsr
 }
```

You can add the secondary server's session manager in the `/opt/webadm/conf/servers.xml` for session manager redundancy:

```
<SessionServer name="Session Server 1"
 host="localhost"
 port="4000" />
<SessionServer name="Session Server 2"
 host="secondary node IP"
 port="4000" />
<PkiServer name="PKI Server"
 host="localhost"
 port="5000"
 secret="secret" />
```

Restart the WebADM server with the command: `/opt/webadm/bin/webadm restart`.

2. On the primary server, you must allow network communication to the session manager and PKI server ports from the secondary server. On Linux edit the `/etc/sysconfig/iptables` file and the line:

```
# Port for PKI server
-A INPUT -p tcp -m tcp -s <secondary node IP> -j ACCEPT --dport 5000

# Port for Session Manager access & session replications (for WebADM >= 1.3.x)
-A INPUT -p tcp -m tcp -m multiport -s <secondary node IP> -j ACCEPT --dports
11211,11212
-A INPUT -p udp -m udp -m multiport -s <secondary node IP> -j ACCEPT --dports
11211,11212

# Port for Session Manager access & session replications (for WebADM 1.4.x)
-A INPUT -p tcp -m tcp -s <secondary node IP> -j ACCEPT --dport 4000

#Port TCP 5000 is used for the PKI server.
#Port TCP 11211 is used for the session manager on WebADM 1.3.x.
#Port TCP 4000 is used for the session manager on WebADM >= 1.4.x.
```

Also add a firewall rule for SOAP services inter-communications:

```
-A INPUT -p tcp -m tcp -m multiport -s <secondary node IP> -j ACCEPT --dports 8080,8443
```

Restart the local firewall with the command:

```
/etc/init.d/iptables restart
```

3. On the secondary server, run the setup script in slave mode with the command:

```
/opt/webadm/bin/setup slave
```

You will be asked for the PKI server IP address, port, secret. The address is the primary node IP. The port is 5000. And the secret is 'secret' or the secret you have defined in the `/opt/webadm/conf/rsignd.conf` file on the primary server for the secondary server client. The SSL certificates are generated on the primary node and the CA certificate is installed in the local CA trust list.

4. On the secondary server, configure the `/opt/webadm/conf/servers.xml` file to use the session manager and PKI server from the primary server.

```
<SessionServer name="Session Server 1"
 host="primary node IP”
 port="4000" />

<SessionServer name="Session Server 2"
 host="localhost"
 port="4000" />

<PkiServer name="PKI Server"
 host="primary node IP"
 port="5000"
 secret="secret" />
```

⚠ Warning

Note here that the first declared session manager is the primary server. And there is no PKI server redundancy.

5. On the secondary server, add the firewall rules to allow communications from the primary server.

```
# Ports for Session Manager access & session replications
-A INPUT -p tcp -m tcp -s <primary node IP> -j ACCEPT --dport 4000
# Ports for WebADM SOAP server
-A INPUT -p tcp -m tcp -m multiport -s <primary node IP> -j ACCEPT --dports 8080,8443
```

You can now start the WebADM server on the secondary node. IMPORTANT: If you get a message like “Connected Session server: ERROR (no servers available)” when starting the WebADM server, then be sure the TCP port 4000 is correctly opened in both directions (on both server for the other node IP). You can do the following command to check if the remote port is opened.

```
telnet <Other Node IP> 4000
```

6. On the secondary node, configure the Radius Bridge exactly like on the primary node. You cluster configuration will look like this:

5.3 LDAP Replication

LDAP replication may differ according to the chosen LDAP implementation. With Active Directory, replication is handled by the Domain Controllers. With Novell eDirectory, replication requires a partition to be set up and replication should be configured with Novell eManager. With RCDevs Directory Server and more generally with OpenLDAP, the replication uses the syncprov overlay. The recommended is a master-master mirror configuration. On the master node, edit the `/opt/slapd/conf/slapd.conf` file, uncomment the replication block and configure it this way:

```
serverID 1
syncrepl rid=001
 provider=ldap://<secondary node IP>
 bindmethod=simple
 binddn="cn=admin,o=Root"
 credentials="your admin password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="60 +"
mirrormode on
```

On the secondary node, configure the replication this way:

```
serverID 2
syncrepl rid=001
 provider=ldap://<primary node IP>
 bindmethod=simple
 binddn="cn=admin,o=Root"
 credentials="your admin password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="60 +"
mirrormode on
```

On both node, be sure to authorize the LDAP port at the firewall level by adding the rules below:

On theprimary node:

```
-A INPUT -p tcp -m tcp -s <secondary node IP> -j ACCEPT --dport 389
```

On secondary node:

```
-A INPUT -p tcp -m tcp -s <primary node IP> -j ACCEPT --dport 389
```

6. Common Cluster Scenarios

Depending on your cluster usage (failover+load-balancing or failover only), you may configure and use your systems in different manners. The two scenarios explained below are the most common use of WebADM cluster. Yet other configurations are possible and you may understand in details how WebADM services and connectors work in order to fine-tune your cluster setup.

6.1 Load-balanced + Failover WebADM Cluster

This is the scenario which corresponds to our previous example. Both WebADM servers, Web services, WebApps can be used at the same time. The remote services (LDAP servers and SQL servers) should be used in the same order by both servers and they need to be replicated. Unless the LDAP servers use a real-time replication, it is required to use one (and the same) server at a time. Else the user data on the LDAP store could become inconsistent on the different nodes of your cluster during the LDAP replication delay.

The session management services must be used in the same order too. This is required for session sharing and cluster-level operation locking since both WebADM servers are supposed to randomly handle client requests at the same time.

The PKI server runs on the primary WebADM server only. The second server is configured to contact Server 1 for any PKI operation. This is a requirement in any cluster installation since there can be only one certificate authority on the cluster. Note that having the PKI service down does not impact the normal operations of the cluster.

On Server 1

```
LDAP Servers: LDAP 1, LDAP 2
SQL Servers: SQL 1, SQL 2
Session Manager: Localhost, Server 2
PKI Server: Localhost
```

On Server 2

```
LDAP Servers: LDAP 1, LDAP 2
SQL Servers: SQL 1, SQL 2
Session Manager: Server 1, Localhost
PKI Server: Server 1
```

6.2 Failover WebADM Cluster

In this mode, only the primary WebADM server is used in a normal situation. The secondary server ensures redundancy and is used only in the event where the primary server is not available.

The remote services (LDAP servers and SQL servers) can be used in the same order or in a different order. This is not important since the two cluster nodes are not used at the same time and do not require real-time LDAP data consistency. With clusters having the LDAP services deployed directly on the cluster nodes (Ex. RCDevs Directory Server), both servers may be connected to their local LDAP only.

Both servers can use their local session manager only as they do not need to share sessions and distributed locks.

The PKI server still needs to be run on the primary WebADM server only (for the same reasons as explained previously).

On Server 1

```
LDAP Servers: LDAP 1, LDAP 2
SQL Servers: SQL 1, SQL 2
Session Manager: Localhost
PKI Server: Localhost
```

On Server 2 (Alternative 1)

```
LDAP Servers: LDAP 1, LDAP 2
SQL Servers: SQL 1, SQL 2
Session Manager: Localhost
PKI Server: Server 1
```

On Server 2 (Alternative 2)

```
LDAP Servers: LDAP 2, LDAP 1
SQL Servers: SQL 2, SQL 1
Session Manager: Localhost
PKI Server: Server 1
```

7. Client Configuration

Your client applications using WebADM services or RADIUS can now use both cluster nodes, either at the same time with a round-robin policy for load-balancing, or in failover mode.

7.1 Web Application Using SOAP

Your web application can make SOAP calls to any web service node. With the shared session manager, client requests can come to any of the nodes, even if they are part of the same sequential OpenOTP authentication session.

7.2 VPN Server Using RADIUS

The VPN client can send RADIUS requests to any of the cluster nodes. With the shared session manager, even sequential RADIUS operations such as Challenge-Responses can come to any of the nodes.

7.3 End-user WebApps

The WebApps can be deployed on cluster nodes like the web services and RADIUS services. The shared session manager will ensure that user sessions are opened and synchronized on any of the cluster nodes and client accesses can even come randomly to any of the cluster nodes.

8. Dedicated Node Roles

WebADM is composed of several components which can be assigned to a specific node in your cluster. You can disable a component (and also a role) on a node by editing the `/opt/webadm/conf/webadm.conf` file.

By default, a node has all the roles enabled:

```
enable_admin Yes
enable_manager Yes
enable_webapps Yes
enable_websrvs Yes
```

- The Admin Portal: It is preferred to have an internal node dedicated to server administration and to disable the Admin Portal on the front-end nodes especially when the HTTP services for WebApps are exposed on the Internet. If the Admin node uses the common session manager, it will be able to inform all the other nodes of an LDAP configuration change immediately (ex. OpenOTP setting update).
- The WebApps: They can be deployed on the internal network or on the public network of the company (i.e. The DMZ) to be used by the users from the Internet.
- The web services: It is preferred not to allow public access to SOAP services and RADIUS services. You should enable connections from the local client applications only. And you should allow remote client accesses only through secure connectivity networks such as IPSec transport.

The RCDevs SMSHub web service can be deployed on one node and serve the role of an internal SMS gateway when used with multiple OpenOTP service nodes.

9. Step by Step HA Cluster

9.1 CentOS 7.6 - 4 Nodes

In the following step by step example, we will set up a High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB (TLS) replication and with the RCDevs Directory Server LDAP (TLS) replication.

The HA Cluster will have 4 nodes. The following commands should be run as root. —NODES 1234— means running the commands on every node 1,2,3 and 4.

Warning

Note that you must really do this setup step by step. It will not work if one step is omitted or not following the order.

WebADM requires an accurate system clock, therefore, synchronize the clock. Use `chronyc makestep` for the RCDevs Virtual Appliance and `ntpq -p` if NTP service is used instead.

To simplify the setup can disable the firewall and enable it after having successfully established the replication. Please have a look at RCDevs Communication Ports. It describes the ports and protocols used by RCDevs products between different components. At RCDevs Hardening Guide is an example of the iptables firewall rules for a high availability cluster with 4 nodes.

```
---NODES 1234---
[root@rcdevs1 ~]# cat /etc/system-release
CentOS Linux release 7.6.1810 (Core)
[root@rcdevs1 ~]# yum install chrony
...
Installed:
  chrony.x86_64 0:3.2-2.el7
...
Complete!
[root@rcdevs1 ~]# systemctl start chronyd
[root@rcdevs1 ~]# systemctl enable chronyd
[root@rcdevs1 ~]# chronyc makestep
200 OK
[root@rcdevs1 ~]# systemctl status chronyd -l
● chronyd.service - NTP client/server
   Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled; vendor preset:
enabled)
   Active: active (running) since Thu 2019-02-07 10:28:42 CET; 39s ago
     Docs: man:chronyd(8)
           man:chrony.conf(5)
 Main PID: 16580 (chronyd)
   CGroup: /system.slice/chronyd.service
           └─16580 /usr/sbin/chronyd

Feb 07 10:28:42 rcdevs1.webadm1 systemd[1]: Starting NTP client/server...
Feb 07 10:28:42 rcdevs1.webadm1 chronyd[16580]: chronyd version 3.2 starting (+CMDMON
+NTP +REFCLOCK +RTC +PRIVDROP +SCFILTER +SECHASH +SIGND +ASYNCDNS +IPV6 +DEBUG)
Feb 07 10:28:42 rcdevs1.webadm1 chronyd[16580]: Initial frequency -300.000 ppm
Feb 07 10:28:42 rcdevs1.webadm1 systemd[1]: Started NTP client/server.
Feb 07 10:28:47 rcdevs1.webadm1 chronyd[16580]: Selected source 188.42.54.79
Feb 07 10:29:06 rcdevs1.webadm1 chronyd[16580]: System clock was stepped by 0.000002
seconds
[root@rcdevs1 ~]# systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@rcdevs1 ~]# reboot
```

Be sure that you have a different hostname for each node and put them into `/etc/hosts`. To change the hostname use the command `hostnamectl set-hostname "rcdevs1.webadm1"`.

