# Implementing, Deploying and Managing a High Availability Distributed Solution on

# AXIGEN Mail Server

**Last Updated on: April 15, 2010**

**GECAD Technologies**
10A Dimitrie Pompei Blvd., BUCHAREST 2,
ROMANIA. Zip code: 020337.
Tel.: +40 21 303 20 80
Fax: +40 21 303 20 81
http://www.axigen.com

## Copyright & trademark notices

This article applies to version 7.0 or higher of the AXIGEN Mail Server.

## Notices



## Copyright Acknowledgement



GECAD TECHNOLOGIES S.A.

10A Dimitrie Pompei Blvd., Connect Business Center, 2nd fl., Bucharest 2, ROMANIA;
Phone: +40-21-303-2080
Fax: +40-21-303-2081
Email Sales: sales@axigen.com
Email Technical Support: support@axigen.com
Website: http://www.axigen.com

## Summary:

# 1 Introduction

## 1.1 Overview

This document describes an implementation of a large-scale messaging solution relying on the AXIGEN Mail Server software. The global architecture for