```
---NODES 1234---
[root@rcdevs1 ~]# hostname
rcdevs1.webadm1
[root@rcdevs1 ~]# vi /etc/hosts
127.0.0.1      localhost
192.168.3.80   rcdevs1.webadm1
192.168.3.81   rcdevs1.webadm2
192.168.3.82   rcdevs1.webadm3
192.168.3.83   rcdevs1.webadm4
[root@rcdevs1 ~]#
```

9.1.1 Directory Server Replication

Use the RCDevs Repository to install the RCDevs Directory Server. The setup script creates the DS system user (slapd), server certificates, filesystem permissions and initializes your LDAP database. During the setup of `/opt/slapd/bin/setup` it will ask to set up an admin password. In this guide, we will use `password` for the LDAP admin password.

```
---NODES 1234---
[root@rcdevs1 ~]# yum install https://www.rcdevs.com/repos/redhat/rcdevs_release-1.0.0-
0.noarch.rpm
...
Installed:
  rcdevs_release.noarch 0:1.0.0-0

Complete!
[root@rcdevs1 ~]# yum install slapd
...
Installed:
  slapd.x86_64 0:1.0.9-0

Complete!
[root@rcdevs1 ~]# /opt/slapd/bin/setup
Checking system architecture...Ok
Enter the server fully qualified host name (FQDN): slapd.local
Is this server a standalone LDAP or a replication peer in an LDAP cluster?
Enter 's' for standalone server or 'r' for a replication peer: s
Enter an admin password: Creating self-signed certificate... Ok
Initializing LDAP data... Ok
Setting file permissions... Ok
Starting LDAP Directory... Ok
Setting Admin password... Ok
Do you want LDAP Directory to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register LDAP Directory logrotate script (y/n)? y
Adding logrotate script... Ok
Do you want to register LDAP Directory DB backup script (y/n)? y
Adding DB backup script... Ok
LDAP Directory has successfully been setup.
[root@rcdevs1 ~]#
```

9.1.1.1 Adjust slapd.conf

With RCDevs Directory Server and more generally with OpenLDAP, the replication uses the syncprov overlay. The recommended configuration is a Master-Master Mirror. On the —NODE 1—, edit the `/opt/slapd/conf/slapd.conf` file. Uncomment the replication block, configure it as follows and restart the slapd service.

```
---NODE 1---
[root@rcdevs1 ~]# vi /opt/slapd/conf/slapd.conf
serverID 1
syncrepl rid=001
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on
[root@rcdevs1 ~]# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
[root@rcdevs1 ~]#
```

Setup the RCDevs Directory Server for —NODE 234—.

```
---NODE 2---
[root@rcdevs2 ~]# vi /opt/slapd/conf/slapd.conf
serverID 2
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on
[root@rcdevs2 ~]# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
[root@rcdevs2 ~]#
```

```
---NODE 3---
[root@rcdevs3 ~]# vi /opt/slapd/conf/slapd.conf
serverID 3
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on
[root@rcdevs3 ~]# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
[root@rcdevs3 ~]#
```

```
---NODE 4---
[root@rcdevs4 ~]# vi /opt/slapd/conf/slapd.conf
serverID 4
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on
[root@rcdevs4 ~]# /opt/slapd/bin/slapd restart
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
[root@rcdevs4 ~]#
```

9.1.2 MariaDB Replication

Let’s install MariaDB. After having installed MySQL/MariaDB, please run the script called `mysql_secure_installation`. It will ask you to change the root password, remove the ability for anyone to log into MySQL by default, disable logging in remotely with the administrator account and remove some test databases that are insecure.

```
---NODES 1234---
[root@rcdevs1 ~]# yum install mariadb-server
...
Installed:
  mariadb-server.x86_64 1:5.5.60-1.el7_5
...
Complete!
[root@rcdevs1 ~]# systemctl start mariadb
[root@rcdevs1 ~]# systemctl enable mariadb
Created symlink from /etc/systemd/system/multi-user.target.wants/mariadb.service to
/usr/lib/systemd/system/mariadb.service.
[root@rcdevs1 ~]# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n]
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n]
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n]
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
```

```
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n]
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n]
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[root@rcdevs1 ~]#
```

9.1.2.1 Adjust server.cnf

Let’s setup the MULTI-MASTER MariaDB replication. First edit the MariaDB configuration file `/etc/my.cnf.d/server.cnf`.

Note

For CentOS 8, the MariaDB configuration file is located at `/etc/my.cnf.d/mariadb-server.cnf`.

```
---NODE 1---
[root@rcdevs1 ~]# vi /etc/my.cnf.d/server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.80
# Note: Set "server-id" to 1 for Node 1.
server-id       = 1
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 1 for Node 1.
auto-increment-offset = 1
replicate-do-db = webadm
log_bin         = mariadb-bin
log-basename    = mariadb
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10

# this is only for embedded server
[embedded]

# This group is only read by MariaDB-5.5 servers.
# If you use the same .cnf file for MariaDB of different versions,
# use this group for options that older servers don't understand
[mysqld-5.5]

# These two groups are only read by MariaDB servers, not by MySQL.
# If you use the same .cnf file for MySQL and MariaDB,
# you can put MariaDB-only options here
[mariadb]

[mariadb-5.5]
[root@rcdevs1 ~]#
```

```
---NODE 2---
[root@rcdevs2 ~]# vi /etc/my.cnf.d/server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.81
# Note: Set "server-id" to 2 for Node 2.
server-id       = 2
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 2 for Node 2.
auto-increment-offset = 2
replicate-do-db = webadm
log_bin         = mariadb-bin
log-basename    = mariadb
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10

# this is only for embedded server
[embedded]

# This group is only read by MariaDB-5.5 servers.
# If you use the same .cnf file for MariaDB of different versions,
# use this group for options that older servers don't understand
[mysqld-5.5]

# These two groups are only read by MariaDB servers, not by MySQL.
# If you use the same .cnf file for MySQL and MariaDB,
# you can put MariaDB-only options here
[mariadb]

[mariadb-5.5]
[root@rcdevs2 ~]#
```

```
---NODE 3---
[root@rcdevs3 ~]# vi /etc/my.cnf.d/server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.82
# Note: Set "server-id" to 3 for Node 3.
server-id       = 3
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 3 for Node 3.
auto-increment-offset = 3
replicate-do-db = webadm
log_bin         = mariadb-bin
log-basename    = mariadb
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10

# this is only for embedded server
[embedded]

# This group is only read by MariaDB-5.5 servers.
# If you use the same .cnf file for MariaDB of different versions,
# use this group for options that older servers don't understand
[mysqld-5.5]

# These two groups are only read by MariaDB servers, not by MySQL.
# If you use the same .cnf file for MySQL and MariaDB,
# you can put MariaDB-only options here
[mariadb]

[mariadb-5.5]
[root@rcdevs3 ~]#
```

```
---NODE 4---
[root@rcdevs4 ~]# vi /etc/my.cnf.d/server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.83
# Note: Set "server-id" to 4 for Node 4.
server-id       = 4
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 4 for Node 4.
auto-increment-offset = 4
replicate-do-db = webadm
log_bin         = mariadb-bin
log-basename    = mariadb
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10

# this is only for embedded server
[embedded]

# This group is only read by MariaDB-5.5 servers.
# If you use the same .cnf file for MariaDB of different versions,
# use this group for options that older servers don't understand
[mysqld-5.5]

# These two groups are only read by MariaDB servers, not by MySQL.
# If you use the same .cnf file for MySQL and MariaDB,
# you can put MariaDB-only options here
[mariadb]

[mariadb-5.5]
[root@rcdevs4 ~]#
```

Restart the MariaDB service and check its status.

```
---NODES 1234---
[root@rcdevs1 ~]# systemctl restart mariadb
[root@rcdevs1 ~]# systemctl status mariadb
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset:
disabled)
   Active: active (running) since Thu 2019-02-07 11:49:52 CET; 27s ago
  Process: 7603 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID (code=exited,
status=0/SUCCESS)
  Process: 7571 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exited,
status=0/SUCCESS)
 Main PID: 7602 (mysqld_safe)
   CGroup: /system.slice/mariadb.service
           ├─7602 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─7908 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --plugin-
dir=/usr/lib64/mysql/plugin --log-error=/var/log/mariadb/mariadb.log --pid-
file=/var/run/mariadb/mariadb.pid --socket=/var/lib/mysql/mysql.sock

Feb 07 11:49:50 rcdevs1.webadm1 systemd[1]: Starting MariaDB database server...
Feb 07 11:49:50 rcdevs1.webadm1 mariadb-prepare-db-dir[7571]: Database MariaDB is
probably initialized in /var/lib/mysql already, nothing is done.
Feb 07 11:49:50 rcdevs1.webadm1 mariadb-prepare-db-dir[7571]: If this is not the case,
make sure the /var/lib/mysql is empty before running mariadb-prepare-db-dir.
Feb 07 11:49:50 rcdevs1.webadm1 mysqld_safe[7602]: 190207 11:49:50 mysqld_safe Logging
to '/var/log/mariadb/mariadb.log'.
Feb 07 11:49:50 rcdevs1.webadm1 mysqld_safe[7602]: 190207 11:49:50 mysqld_safe Starting
mysqld daemon with databases from /var/lib/mysql
Feb 07 11:49:52 rcdevs1.webadm1 systemd[1]: Started MariaDB database server.
[root@rcdevs1 ~]# yum install net-tools
...
Installed:
  net-tools.x86_64 0:2.0-0.24.20131004git.el7

Complete!
[root@rcdevs1 ~]# netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 192.168.3.80:3306       0.0.0.0:*               LISTEN
7908/mysqld
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
6567/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
6812/master
tcp        0      0 0.0.0.0:636             0.0.0.0:*               LISTEN
6755/rcdevs-slapd
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN
6755/rcdevs-slapd
tcp6       0      0 :::22                   :::*                    LISTEN
6567/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN
```

```
6812/master
udp        0      0 127.0.0.1:323           0.0.0.0:*
6250/chronyd
udp6       0      0 ::1:323                 :::*
6250/chronyd
[root@rcdevs1 ~]#
```

9.1.2.2 Database Replication

WebADM uses a database to store audit logs and localized messages. Application configurations, users and their metadata are directly stored in LDAP rather than in the databases. You must create a webadm database on your SQL server and a webadm user with password webadm, having full permissions on that database.

Let's log in to MariaDB as the root user. Create the webadm user and grant privileges on replication.

```
---NODES 1234---
[root@rcdevs1 ~]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 2
Server version: 5.5.60-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE webadm;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> GRANT USAGE ON webadm.* to 'webadm'@'localhost' identified by
'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'localhost';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.80' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.81' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.82' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.83' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.80';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.81';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.82';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.83';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.80';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.81';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.82';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.83';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> STOP SLAVE;
Query OK, 0 rows affected, 1 warning (0.00 sec)

MariaDB [(none)]>
```

```
---NODE 1234---
MariaDB [(none)]> SHOW MASTER STATUS;
+--------------------+----------+--------------+------------------+
| File               | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+--------------------+----------+--------------+------------------+
| mariadb-bin.000001 |     2215 | webadm       |                  |
+--------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

MariaDB [(none)]>
```

Warning

The output of `SHOW MASTER STATUS` will reveal the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number.

Let's start with the —NODE 2— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 1—.

```
---NODE 2---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.80', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000001',
MASTER_LOG_POS = 2215;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>
```

Continue with the —NODE 3— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 2—.

```
---NODE 3---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.81', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000001',
MASTER_LOG_POS = 2215;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>
```

Continue with the —NODE 4— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 3—.

```
---NODE 4---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.82', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000001',
MASTER_LOG_POS = 2215;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>
```

At last the —NODE 1— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 4—.

```
---NODE 1---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.83', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000001',
MASTER_LOG_POS = 2215;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>

---NODE 1234---
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

9.1.2.3 Verify Replication Status

```
---NODE 1---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.83
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000001
          Read_Master_Log_Pos: 2215
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2215
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 4
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 2---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.80
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000001
          Read_Master_Log_Pos: 2215
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2215
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 3---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.81
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000001
          Read_Master_Log_Pos: 2215
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2215
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
1 row in set (0.01 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 4---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.82
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000001
          Read_Master_Log_Pos: 2215
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2215
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 3
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

9.1.3 WebADM HA Cluster

Use the RCDevs Repository to install WebADM with all WebApps and Services.

```
---NODES 1234---
[root@rcdevs1 ~]# yum install webadm_all_in_one
...
Installed:
  webadm_all_in_one.noarch 0:1.0.0-0

Dependency Installed:
  openid.noarch 0:1.3.0-1   openotp.noarch 0:1.4.2-1  opensso.noarch 0:1.0.8-0
  pwreset.noarch 0:1.0.12-1 selfdesk.noarch 0:1.1.8-1 selfreg.noarch 0:1.1.8-0
  smshub.noarch 0:1.1.2-0   spankey.noarch 0:2.0.2-2  tiqr.noarch 0:1.2.5-3
  webadm.x86_64 0:1.6.9-3

Complete!
[root@rcdevs1 ~]#
```

Run the WebADM setup script on —NODE 1—. It initializes the WebADM PKI, etc…

```
---NODE 1---
[root@rcdevs1 ~]# /opt/webadm/bin/setup
Checking system architecture...Ok
Setup WebADM as master server or slave (secondary server in a cluster) (m/s)? m
WebADM proposes 3 default configuration templates:
  1) Default configuration (Novell, eDirectory, Oracle, OpenLDAP)
  2) Active Directory with schema extention (preferred with AD)
  3) Active Directory without schema extention
Choose a template number or press enter for default: 1
Enter the server fully qualified host name (FQDN): webadm.local
Enter your organization name: RCDevs
Generating CA private key... Ok
Creating CA certificate... Ok
Generating SSL private key... Ok
Creating SSL certificate request... Ok
Signing SSL certificate with CA... Ok
Adding CA certificate to the local trust list... Ok
Setting file permissions... Ok
Adding system user to dialout group... Ok
Do you want WebADM to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register WebADM logrotate script (y/n)? y
Adding logrotate scripts... Ok
Do you want to generate a new secret key in webadm.conf (y/n)? y
Generating secret key string... Ok
WebADM has successfully been setup.
[root@rcdevs1 ~]#
```

9.1.3.1 Enterprise License

⚠ Warning

Any high availability and clustering feature require an RCDevs Enterprise license. Without a valid license file, the HA and cluster features are automatically disabled.

Copy your Enterprise License into the `/opt/webadm/conf` folder.

```
---NODE 1---
[root@rcdevs1 ~]# cp license.key /opt/webadm/conf
[root@rcdevs1 ~]#
```

9.1.3.2 Adjust servers.xml

Edit on —NODE 1— the `/opt/webadm/conf/servers.xml` file. Adjust the LDAP Server, SQL Server, Session Server, and PKI Server parameters.

```
---NODE 1---
[root@rcdevs1 ~]# vi /opt/webadm/conf/servers.xml
<?xml version="1.0" encoding="UTF-8" ?>

<Servers>

<!--
******************************************
***  WebADM Remote Server Connections  ***
******************************************

You can configure multiple instances for each of the following servers.
At login, WebADM will try to connect the configured servers in the same
order they appear in this file and uses the first one it successfully
establishes the connection to. If the server connection goes down, it
will automatically failover to the next configured server.

At least one LDAP server is required to run WebADM.
Supported servers: OpenLDAP, Active Directory, Novell eDirectory, 389.

Allowed LDAP parameters are:
 - name: server friendly name
 - host: server hostname or IP address
 - port: LDAP port number
   default and TLS: 389
   default SSL: 636
 - encryption: connection type
   allowed type are NONE, SSL and TLS
   default: 'NONE'
 - ca_cert: Trusted CA for SSL and TLS
 - cert_file: client certificate file
 - cert_key: client certificate key
-->

<LdapServer name="LDAP Server"
 host="192.168.3.80"
 port="389"
 encryption="TLS"
 ca_file="" />
<LdapServer name="LDAP Server 2"
 host="192.168.3.81"
 port="389"
 encryption="TLS"
 ca_file="" />
<LdapServer name="LDAP Server3"
 host="192.168.3.82"
 port="389"
 encryption="TLS"
 ca_file="" />
<LdapServer name="LDAP Server 4"
 host="192.168.3.83"
 port="389"
 encryption="TLS"
 ca_file="" />

<!--
SQL servers are used for logs; message localizations and inventories.
Supported servers: MySQL5, MySQL8, PostgreSQL, MSSQL, Sybase, Oracle, SQLite.

Allowed LDAP parameters are:
 - type: MySQL5, MySQL8, MariaDB, PostgreSQL, MSSQL, Sybase, Oracle or SQLite.
 - name: server friendly name
 - host: server hostname or IP address
 - port: SQL port number (depends on server type)
 - user: database user
 - password: database password
 - database: database name
 - tnsname: Oracle TNS name (Oracle only)

With SQLite, only the 'database' must be set and other parameters are
ignored. The database is the full path to an SQLite DB file where WebADM
has full write access.

With Oracle, you can optionally use TNS names. If the 'tnsname' is set
then the 'host' and 'port' parameters are ignored and a tnsnames.ora
```

```
file must exist under the conf/ directory.
-->

<SqlServer name="SQL Server"
 type="MySQL8"
 host="192.168.3.80"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 2"
 type="MySQL8"
 host="192.168.3.81"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 3"
 type="MySQL8"
 host="192.168.3.82"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 4"
 type="MySQL8"
 host="192.168.3.83"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />

<!--
A session server is required for web services using sessions
such as OpenOTP. You can specify one or more SQL servers here.
The session server is included in WebADM. So you can keep the
default settings here.
-->

<SessionServer name="Session Server"
 host="192.168.3.80"
 port="4000"
 secret="" />
<SessionServer name="Session Server 2"
 host="192.168.3.81"
 port="4000"
 secret="" />
<SessionServer name="Session Server 3"
 host="192.168.3.82"
 port="4000"
 secret="" />
<SessionServer name="Session Server 4"
 host="192.168.3.83"
```

```
  host="192.168.3.83"
  port="4000"
  secret="" />

 <!--
 A PKI server (or CA) is required for signing user certificates.
 The RSign PKI server is included in WebADM. So you can keep the
 default settings here.
 -->

 <PkiServer name="PKI Server"
  host="192.168.3.80"
  port="5000"
  secret="secret"
  ca_file="" />
 ...
 [root@rcdevs1 ~]#
```

9.1.3.3 Adjust rsignd.conf

On the —NODE 1—, allow client PKI connections to the Rsignd PKI server. This is done by adding the client configuration blocks for the other nodes in the `/opt/webadm/conf/rsignd.conf` file. The password/secret for the PKI server will be in this case `secret`.

```
---NODE 1---
[root@rcdevs1 ~]# vi /opt/webadm/conf/rsignd.conf
#
# WebADM PKI Server Configuration
#
...
#
# Client sections
#
# Declare here the Rsign clients with IP addresses or hostnames.
# In cluster mode, the client WebADM server(s) must be defined here!

client {
 hostname 192.168.3.80
 secret secret
}
client {
 hostname 192.168.3.81
 secret secret
}
client {
 hostname 192.168.3.82
 secret secret
}
client {
 hostname 192.168.3.83
 secret secret
}

[root@rcdevs1 ~]#
```

9.1.3.4 Start WebADM

Start WebADM and login for the 1st time into the graphical setup.

```
---NODE 1---
[root@rcdevs1 ~]# /opt/webadm/bin/webadm start
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM PKI server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server (192.168.3.80)

Checking LDAP proxy user access... ERROR
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm master)
[root@rcdevs1 ~]#
```

Now we connect to the WebADM Admin Portal on `https://192.168.3.80`.

Important

If you use RCDevs Directory Server, the admin DN is *cn=admin,o=root*. The default password is *password*.

WebADM Admin Portal Login (RCDevs Directory Server)

The Setup button will appear on the home page when you enter the WebADM Admin Portal.

Now click on the `Create/Update SQL database tables`, `Create WebADM proxy user`, `Setup permissions` and `Create default containers and objects` buttons to complete the setup.

We will be able to use the `admin` user after the first configuration.

9.1.3.5 Setup WebADM Slaves

The WebADM setup script must be run using the `slave` parameter with the command `/opt/webadm/bin/setup slave` on —NODE 234—. The master PKI server address is in this case `192.168.3.80`. The master PKI server secret is `secret` as defined before in 9.2.3.3 Adjust rsignd.conf.

```
---NODE 234---
[root@rcdevs2 ~]# /opt/webadm/bin/setup slave
Checking system architecture...Ok
WebADM proposes 3 default configuration templates:
  1) Default configuration (Novell, eDirectory, Oracle, OpenLDAP)
  2) Active Directory with schema extention (preferred with AD)
  3) Active Directory without schema extention
Choose a template number or press enter for default: 1
Enter the server fully qualified host name (FQDN): webadm.local
Enter the master PKI server address: 192.168.3.80
Enter the master PKI server port (enter for default):
Enter the master PKI server secret: secret
Testing PKI server conection... Ok
Retrieving PKI CA certificate...Ok
Reading organization name from CA certificate...
Generating SSL private key... Ok
Creating SSL certificate request... Ok
Signing SSL certificate with PKI server... Ok
Adding CA certificate to the local trust list... Ok
Setting file permissions... Ok
Adding system user to dialout group... Ok
Do you want WebADM to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register WebADM logrotate script (y/n)? y
Adding logrotate scripts... Ok
WebADM has successfully been setup.
[root@rcdevs2 ~]#
```

9.1.3.6 Copy Setup Files to Slaves

Finally, save the WebADM configuration and copy it to the other —NODE 234—. At last, start WebADM on the other —NODE 234—. Now the High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB replication and with the RCDevs Directory Server LDAP

(TLS) replication is running.

```
---NODE 1---
[root@rcdevs1 ~]# cd /
[root@rcdevs1 /]# tar czvf /tmp/webadm_conf.tar.gz /opt/webadm/conf
tar: Removing leading `/' from member names
/opt/webadm/conf/
/opt/webadm/conf/objects.xml
/opt/webadm/conf/objects.xml.default
/opt/webadm/conf/rsignd.conf.default
/opt/webadm/conf/servers.xml.default
/opt/webadm/conf/webadm.conf
/opt/webadm/conf/webadm.conf.default
/opt/webadm/conf/webadm.conf.bak
/opt/webadm/conf/objects.xml.bak
/opt/webadm/conf/rsignd.conf.bak
/opt/webadm/conf/servers.xml.bak
/opt/webadm/conf/license.key
/opt/webadm/conf/servers.xml
/opt/webadm/conf/rsignd.conf
[root@rcdevs1 /]# scp /tmp/webadm_conf.tar.gz root@192.168.3.81:/tmp/
root@192.168.3.81's password:
webadm_conf.tar.gz                            100%   17KB   7.7MB/s   00:00
[root@rcdevs1 /]# scp /tmp/webadm_conf.tar.gz root@192.168.3.82:/tmp/
root@192.168.3.82's password:
webadm_conf.tar.gz                            100%   17KB   7.3MB/s   00:00
[root@rcdevs1 /]# scp /tmp/webadm_conf.tar.gz root@192.168.3.83:/tmp/
root@192.168.3.83's password:
webadm_conf.tar.gz                            100%   17KB   7.5MB/s   00:00
[root@rcdevs1 /]# rm /tmp/webadm_conf.tar.gz
[root@rcdevs1 /]#
```

```
---NODE 234---
[root@rcdevs2 ~]# cp /tmp/webadm_conf.tar.gz /
[root@rcdevs2 ~]# cd /
[root@rcdevs2 /]# tar xzvf /tmp/webadm_conf.tar.gz
opt/webadm/conf/
opt/webadm/conf/objects.xml
opt/webadm/conf/objects.xml.default
opt/webadm/conf/rsignd.conf.default
opt/webadm/conf/servers.xml.default
opt/webadm/conf/webadm.conf
opt/webadm/conf/webadm.conf.default
opt/webadm/conf/webadm.conf.bak
opt/webadm/conf/objects.xml.bak
opt/webadm/conf/rsignd.conf.bak
opt/webadm/conf/servers.xml.bak
opt/webadm/conf/license.key
opt/webadm/conf/servers.xml
opt/webadm/conf/rsignd.conf
[root@rcdevs2 /]# rm webadm_conf.tar.gz
[root@rcdevs2 /]# /opt/webadm/bin/webadm start
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server (192.168.3.80)

Checking LDAP proxy user access... Ok
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm slave)
Session replication status: Active (0.0009 sec)
[root@rcdevs2 /]#
```

Now verify if the `Network Service Statuses` under the `Admin` tab are online. That’s it, successfully set up a High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB replication and with the RCDevs Directory Server LDAP (TLS) replication.

9.1.4 MariaDB TLS Replication

Let's enable TLS for the MULTI-MASTER MariaDB replication.

```
---NODE 1234---
[root@rcdevs1 /]# mkdir /var/lib/mysql/ssl/
[root@rcdevs1 /]# cd /var/lib/mysql/ssl/
[root@rcdevs1 ssl]#
```

9.1.4.1 Export Certificates

Instead of using your own certificates, one can issue and export SSL Certificate over WebADM GUI under the Admin tab.

Click on `Download WebADM CA Certificate` to download it.

Now click on `Issue Server or Client SSL Certificate`, add an `FQDN: mariadbserver` and select `Server`.

Download the Key and Cert File.

Click on `Issue Server or Client SSL Certificate`, add an `FQDN: mariadbclient` and select `Client`.

Download Cert & Key File.

Finally, copy the following certificates `ca.crt` and `mariadbclient.crt` in to the WebADM configuration folder `/opt/webadm/conf/`.

9.1.4.2 Copy Certificates to all the Nodes

Copy the following certificates `ca.crt`, `mariadbserver.crt` and `mariadbserver.key` to all the nodes — NODE 1234—.

```
administor:Downloads$ ssh root@192.168.3.80 mkdir /tmp/ssl/
root@192.168.3.80's password:
administor:Downloads$ ssh root@192.168.3.81 mkdir /tmp/ssl/
root@192.168.3.81's password:
administor:Downloads$ ssh root@192.168.3.82 mkdir /tmp/ssl/
root@192.168.3.82's password:
administor:Downloads$ ssh root@192.168.3.83 mkdir /tmp/ssl/
root@192.168.3.83's password:
administor:Downloads$ scp *.pem root@192.168.3.80:/tmp/ssl/
root@192.168.3.80's password:
ca.crt                                  100% 1142     1.7MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$ scp *.pem root@192.168.3.81:/tmp/ssl/
root@192.168.3.81's password:
ca.crt                                  100% 1142     1.6MB/s   00:00
mariadbserver.crt                       100% 1092     1.6MB/s   00:00
mariadbserver.key        100% 1092     1.6MB/s   00:00
administor:Downloads$ scp *.pem root@192.168.3.82:/tmp/ssl/
root@192.168.3.82's password:
ca.crt                                  100% 1142     1.5MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$ scp *.pem root@192.168.3.83:/tmp/ssl/
root@192.168.3.83's password:
ca.crt                                  100% 1142     1.6MB/s   00:00
mariadbserver.crt                       100% 1092     1.4MB/s   00:00
mariadbserver.key                       100% 1092     1.6MB/s   00:00
administor:Downloads$
```

⚠ Warning

Set the owner to mysql and the rights for the MariaDB certificate files.

```
---NODE 1234---
[root@rcdevs1 ssl]# mv /tmp/ssl/* /var/lib/mysql/ssl/
[root@rcdevs1 ssl]# chown mysql:mysql /var/lib/mysql/ssl/
[root@rcdevs1 ssl]# chown mysql:mysql /var/lib/mysql/ssl/*
[root@rcdevs1 ssl]# chmod 640 /var/lib/mysql/ssl/*
[root@rcdevs1 ssl]# rm -r /tmp/ssl/
[root@rcdevs1 ssl]#
```

9.1.4.3 Adjust server.cnf

Edit the MariaDB configuration file `/etc/my.cnf.d/server.cnf` on all the nodes —NODE 1234— to add the path of the certificates, `ssl-ca`, `ssl-cert` and `ssl-key`. Afterwards, restart the MariaDB service.

```
---NODE 1234---
[root@rcdevs1 ssl]# vi /etc/my.cnf.d/server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.80
server-id       = 1
replicate-same-server-id = 0
auto-increment-increment = 4
auto-increment-offset = 1
replicate-do-db = webadm
log_bin         = mariadb-bin
log-basename    = mariadb
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10
ssl-ca=/var/lib/mysql/ssl/ca.crt
ssl-cert=/var/lib/mysql/ssl/mariadbserver.crt
ssl-key=/var/lib/mysql/ssl/mariadbserver.key
...

[root@rcdevs1 ssl]# systemctl restart mariadb
```

```
[root@rcdevs1 ssl]# systemctl restart mariadb
[root@rcdevs1 ssl]# systemctl status mariadb -l
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset:
disabled)
   Active: active (running) since Thu 2019-02-07 13:39:53 CET; 3s ago
  Process: 24381 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID (code=exited,
status=0/SUCCESS)
  Process: 24349 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exited,
status=0/SUCCESS)
 Main PID: 24380 (mysqld_safe)
   CGroup: /system.slice/mariadb.service
           ├─24380 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─24722 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --
plugin-dir=/usr/lib64/mysql/plugin --log-error=/var/log/mariadb/mariadb.log --pid-
file=/var/run/mariadb/mariadb.pid --socket=/var/lib/mysql/mysql.sock

Feb 07 13:39:51 rcdevs1.webadm1 systemd[1]: Starting MariaDB database server...
Feb 07 13:39:51 rcdevs1.webadm1 mariadb-prepare-db-dir[24349]: Database MariaDB is
probably initialized in /var/lib/mysql already, nothing is done.
Feb 07 13:39:51 rcdevs1.webadm1 mariadb-prepare-db-dir[24349]: If this is not the case,
make sure the /var/lib/mysql is empty before running mariadb-prepare-db-dir.
Feb 07 13:39:51 rcdevs1.webadm1 mysqld_safe[24380]: 190207 13:39:51 mysqld_safe Logging
to '/var/log/mariadb/mariadb.log'.
Feb 07 13:39:51 rcdevs1.webadm1 mysqld_safe[24380]: 190207 13:39:51 mysqld_safe
Starting mysqld daemon with databases from /var/lib/mysql
Feb 07 13:39:53 rcdevs1.webadm1 systemd[1]: Started MariaDB database server.
[root@rcdevs1 ssl]# netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN
8171/webadm-rsignd
tcp        0      0 192.168.3.80:3306       0.0.0.0:*               LISTEN
24722/mysqld
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN
8217/webadm-httpd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
8217/webadm-httpd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
6567/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
6812/master
tcp        0      0 0.0.0.0:8443            0.0.0.0:*               LISTEN
8217/webadm-httpd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
8217/webadm-httpd
tcp        0      0 0.0.0.0:636             0.0.0.0:*               LISTEN
6755/rcdevs-slapd
tcp        0      0 0.0.0.0:4000            0.0.0.0:*               LISTEN
8164/webadm-session
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN
6755/rcdevs-slapd
```

```
6755/rcdevs-slapd
tcp6       0      0 :::22                   :::*                    LISTEN
6567/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN
6812/master
tcp6       0      0 :::4000                 :::*                    LISTEN
8164/webadm-session
udp        0      0 127.0.0.1:323           0.0.0.0:*
6250/chronyd
udp6       0      0 ::1:323                 :::*
6250/chronyd
[root@rcdevs1 ssl]#
```

9.1.4.4 Enable SSL/TLS

Log in to MariaDB as the root user and enable the SSL/TLS.

```
---NODE 1234---
[root@rcdevs1 ssl]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 102
Server version: 5.5.60-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'localhost' REQUIRE
X509;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.80' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.81' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.82' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.83' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'localhost' REQUIRE SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.80' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.81' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.82' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.83' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> STOP SLAVE;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

```
---NODE 1234---
MariaDB [(none)]> SHOW MASTER STATUS;
+--------------------+----------+--------------+------------------+
| File               | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+--------------------+----------+--------------+------------------+
| mariadb-bin.000002 |     1739 | webadm       |                  |
+--------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

MariaDB [(none)]>
```

Warning

The output of `SHOW MASTER STATUS` will reveal the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number.

Let's start with the —NODE 2— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 1—.

```
---NODE 2---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.80', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000002',
MASTER_LOG_POS = 1739, MASTER_SSL=1;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

Continue with the —NODE 3— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 2—.

```
---NODE 3---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.81', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000002',
MASTER_LOG_POS = 1739, MASTER_SSL=1;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>
```

Continue with the —NODE 4— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 3—.

```
---NODE 4---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.82', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000002',
MASTER_LOG_POS = 1739, MASTER_SSL=1;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

At last the —NODE 1— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 4—.

```
---NODE 1---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.83', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mariadb-bin.000002',
MASTER_LOG_POS = 1739, MASTER_SSL=1;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>

---NODE 1234---
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

9.1.4.5 Verify TLS Status

Verify MariaDB TLS as follows:

```
---NODE 1---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.83
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000002
          Read_Master_Log_Pos: 1739
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 1739
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 4
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 2---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.80
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000002
          Read_Master_Log_Pos: 1739
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 1739
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 3---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.81
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000002
          Read_Master_Log_Pos: 1739
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 1739
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 4---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.82
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000002
          Read_Master_Log_Pos: 1739
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 1739
              Relay_Log_Space: 821
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 3
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 1234---
MariaDB [(none)]> SHOW VARIABLES LIKE '%ssl%';
+---------------+------------------------------------+
| Variable_name | Value                              |
+---------------+------------------------------------+
| have_openssl  | YES                                |
| have_ssl      | YES                                |
| ssl_ca        | /var/lib/mysql/ssl/ca.crt          |
| ssl_capath    |                                    |
| ssl_cert      | /var/lib/mysql/ssl/mariadbserver.crt |
| ssl_cipher    |                                    |
| ssl_key       | /var/lib/mysql/ssl/mariadbserver.key |
+---------------+------------------------------------+
7 rows in set (0.00 sec)

MariaDB [(none)]> status;
--------------
mysql  Ver 15.1 Distrib 5.5.60-MariaDB, for Linux (x86_64) using readline 5.1

Connection id:  4
Current database:
Current user:  root@localhost
SSL:   Cipher in use is DHE-RSA-AES256-GCM-SHA384
Current pager:  stdout
Using outfile:  ''
Using delimiter: ;
Server:   MariaDB
Server version:  5.5.60-MariaDB MariaDB Server
Protocol version: 10
Connection:  Localhost via UNIX socket
Server characterset: latin1
Db     characterset: latin1
Client characterset: latin1
Conn.  characterset: latin1
UNIX socket:  /var/lib/mysql/mysql.sock
Uptime:   4 min 7 sec

Threads: 2  Questions: 15  Slow queries: 0  Opens: 0  Flush tables: 2  Open tables: 26
Queries per second avg: 0.060
--------------
```

9.1.4.6 Adjust servers.xml

Finally, adjust the parameter encryption from `NONE` to `TLS` and add the certificates in the configuration file `/opt/webadm/conf/servers.xml` of all nodes —NODE 1234—. Afterward, restart WebADM to enable TLS for MULTI-MASTER MariaDB replication.

Note

Please use the `MySQL8` instead of the `MariaDB` driver for certificate based authentication.

```
---NODE 1234---
[root@rcdevs1 ssl]# vi /opt/webadm/conf/servers.xml
<SqlServer name="SQL Server"
 type="MySQL8"
 host="192.168.3.80"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 2"
 type="MySQL8"
 host="192.168.3.81"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 3"
 type="MySQL8"
 host="192.168.3.82"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 4"
 type="MySQL8"
 host="192.168.3.83"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>

[root@rcdevs1 ssl]# /opt/webadm/bin/webadm restart
Stopping WebADM HTTP server... Ok
Stopping WebADM Watchd server..... Ok
Stopping WebADM PKI server... Ok
Stopping WebADM Session server... Ok
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM PKI server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server 2 (192.168.3.81)

Checking LDAP proxy user access... Ok
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm slave)
Session replication status: Active (0.0014 sec)
[root@rcdevs1 ssl]#
```

9.1.4.7 Iptables Firewall Rules

At RCDevs Hardening Guide is an example of the iptables firewall rules for a high availability cluster with 4 nodes.

9.2 Ubuntu 18.04 - 4 Nodes

In the following step by step example, we will set up a High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB (TLS) replication and with the RCDevs Directory Server LDAP (TLS) replication.

The HA Cluster will have 4 nodes. The following commands should be run as root. —NODES 1234— means running the commands on every node 1,2,3 and 4.

⚠ Warning

Note that you must really do this setup step by step. It will not work if one step is omitted or not following the order.

WebADM requires an accurate system clock, therefore, synchronize the clock.

```
---NODES 1234---
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-45-generic x86_64)
webadm1@ubuntu18-webadm1:~$ sudo su
[sudo] password for webadm1:
root@ubuntu18-webadm1:/home/webadm1# systemctl restart systemd-timesyncd
root@ubuntu18-webadm1:/home/webadm1#
```

Be sure that you have a different hostname for each node and put them into `/etc/hosts`. To change the hostname use the command `hostnamectl set-hostname "ubuntu18-webadm1"`.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# hostname
ubuntu18-webadm1
root@ubuntu18-webadm1:/home/webadm1# vi /etc/hosts
127.0.0.1     localhost
192.168.3.80  ubuntu18-webadm1
192.168.3.81  ubuntu18-webadm2
192.168.3.82  ubuntu18-webadm3
192.168.3.83  ubuntu18-webadm4
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.1 Directory Server Replication

Use the RCDevs Repository to install the RCDevs Directory Server. The setup script creates the DS system user (slapd), server certificates, filesystem permissions and initializes your LDAP database. During the setup of `/opt/slapd/bin/setup` it will ask to set up an admin password. In this guide, we will use `password` for the LDAP admin password.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# wget https://www.rcdevs.com/repos/debian/rcdevs-
release_1.0.1-0_all.deb
--2019-02-06 09:42:56--  https://www.rcdevs.com/repos/debian/rcdevs-release_1.0.1-
0_all.deb
Resolving www.rcdevs.com (www.rcdevs.com)... 78.141.172.203
Connecting to www.rcdevs.com (www.rcdevs.com)|78.141.172.203|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2526 (2.5K)
Saving to: ‘rcdevs-release_1.0.1-0_all.deb’
```

```
rcdevs-release_1.0. 100%[===================>]   2.47K  --.-KB/s    in 0s

2019-02-06 09:42:56 (86.2 MB/s) - ‘rcdevs-release_1.0.1-0_all.deb’ saved [2526/2526]

root@ubuntu18-webadm1:/home/webadm1# apt-get install ./rcdevs-release_1.0.1-0_all.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'rcdevs-release' instead of './rcdevs-release_1.0.1-0_all.deb'
The following NEW packages will be installed:
  rcdevs-release
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/2,526 B of archives.
After this operation, 1,024 B of additional disk space will be used.
Get:1 /home/webadm1/rcdevs-release_1.0.1-0_all.deb rcdevs-release all 1.0.0-0 [2,526 B]
Selecting previously unselected package rcdevs-release.
(Reading database ... 102328 files and directories currently installed.)
Preparing to unpack .../rcdevs-release_1.0.1-0_all.deb ...
Unpacking rcdevs-release (1.0.0-0) ...
Setting up rcdevs-release (1.0.0-0) ...
root@ubuntu18-webadm1:/home/webadm1# apt-get update
Get:1 http://rcdevs.com/repos/debian ./ InRelease [1,074 B]
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:5 http://archive.ubuntu.com/ubuntu bionic-security InRelease
Get:6 http://rcdevs.com/repos/debian ./ Packages [12.8 kB]
Fetched 13.9 kB in 1s (26.6 kB/s)
Reading package lists... Done
root@ubuntu18-webadm1:/home/webadm1# apt-get install rcdevs-slapd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  rcdevs-slapd
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,988 kB of archives.
After this operation, 19.7 MB of additional disk space will be used.
Get:1 http://rcdevs.com/repos/debian ./ rcdevs-slapd 1.0.9-0 [6,988 kB]
Fetched 6,988 kB in 0s (48.2 MB/s)
Selecting previously unselected package rcdevs-slapd.
(Reading database ... 102332 files and directories currently installed.)
Preparing to unpack .../rcdevs-slapd_1.0.9-0_amd64.deb ...
Unpacking rcdevs-slapd (1.0.9-0) ...
Setting up rcdevs-slapd (1.0.9-0) ...
Directory Server needs to be configured.
Please run /opt/slapd/bin/setup.
root@ubuntu18-webadm1:/home/webadm1# /opt/slapd/bin/setup
Checking system architecture...Ok
Enter the server fully qualified host name (FQDN): slapd.local
Is this server a standalone LDAP or a replication peer in an LDAP cluster?
Enter 's' for standalone server or 'r' for a replication peer: s
Enter an admin password: Creating self-signed certificate... Ok
Initializing LDAP data... Ok
Setting file permissions... Ok
Starting LDAP Directory... Ok
Setting Admin password... Ok
Do you want LDAP Directory to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register LDAP Directory logrotate script (y/n)? y
Adding logrotate script... Ok
Do you want to register LDAP Directory DB backup script (y/n)? y
Adding DB backup script... Ok
LDAP Directory has successfully been setup.
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.1.1 Adjust slapd.conf

With RCDevs Directory Server and more generally with OpenLDAP, the replication uses the syncprov overlay. The recommended configuration is a Master-Master Mirror. On the —NODE 1—, edit the `/opt/slapd/conf/slapd.conf` file. Uncomment the replication block, configure it as follows and restart the slapd service.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# vi /opt/slapd/conf/slapd.conf
...
# The rest of the configuration is for LDAP clustering (mirror replication).
# Uncomment all the following lines to setup your LDAP server in mirror mode
# replication with remote server ldap2.example.com.
# For more details see http://www.openldap.org/doc/admin23/syncrepl.html

serverID 1
syncrepl rid=001
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on

root@ubuntu18-webadm1:/home/webadm1# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
root@ubuntu18-webadm1:/home/webadm1#
```

Setup the RCDevs Directory Server for —NODE 234—.

```
---NODE 2---
root@ubuntu18-webadm2:/home/webadm2# vi /opt/slapd/conf/slapd.conf
...
# The rest of the configuration is for LDAP clustering (mirror replication).
# Uncomment all the following lines to setup your LDAP server in mirror mode
# replication with remote server ldap2.example.com.
# For more details see http://www.openldap.org/doc/admin23/syncrepl.html

serverID 2
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on

root@ubuntu18-webadm2:/home/webadm2# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
root@ubuntu18-webadm2:/home/webadm2#
```

```
---NODE 3---
root@rcdevs3-webadm3:/home/webadm3# vi /opt/slapd/conf/slapd.conf
...
# The rest of the configuration is for LDAP clustering (mirror replication).
# Uncomment all the following lines to setup your LDAP server in mirror mode
# replication with remote server ldap2.example.com.
# For more details see http://www.openldap.org/doc/admin23/syncrepl.html

serverID 3
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.83
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on

root@rcdevs3-webadm3:/home/webadm3# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
root@rcdevs3-webadm3:/home/webadm3#
```

```
---NODE 4---
root@rcdevs4-webadm4:/home/webadm4# vi /opt/slapd/conf/slapd.conf
...
# The rest of the configuration is for LDAP clustering (mirror replication).
# Uncomment all the following lines to setup your LDAP server in mirror mode
# replication with remote server ldap2.example.com.
# For more details see http://www.openldap.org/doc/admin23/syncrepl.html

serverID 4
syncrepl rid=001
 provider=ldap://192.168.3.80
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=002
 provider=ldap://192.168.3.81
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
syncrepl rid=003
 provider=ldap://192.168.3.82
 bindmethod=simple
 binddn="cn=admin,o=root"
 credentials="password"
 starttls=yes
 tls_reqcert=never
 searchbase=""
 schemachecking=on
 type=refreshAndPersist
 retry="10 5 60 +"
mirrormode on

root@rcdevs4-webadm4:/home/webadm4# /opt/slapd/bin/slapd restart
Stopping RCDevs LDAP Directory... Ok
Checking system architecture... Ok
Checking server configuration... Ok
Starting RCDevs LDAP Directory... Ok
root@rcdevs4-webadm4:/home/webadm4#
```

9.2.2 MariaDB Replication

Let's install MariaDB. After having installed MySQL/MariaDB, please run the script called `mysql_secure_installation`. It will ask you to change the root password, remove the ability for anyone to log into MySQL by default, disable logging in remotely with the administrator account and remove some test databases that are insecure.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# apt-get install mariadb-server
...
Setting up mariadb-client-10.1 (1:10.1.34-0ubuntu0.18.04.1) ...
Setting up mariadb-server-10.1 (1:10.1.34-0ubuntu0.18.04.1) ...
...
root@ubuntu18-webadm1:/home/webadm1# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n]
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n]
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
```

```
Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n]
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n]
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n]
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.2.1 Adjust server.cnf

Let's setup the MULTI-MASTER MariaDB replication. First edit the MariaDB configuration file `/etc/mysql/mariadb.conf.d/50-server.cnf`. Therefore add under `[mysqld]` the block from `bind-address` until `relay-log-index`.

⚠ Warning

Note that you must disable the local bind-address for a MULTI-MASTER MariaDB replication with `#bind-address = 127.0.0.1`. You will find it under the `[mysqld]` section.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# vi /etc/mysql/mariadb.conf.d/50-server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.80
# Note: Set "server-id" to 1 for Node 1.
server-id       = 1
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 1 for Node 1.
auto-increment-offset = 1
replicate-do-db = webadm
log_bin         = mysql-bin
log-basename    = mysql
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

#
# * Basic Settings
#
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
lc-messages-dir = /usr/share/mysql
skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
# bind-address          = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size         = 16M
max_allowed_packet      = 16M
thread_stack            = 192K
thread_cache_size       = 8
# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
myisam-recover         = BACKUP
#max_connections        = 100
```

```
#table_cache            = 64
#thread_concurrency     = 10

#
# * Query Cache Configuration
#
query_cache_limit       = 1M
query_cache_size        = 16M

#
# * Logging and Replication
#
# Both location gets rotated by the cronjob.
# Be aware that this log type is a performance killer.
# As of 5.1 you can enable the log at runtime!
#general_log_file        = /var/log/mysql/mysql.log
#general_log             = 1
#
# Error log - should be very few entries.
#
log_error = /var/log/mysql/error.log
#
# Enable the slow query log to see queries with especially long duration
#slow_query_log_file    = /var/log/mysql/mariadb-slow.log
#long_query_time = 10
#log_slow_rate_limit    = 1000
#log_slow_verbosity     = query_plan
#log-queries-not-using-indexes
#
# The following can be used as easy to replay backup logs or for replication.
# note: if you are setting up a replication slave, see README.Debian about
#       other settings you may need to change.
#server-id              = 1
#log_bin                        = /var/log/mysql/mysql-bin.log
expire_logs_days        = 10
max_binlog_size   = 100M
#binlog_do_db           = include_database_name
#binlog_ignore_db       = include_database_name

#
# * InnoDB
#
# InnoDB is enabled by default with a 10MB datafile in /var/lib/mysql/.
# Read the manual for more InnoDB related options. There are many!

#
# * Security Features
#
# Read the manual, too, if you want chroot!
# chroot = /var/lib/mysql/
#
# For generating SSL certificates I recommend the OpenSSL GUI "tinyca".
```

```
#
# ssl-ca=/etc/mysql/cacert.pem
# ssl-cert=/etc/mysql/server-cert.pem
# ssl-key=/etc/mysql/server-key.pem
#
# * Character sets
#
# MySQL/MariaDB default is Latin1, but in Debian we rather default to the full
# utf8 4-byte character set. See also client.cnf
#
character-set-server  = latin1
collation-server      = latin1_swedish_ci

#
# * Unix socket authentication plugin is built-in since 10.0.22-6
#
# Needed so the root database user can authenticate without a password but
# only when running as the unix root user.
#
# Also available for other users if required.
# See https://mariadb.com/kb/en/unix_socket-authentication-plugin/

# this is only for embedded server
[embedded]

# This group is only read by MariaDB servers, not by MySQL.
# If you use the same .cnf file for MySQL and MariaDB,
# you can put MariaDB-only options here
[mariadb]

# This group is only read by MariaDB-10.0 servers.
# If you use the same .cnf file for MariaDB of different versions,
# use this group for options that older servers don't understand
[mariadb-10.0]

root@ubuntu18-webadm1:/home/webadm1#
```

```
---NODE 2---
root@ubuntu18-webadm2:/home/webadm2# vi /etc/mysql/mariadb.conf.d/50-server.cnf
[mysqld]
bind-address    = 192.168.3.81
# Note: Set "server-id" to 2 for Node 2.
server-id       = 2
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 2 for Node 2.
auto-increment-offset = 2
replicate-do-db = webadm
log_bin  = mysql-bin
log-basename = mysql
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

...
#bind-address = 127.0.0.1
...
root@ubuntu18-webadm2:/home/webadm2#
```

```
---NODE 3---
root@rcdevs3-webadm3:/home/webadm3# vi /etc/mysql/mariadb.conf.d/50-server.cnf
[mysqld]
bind-address    = 192.168.3.82
# Note: Set "server-id" to 3 for Node 3.
server-id       = 3
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 3 for Node 3.
auto-increment-offset = 3
replicate-do-db = webadm
log_bin  = mysql-bin
log-basename = mysql
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

...
#bind-address = 127.0.0.1
...
root@rcdevs3-webadm3:/home/webadm3#
```

```
---NODE 4---
root@rcdevs4-webadm4:/home/webadm4# vi /etc/mysql/mariadb.conf.d/50-server.cnf
[mysqld]
bind-address    = 192.168.3.83
# Note: Set "server-id" to 4 for Node 4.
server-id       = 4
replicate-same-server-id = 0
# Note: Set "auto-increment-increment" to 4 because there are 4 Nodes.
auto-increment-increment = 4
# Note: Set "auto-increment-offset" to 4 for Node 4.
auto-increment-offset = 4
replicate-do-db = webadm
log_bin  = mysql-bin
log-basename = mysql
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

...
#bind-address = 127.0.0.1
...
root@rcdevs4-webadm4:/home/webadm4#
```

Restart the MariaDB service and check its status.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# systemctl restart mysql
root@ubuntu18-webadm1:/home/webadm1# systemctl status mysql -l
● mariadb.service - MariaDB 10.1.34 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset:
enabled)
   Active: active (running) since Wed 2019-02-06 10:39:43 UTC; 2min 5s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
  Process: 4074 ExecStartPost=/bin/sh -c systemctl unset-environment
_WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 4070 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
  Process: 3915 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= ||
VAR=`/usr/bin/galera_recovery`; [ $? -eq 0 ]   && systemctl set-environm
  Process: 3906 ExecStartPre=/bin/sh -c systemctl unset-environment
_WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 3884 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d
/var/run/mysqld (code=exited, status=0/SUCCESS)
 Main PID: 4041 (mysqld)
   Status: "Taking your SQL requests now..."
    Tasks: 28 (limit: 2292)
   CGroup: /system.slice/mariadb.service
           └─4041 /usr/sbin/mysqld
```

```
Feb 06 10:39:41 ubuntu18-webadm1 systemd[1]: Starting MariaDB 10.1.34 database
server...
Feb 06 10:39:42 ubuntu18-webadm1 mysqld[4041]: 2019-02-06 10:39:42 139789789052032
[Note] /usr/sbin/mysqld (mysqld 10.1.34-MariaDB-0ubuntu0.18.04.1) starting a
Feb 06 10:39:42 ubuntu18-webadm1 /etc/mysql/debian-start[4087]: Checking for insecure
root accounts.
Feb 06 10:39:42 ubuntu18-webadm1 /etc/mysql/debian-start[4091]: Triggering myisam-
recover for all MyISAM tables and aria-recover for all Aria tables
Feb 06 10:39:43 ubuntu18-webadm1 systemd[1]: Started MariaDB 10.1.34 database server.
root@ubuntu18-webadm1:/home/webadm1# netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN
1528/webadm-rsignd
tcp        0      0 192.168.3.80:3306       0.0.0.0:*               LISTEN
4401/mysqld
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
818/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
1257/sshd
tcp        0      0 0.0.0.0:8443            0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:636             0.0.0.0:*               LISTEN
1313/rcdevs-slapd
tcp        0      0 0.0.0.0:4000            0.0.0.0:*               LISTEN
1462/webadm-session
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN
1313/rcdevs-slapd
tcp6       0      0 :::22                   :::*                    LISTEN
1257/sshd
tcp6       0      0 :::4000                 :::*                    LISTEN
1462/webadm-session
udp        0      0 127.0.0.53:53           0.0.0.0:*
818/systemd-resolve
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.2.2 Database Replication

WebADM uses a database to store audit logs and localized messages. Application configurations, users and their metadata are directly stored in LDAP rather than in the databases. You must create a webadm database on your SQL server and a webadm user with password webadm, having full permissions on that database.

Let’s log in to MariaDB as the root user. Create the webadm user and grant privileges on replication.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.1.34-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE webadm;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> GRANT USAGE ON webadm.* to 'webadm'@'localhost' identified by
'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'localhost';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.80' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.81' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.82' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> CREATE USER 'webadm'@'192.168.3.83' identified by 'webadm';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.80';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.81';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.82';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.83';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.80';
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.81';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.82';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.83';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> STOP SLAVE;
Query OK, 0 rows affected, 1 warning (0.00 sec)

MariaDB [(none)]> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000001 |     2853 | webadm       |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

MariaDB [(none)]>
```

⚠ Warning

The output of `SHOW MASTER STATUS` will reveal the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number.

Let's start with the —NODE 2— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 1—.

```
---NODE 2---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.80', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000001',
MASTER_LOG_POS = 2853;
Query OK, 0 rows affected (0.05 sec)

MariaDB [(none)]>
```

Continue with the —NODE 3— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 2—.

```
---NODE 3---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.81', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000001',
MASTER_LOG_POS = 2853;
Query OK, 0 rows affected (0.06 sec)

MariaDB [(none)]>
```

Continue with the —NODE 4— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 3—.

```
---NODE 4---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.82', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000001',
MASTER_LOG_POS = 2853;
Query OK, 0 rows affected (0.06 sec)

MariaDB [(none)]>
```

At last the —NODE 1— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 4—.

```
---NODE 1---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.83', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000001',
MASTER_LOG_POS = 2853;
Query OK, 0 rows affected (0.05 sec)

MariaDB [(none)]>
```

```
---NODE 1234---
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.00 sec)
```

9.2.2.3 Verify Replication Status

```
---NODE 1---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
```

```
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.83
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 2853
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 537
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2853
              Relay_Log_Space: 831
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 4
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 2---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.80
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 2853
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 537
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2853
              Relay_Log_Space: 831
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
```

```
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 3---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.81
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 2853
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 537
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2853
              Relay_Log_Space: 831
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos: 
      Replicate_Do_Domain_Ids: 
  Replicate_Ignore_Domain_Ids: 
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

```
---NODE 4---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.82
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 2853
               Relay_Log_File: slave-relay.000002
                Relay_Log_Pos: 537
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB: 
           Replicate_Do_Table: 
       Replicate_Ignore_Table: 
      Replicate_Wild_Do_Table: 
  Replicate_Wild_Ignore_Table: 
                   Last_Errno: 0
                   Last_Error: 
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2853
              Relay_Log_Space: 831
              Until_Condition: None
               Until_Log_File: 
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File: 
           Master_SSL_CA_Path: 
              Master_SSL_Cert: 
            Master_SSL_Cipher: 
               Master_SSL_Key: 
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error: 
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 3
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]> exit
Bye
```

9.2.3 WebADM HA Cluster

Use the RCDevs Repository to install WebADM with all WebApps and Services.

```
---NODES 1234---
root@ubuntu18-webadm1:/home/webadm1# apt-get install webadm-all-in-one
...
Setting up openotp (1.4.2-1) ...
Setting up pwreset (1.0.12-1) ...
Setting up webadm (1.6.9-3) ...
WebADM Server needs to be configured.
Please run /opt/webadm/bin/setup.
Setting up tiqr (1.2.5-3) ...
Setting up selfreg (1.1.8-0) ...
Setting up smshub (1.1.2-0) ...
Setting up selfdesk (1.1.8-1) ...
Setting up spankey (2.0.2-2) ...
Setting up opensso (1.0.8-0) ...
Setting up openid (1.3.0-1) ...
Setting up webadm-all-in-one (1.0.0-0) ...
root@ubuntu18-webadm1:/home/webadm1#
```

Run the WebADM setup script on —NODE 1—. It initializes the WebADM PKI, etc…

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# /opt/webadm/bin/setup
Checking system architecture...Ok
Setup WebADM as master server or slave (secondary server in a cluster) (m/s)? m
WebADM proposes 3 default configuration templates:
  1) Default configuration (Novell, eDirectory, Oracle, OpenLDAP)
  2) Active Directory with schema extention (preferred with AD)
  3) Active Directory without schema extention
Choose a template number or press enter for default: 1
Enter the server fully qualified host name (FQDN): webadm.local
Enter your organization name: RCDevs
Generating CA private key... Ok
Creating CA certificate... Ok
Generating SSL private key... Ok
Creating SSL certificate request... Ok
Signing SSL certificate with CA... Ok
Adding CA certificate to the local trust list... Ok
Setting file permissions... Ok
Adding system user to dialout group... Ok
Do you want WebADM to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register WebADM logrotate script (y/n)? y
Adding logrotate scripts... Ok
Do you want to generate a new secret key in webadm.conf (y/n)? y
Generating secret key string... Ok
WebADM has successfully been setup.
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.3.1 Enterprise License

Warning

Any high availability and clustering feature require an RCDevs Enterprise license. Without a valid license file, the HA and cluster features are automatically disabled.

Copy your Enterprise License into the `/opt/webadm/conf` folder.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# cp license.key /opt/webadm/conf
```

9.2.3.2 Adjust servers.xml

Edit on —NODE 1— the `/opt/webadm/conf/servers.xml` file. Adjust the LDAP Server, SQL Server, Session Server, and PKI Server parameters.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# vi /opt/webadm/conf/servers.xml
<?xml version="1.0" encoding="UTF-8" ?>

<Servers>

<!--
******************************************
***  WebADM Remote Server Connections  ***
******************************************

You can configure multiple instances for each of the following servers.
At login, WebADM will try to connect the configured servers in the same
order they appear in this file and uses the first one it successfully
establishes the connection to. If the server connection goes down, it
will automatically failover to the next configured server.

At least one LDAP server is required to run WebADM.
Supported servers: OpenLDAP, Active Directory, Novell eDirectory, 389.

Allowed LDAP parameters are:
 - name: server friendly name
 - host: server hostname or IP address
 - port: LDAP port number
   default and TLS: 389
   default SSL: 636
 - encryption: connection type
   allowed type are NONE, SSL and TLS
   default: 'NONE'
 - ca_cert: Trusted CA for SSL and TLS
 - cert_file: client certificate file
 - cert_key: client certificate key
-->

<LdapServer name="LDAP Server"
 host="192.168.3.80"
 port="389"
 encryption="TLS"
 ca_file="" />
<LdapServer name="LDAP Server 2"
 host="192.168.3.81"
 port="389"
 ca_file="" />
<LdapServer name="LDAP Server3"
 host="192.168.3.82"
 port="389"
 encryption="TLS"
 ca_file="" />
<LdapServer name="LDAP Server 4"
 host="192.168.3.83"
 port="389"
 encryption="TLS"
 ca_file="" />

<!--
SQL servers are used for logs; message localizations and inventories.
Supported servers: MySQL5, MySQL8, PostgreSQL, MSSQL, Sybase, Oracle, SQLite.

Allowed LDAP parameters are:
 - type: MySQL5, MySQL8, MariaDB, PostgreSQL, MSSQL, Sybase, Oracle or SQLite.
 - name: server friendly name
 - host: server hostname or IP address
 - port: SQL port number (depends on server type)
 - user: database user
 - password: database password
 - database: database name
 - tnsname: Oracle TNS name (Oracle only)

With SQLite, only the 'database' must be set and other parameters are
ignored. The database is the full path to an SQLite DB file where WebADM
has full write access.

With Oracle, you can optionally use TNS names. If the 'tnsname' is set
then the 'host' and 'port' parameters are ignored and a tnsnames.ora
file must exist under the conf/ directory.
-->

<SqlServer name="SQL Server"
 type="MySQL8"
 host="192.168.3.80"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 2"
 type="MySQL8"
 host="192.168.3.81"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 3"
 type="MySQL8"
 host="192.168.3.82"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />
<SqlServer name="SQL Server 4"
 type="MySQL8"
 host="192.168.3.83"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="NONE" />

<!--
A session server is required for web services using sessions
such as OpenOTP. You can specify one or more SQL servers here.
The session server is included in WebADM. So you can keep the
default settings here.
-->

<SessionServer name="Session Server"
 host="192.168.3.80"
 port="4000"
 secret="" />
<SessionServer name="Session Server 2"
 host="192.168.3.81"
 port="4000"
 secret="" />
<SessionServer name="Session Server 3"
 host="192.168.3.82"
 port="4000"
 secret="" />
<SessionServer name="Session Server 4"
 host="192.168.3.83"
 port="4000"
 secret="" />

<!--
A PKI server (or CA) is required for signing user certificates.
The RSign PKI server is included in WebADM. So you can keep the
default settings here.
-->

<PkiServer name="PKI Server"
 host="192.168.3.80"
 port="5000"
 secret="secret"
 ca_file="" />
...
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.3.3 Adjust rsignd.conf

On the —NODE 1—, allow client PKI connections to the Rsignd PKI server. This is done by adding the client configuration blocks for the other nodes in the `/opt/webadm/conf/rsignd.conf` file. The password/secret for the PKI server will be in this case `secret`.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# vi /opt/webadm/conf/rsignd.conf
#
# WebADM PKI Server Configuration
#
...
#
# Client sections
#
# Declare here the Rsign clients with IP addresses or hostnames.
# In cluster mode, the client WebADM server(s) must be defined here!

client {
 hostname 192.168.3.80
 secret secret
}
client {
 hostname 192.168.3.81
 secret secret
}
client {
 hostname 192.168.3.82
 secret secret
}
client {
 hostname 192.168.3.83
 secret secret
}

root@ubuntu18-webadm1:/home/webadm1#
```

9.2.3.4 Start WebADM

Start WebADM and login for the 1st time into the graphical setup.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# /opt/webadm/bin/webadm start
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM PKI server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server (192.168.3.80)

Checking LDAP proxy user access... ERROR
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm master)
root@ubuntu18-webadm1:/home/webadm1#
```

Now we connect to the the WebADM Admin Portal on `https://192.168.3.80`.

Important

If you use RCDevs Directory Server, the admin DN is *cn=admin,o=root*. The default password is *password*.

WebADM Admin Portal Login (RCDevs Directory Server)

The Setup button will appear on the home page when you enter the WebADM Admin Portal.

Now click on the `Create/Update SQL database tables`, `Create WebADM proxy user`, `Setup permissions` and `Create default containers and objects` buttons to complete the setup.

We will be able to use the `admin` user after the first configuration.

9.2.3.5 Setup WebADM Slaves

The WebADM setup script must be run using the `slave` parameter with the command `/opt/webadm/bin/setup slave` on —NODE 234—. The master PKI server address is in this case `192.168.3.80`. The master PKI server secret is `secret` as defined before in 9.2.3.3 Adjust rsignd.conf.

```
---NODE 234---
root@ubuntu18-webadm2:/home/webadm2# /opt/webadm/bin/setup slave
Checking system architecture...Ok
WebADM proposes 3 default configuration templates:
  1) Default configuration (Novell, eDirectory, Oracle, OpenLDAP)
  2) Active Directory with schema extention (preferred with AD)
  3) Active Directory without schema extention
Choose a template number or press enter for default: 1
Enter the server fully qualified host name (FQDN): webadm.local
Enter the master PKI server address: 192.168.3.80
Enter the master PKI server port (enter for default):
Enter the master PKI server secret: secret
Testing PKI server conection... Ok
Retrieving PKI CA certificate...Ok
Reading organization name from CA certificate...
Generating SSL private key... Ok
Creating SSL certificate request... Ok
Signing SSL certificate with PKI server... Ok
Adding CA certificate to the local trust list... Ok
Setting file permissions... Ok
Adding system user to dialout group... Ok
Do you want WebADM to be automatically started at boot (y/n)? y
Adding systemd service... Ok
Do you want to register WebADM logrotate script (y/n)? y
Adding logrotate scripts... Ok
WebADM has successfully been setup.
root@ubuntu18-webadm2:/home/webadm2#
```

9.2.3.6 Copy Setup Files to Slaves

Finally, save the WebADM configuration and copy it to the other —NODE 234—. At last, start WebADM on the other —NODE 234—. Now the High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB replication and with the RCDevs Directory Server LDAP

(TLS) replication is running.

```
---NODE 1---
root@ubuntu18-webadm1:/home/webadm1# cd /
root@ubuntu18-webadm1:/# tar czvf webadm_conf.tar.gz /opt/webadm/conf
tar: Removing leading `/' from member names
/opt/webadm/conf/
/opt/webadm/conf/servers.xml
/opt/webadm/conf/webadm.conf.default
/opt/webadm/conf/license.key
/opt/webadm/conf/objects.xml
/opt/webadm/conf/rsignd.conf
/opt/webadm/conf/rsignd.conf.default
/opt/webadm/conf/rsignd.conf.bak
/opt/webadm/conf/servers.xml.default
/opt/webadm/conf/objects.xml.bak
/opt/webadm/conf/webadm.conf.bak
/opt/webadm/conf/objects.xml.default
/opt/webadm/conf/servers.xml.bak
/opt/webadm/conf/webadm.conf
root@ubuntu18-webadm1:/# scp webadm_conf.tar.gz webadm2@192.168.3.81:/tmp/
webadm2@192.168.3.81's password:
webadm_conf.tar.gz                            100%   22KB   8.1MB/s   00:00
root@ubuntu18-webadm1:/# scp webadm_conf.tar.gz webadm3@192.168.3.82:/tmp/
webadm3@192.168.3.82's password:
webadm_conf.tar.gz                            100%   22KB  10.9MB/s   00:00
root@ubuntu18-webadm1:/# scp webadm_conf.tar.gz webadm4@192.168.3.83:/tmp/
webadm4@192.168.3.83's password:
webadm_conf.tar.gz                            100%   22KB   9.0MB/s   00:00
root@ubuntu18-webadm1:/# rm webadm_conf.tar.gz
root@ubuntu18-webadm1:/#
```

```
---NODE 234---
root@ubuntu18-webadm2:/home/webadm2# cp /tmp/webadm_conf.tar.gz /
root@ubuntu18-webadm2:/home/webadm2# cd /
root@ubuntu18-webadm2:/# tar xzvf webadm_conf.tar.gz
opt/webadm/conf/
opt/webadm/conf/servers.xml
opt/webadm/conf/webadm.conf.default
opt/webadm/conf/license.key
opt/webadm/conf/objects.xml
opt/webadm/conf/rsignd.conf
opt/webadm/conf/rsignd.conf.default
opt/webadm/conf/rsignd.conf.bak
opt/webadm/conf/servers.xml.default
opt/webadm/conf/objects.xml.bak
opt/webadm/conf/webadm.conf.bak
opt/webadm/conf/objects.xml.default
opt/webadm/conf/servers.xml.bak
opt/webadm/conf/webadm.conf
root@ubuntu18-webadm2:/# rm /tmp/webadm_conf.tar.gz
root@ubuntu18-webadm2:/# /opt/webadm/bin/webadm start
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server (192.168.3.80)

Checking LDAP proxy user access... Ok
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm slave)
Session replication status: Active (0.0012 sec)
root@ubuntu18-webadm2:/#
```

Now verify if the `Network Service Statuses` under the `Admin` tab are online. That's it, successfully set up a High Availability 4 Nodes Cluster with a MULTI-MASTER MariaDB replication and with the RCDevs Directory Server LDAP (TLS) replication.

9.2.4 MariaDB TLS Replication

Let’s enable TLS for the MULTI-MASTER MariaDB replication.

```
---NODE 1234---
root@ubuntu18-webadm1:/# mkdir -p /etc/mysql/ssl/
root@ubuntu18-webadm1:/# cd /etc/mysql/ssl/
root@ubuntu18-webadm1:/etc/mysql/ssl#
```

9.2.4.1 Export Certificates

Instead of using your own certificates, one can issue and export SSL Certificate over WebADM GUI under the Admin tab.

Click on `Download WebADM CA Certificate` to download it.

Now click on `Issue Server or Client SSL Certificate`, add an `FQDN: mariadbserver` and select `Server`.

Download the Key and Cert File.

Click on `Issue Server or Client SSL Certificate`, add an `FQDN: mariadbclient` and select `Client`.

Download Cert & Key File.

Finally, copy the following certificates `ca.crt` and `mariadbclient.crt` in to the WebADM configuration folder `/opt/webadm/conf/`.

9.2.4.2 Copy Certificates to all the Nodes

Copy the following certificates `ca.crt`, `mariadbserver.crt` and `mariadbserver.key` to all the nodes —NODE 1234—.

```
administor:Downloads$ ssh webadm1@192.168.3.80 mkdir /tmp/ssl/
webadm1@192.168.3.80's password:
administor:Downloads$ ssh webadm2@192.168.3.81 mkdir /tmp/ssl/
webadm2@192.168.3.81's password:
administor:Downloads$ ssh webadm3@192.168.3.82 mkdir /tmp/ssl/
webadm3@192.168.3.82's password:
administor:Downloads$ ssh webadm4@192.168.3.83 mkdir /tmp/ssl/
webadm4@192.168.3.83's password:
administor:Downloads$ scp *.pem webadm1@192.168.3.80:/tmp/ssl/
webadm1@192.168.3.80's password:
ca.crt                                  100% 1142     1.7MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$ scp *.pem webadm2@192.168.3.81:/tmp/ssl/
webadm2@192.168.3.81's password:
ca.crt                                  100% 1142     1.7MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$ scp *.pem webadm3@192.168.3.82:/tmp/ssl/
webadm3@192.168.3.82's password:
ca.crt                                  100% 1142     1.7MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$ scp *.pem webadm4@192.168.3.83:/tmp/ssl/
webadm4@192.168.3.83's password:
ca.crt                                  100% 1142     1.7MB/s   00:00
mariadbserver.crt                       100% 1092     1.5MB/s   00:00
mariadbserver.key                       100% 1092     1.4MB/s   00:00
administor:Downloads$
```

⚠ Warning

Set the owner to mysql and the rights for the MariaDB certificate files.

```
---NODE 1234---
root@ubuntu18-webadm1:/home/webadm1# mv /tmp/ssl/* /etc/mysql/ssl
root@ubuntu18-webadm1:/home/webadm1# chown mysql:mysql /etc/mysql/ssl
root@ubuntu18-webadm1:/home/webadm1# chown mysql:mysql /etc/mysql/ssl/*
root@ubuntu18-webadm1:/home/webadm1# chmod 640 /etc/mysql/ssl/*
root@ubuntu18-webadm1:/home/webadm1# rm -r /tmp/ssl/
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.4.3 Adjust server.cnf

Edit the MariaDB configuration file `/etc/mysql/mariadb.conf.d/50-server.cnf` on all the nodes —NODE 1234— to add the path of the certificates, `ssl-ca`, `ssl-cert` and `ssl-key`. Afterward, restart the MariaDB service.

```
---NODE 1234---
root@ubuntu18-webadm1:/home/webadm1# vi /etc/mysql/mariadb.conf.d/50-server.cnf
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
bind-address    = 192.168.3.80
server-id       = 1
replicate-same-server-id = 0
auto-increment-increment = 4
auto-increment-offset = 1
replicate-do-db = webadm
log_bin         = mysql-bin
log-basename    = mysql
binlog-do-db    = webadm
log-slave-updates
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
...
#
# * Security Features
#
# Read the manual, too, if you want chroot!
# chroot = /var/lib/mysql/
```

```
#
# For generating SSL certificates I recommend the OpenSSL GUI "tinyca".
#
# ssl-ca=/etc/mysql/cacert.pem
# ssl-cert=/etc/mysql/server-cert.pem
# ssl-key=/etc/mysql/server-key.pem
ssl-ca=/etc/mysql/ssl/ca.crt
ssl-cert=/etc/mysql/ssl/smariadbserver.crt
ssl-key=/etc/mysql/ssl/mariadbserver.key
...

root@ubuntu18-webadm1:/home/webadm1# systemctl restart mysql
root@ubuntu18-webadm1:/home/webadm1# systemctl status mysql -l
systemctl restart mysql
● mariadb.service - MariaDB 10.1.34 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset:
enabled)
   Active: active (running) since Wed 2019-02-06 14:29:25 UTC; 5s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
  Process: 4438 ExecStartPost=/bin/sh -c systemctl unset-environment
_WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 4432 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
  Process: 4270 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= ||
VAR=`/usr/bin/galera_recovery`; [ $? -eq 0 ]   && systemctl set-environment
_WSREP_START_POSITION=$VAR || exit 1 (code=ex
  Process: 4264 ExecStartPre=/bin/sh -c systemctl unset-environment
_WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 4242 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d
/var/run/mysqld (code=exited, status=0/SUCCESS)
 Main PID: 4401 (mysqld)
   Status: "Taking your SQL requests now..."
    Tasks: 32 (limit: 2292)
   CGroup: /system.slice/mariadb.service
           └─4401 /usr/sbin/mysqld

Feb 06 14:29:24 ubuntu18-webadm1 systemd[1]: Starting MariaDB 10.1.34 database
server...
Feb 06 14:29:24 ubuntu18-webadm1 mysqld[4401]: 2019-02-06 14:29:24 139672427105408
[Note] /usr/sbin/mysqld (mysqld 10.1.34-MariaDB-0ubuntu0.18.04.1) starting as process
4401 ...
Feb 06 14:29:25 ubuntu18-webadm1 /etc/mysql/debian-start[4441]: /usr/bin/mysql_upgrade:
the '--basedir' option is always ignored
Feb 06 14:29:25 ubuntu18-webadm1 /etc/mysql/debian-start[4441]: Looking for 'mysql' as:
/usr/bin/mysql
Feb 06 14:29:25 ubuntu18-webadm1 /etc/mysql/debian-start[4441]: Looking for
'mysqlcheck' as: /usr/bin/mysqlcheck
Feb 06 14:29:25 ubuntu18-webadm1 /etc/mysql/debian-start[4441]: This installation of
MySQL is already upgraded to 10.1.34-MariaDB, use --force if you still need to run
mysql_upgrade
Feb 06 14:29:25 ubuntu18-webadm1 systemd[1]: Started MariaDB 10.1.34 database server.
root@ubuntu18-webadm1:/home/webadm1# netstat -tulpn
Active Internet connections (only servers)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN
1528/webadm-rsignd
tcp        0      0 192.168.3.80:3306       0.0.0.0:*               LISTEN
4401/mysqld
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
818/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
1257/sshd
tcp        0      0 0.0.0.0:8443            0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
1572/webadm-httpd
tcp        0      0 0.0.0.0:636             0.0.0.0:*               LISTEN
1313/rcdevs-slapd
tcp        0      0 0.0.0.0:4000            0.0.0.0:*               LISTEN
1462/webadm-session
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN
1313/rcdevs-slapd
tcp6       0      0 :::22                   :::*                    LISTEN
1257/sshd
tcp6       0      0 :::4000                 :::*                    LISTEN
1462/webadm-session
udp        0      0 127.0.0.53:53           0.0.0.0:*
818/systemd-resolve
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.4.4 Enable SSL/TLS

Log in to MariaDB as the root user and enable the SSL/TLS.

```
---NODE 1234---
root@ubuntu18-webadm1:/home/webadm1# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 363
Server version: 10.1.34-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'localhost' REQUIRE
X509;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.80' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.81' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.82' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON webadm.* to 'webadm'@'192.168.3.83' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'localhost' REQUIRE SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.80' REQUIRE
SSL;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.81' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.82' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'webadm'@'192.168.3.83' REQUIRE
SSL;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> STOP SLAVE;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]>
```

```
---NODE 1---
MariaDB [(none)]> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000003 |     2201 | webadm       |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

---NODE 234---
MariaDB [(none)]> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000003 |     2217 | webadm       |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)
```

⚠ Warning

The output of `SHOW MASTER STATUS` will reveal the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number.

Let's start with the —NODE 2— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 1—.

```
---NODE 2---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.80', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000003',
MASTER_LOG_POS = 2201, MASTER_SSL=1;
Query OK, 0 rows affected (0.03 sec)

MariaDB [(none)]>
```

Continue with the —NODE 3— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 2—.

```
---NODE 3---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.81', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000003',
MASTER_LOG_POS = 2217, MASTER_SSL=1;
Query OK, 0 rows affected (0.04 sec)

MariaDB [(none)]>
```

Continue with the —NODE 4— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 3—.

```
---NODE 4---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.82', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000003',
MASTER_LOG_POS = 2217, MASTER_SSL=1;
Query OK, 0 rows affected (0.03 sec)

MariaDB [(none)]>
```

At last the —NODE 1— and replace the `MASTER_LOG_FILE` name and the `MASTER_LOG_POS` number with the values of `SHOW MASTER STATUS` from —NODE 4—.

```
---NODE 1---
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST = '192.168.3.83', MASTER_USER =
'webadm', MASTER_PASSWORD = 'webadm', MASTER_LOG_FILE = 'mysql-bin.000003',
MASTER_LOG_POS = 2217, MASTER_SSL=1;
Query OK, 0 rows affected (0.04 sec)

MariaDB [(none)]>

---NODE 1234---
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]>
```

9.2.4.5 Verify TLS Status

Verify MariaDB TLS as follows:

```
---NODE 1---
```

```
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.83
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000004
          Read_Master_Log_Pos: 343
               Relay_Log_File: slave-relay.000004
                Relay_Log_Pos: 631
        Relay_Master_Log_File: mysql-bin.000004
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 343
              Relay_Log_Space: 1213
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 4
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]>
```

```
---NODE 2---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.80
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000004
          Read_Master_Log_Pos: 327
               Relay_Log_File: slave-relay.000006
                Relay_Log_Pos: 615
        Relay_Master_Log_File: mysql-bin.000004
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 327
              Relay_Log_Space: 1197
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
```

```
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]>
```

```
---NODE 3---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.3.81
                  Master_User: webadm
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000004
          Read_Master_Log_Pos: 343
               Relay_Log_File: slave-relay.000004
                Relay_Log_Pos: 631
        Relay_Master_Log_File: mysql-bin.000004
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: webadm
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 343
              Relay_Log_Space: 1213
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: Yes
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
 1 row in set (0.00 sec)

 MariaDB [(none)]>
```

```
---NODE 4---
MariaDB [(none)]> SHOW SLAVE STATUS \G
*************************** 1. row ***************************
              Slave_IO_State: Waiting for master to send event
                 Master_Host: 192.168.3.82
                 Master_User: webadm
                 Master_Port: 3306
               Connect_Retry: 60
             Master_Log_File: mysql-bin.000004
         Read_Master_Log_Pos: 343
              Relay_Log_File: slave-relay.000004
               Relay_Log_Pos: 631
       Relay_Master_Log_File: mysql-bin.000004
            Slave_IO_Running: Yes
           Slave_SQL_Running: Yes
             Replicate_Do_DB: webadm
         Replicate_Ignore_DB:
          Replicate_Do_Table:
      Replicate_Ignore_Table:
     Replicate_Wild_Do_Table:
 Replicate_Wild_Ignore_Table:
                  Last_Errno: 0
                  Last_Error:
                Skip_Counter: 0
         Exec_Master_Log_Pos: 343
             Relay_Log_Space: 1213
             Until_Condition: None
              Until_Log_File:
               Until_Log_Pos: 0
          Master_SSL_Allowed: Yes
          Master_SSL_CA_File:
          Master_SSL_CA_Path:
             Master_SSL_Cert:
           Master_SSL_Cipher:
              Master_SSL_Key:
       Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
               Last_IO_Errno: 0
               Last_IO_Error:
              Last_SQL_Errno: 0
              Last_SQL_Error:
```

```
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 3
               Master_SSL_Crl:
           Master_SSL_Crlpath:
                   Using_Gtid: No
                  Gtid_IO_Pos:
      Replicate_Do_Domain_Ids:
  Replicate_Ignore_Domain_Ids:
                Parallel_Mode: conservative
1 row in set (0.00 sec)

MariaDB [(none)]>
```

```
---NODE 1234---
MariaDB [(none)]> SHOW VARIABLES LIKE '%ssl%';
+---------------------+------------------------------------+
| Variable_name       | Value                              |
+---------------------+------------------------------------+
| have_openssl        | NO                                 |
| have_ssl            | YES                                |
| ssl_ca              | /etc/mysql/ssl/ca.crt              |
| ssl_capath          |                                    |
| ssl_cert            | /etc/mysql/ssl/mariadbserver.crt   |
| ssl_cipher          |                                    |
| ssl_crl             |                                    |
| ssl_crlpath         |                                    |
| ssl_key             | /etc/mysql/ssl/mariadbserver.key   |
| version_ssl_library | YaSSL 2.4.4                        |
+---------------------+------------------------------------+
10 rows in set (0.01 sec)

MariaDB [(none)]> status;
--------------
mysql  Ver 15.1 Distrib 10.1.34-MariaDB, for debian-linux-gnu (x86_64) using readline
5.2

Connection id:  54
Current database:
Current user:  root@localhost
SSL:   Cipher in use is DHE-RSA-AES256-SHA
Current pager:  stdout
Using outfile:  ''
Using delimiter: ;
Server:   MariaDB
Server version:  10.1.34-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04
Protocol version: 10
Connection:  Localhost via UNIX socket
Server characterset: latin1
Db     characterset: latin1
Client characterset: latin1
Conn.  characterset: latin1
UNIX socket:  /var/run/mysqld/mysqld.sock
Uptime:   16 min 58 sec

Threads: 2  Questions: 233  Slow queries: 0  Opens: 32  Flush tables: 1  Open tables:
26  Queries per second avg: 0.228
--------------
```

9.2.4.6 Adjust servers.xml

Finally, adjust the parameter encryption from `NONE` to `TLS` and add the certificates in the configuration file `/opt/webadm/conf/servers.xml` of all nodes —NODE 1234—. Afterward, restart WebADM to enable TLS for MULTI-MASTER MariaDB replication.

Note

Please use the `MySQL8` instead of the `MariaDB` driver for certificate based authentication.

```
---NODE 1234---
root@ubuntu18-webadm1:/home/webadm1# vi /opt/webadm/conf/servers.xml
<SqlServer name="SQL Server"
 type="MySQL8"
 host="192.168.3.80"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 2"
 type="MySQL8"
 host="192.168.3.81"
 user="webadm"
 password="webadm"
 database="webadm"
 encryption="TLS"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 3"
 type="MySQL8"
 host="192.168.3.82"
 user="webadm"
 password="webadm"
 database="webadm"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
<SqlServer name="SQL Server 4"
 type="MySQL8"
 host="192.168.3.83"
 user="webadm"
 password="webadm"
 database="webadm"
 ca_file="/opt/webadm/conf/ca.crt"
 cert_file="/opt/webadm/conf/mariadbclient.crt"
 key_file="/opt/webadm/conf/mariadbclient.crt"/>
```

```
root@ubuntu18-webadm1:/home/webadm1# /opt/webadm/bin/webadm restart
Stopping WebADM HTTP server... Ok
Stopping WebADM Watchd server.... Ok
Stopping WebADM PKI server... Ok
Stopping WebADM Session server... Ok
Checking libudev dependency... Ok
Checking system architecture... Ok
Checking server configurations... Ok

Found Trial Enterprise license (LOIC)
Licensed by RCDevs SA to LOIC
Licensed product(s): OpenOTP

Starting WebADM Session server... Ok
Starting WebADM PKI server... Ok
Starting WebADM Watchd server... Ok
Starting WebADM HTTP server... Ok

Checking server connections. Please wait...
Connected LDAP server: LDAP Server (192.168.3.80)
Connected SQL server: SQL Server (192.168.3.80)
Connected PKI server: PKI Server (192.168.3.80)
Connected Session server: Session Server 2 (192.168.3.81)

Checking LDAP proxy user access... Ok
Checking SQL database access... Ok
Checking PKI service access... Ok

Cluster mode enabled with 4 nodes (I'm slave)
Session replication status: Active (0.0014 sec)
root@ubuntu18-webadm1:/home/webadm1#
```

9.2.4.7 Iptables Firewall Rules

At RCDevs Hardening Guide is an example of the iptables firewall rules for a high availability cluster with 4 nodes.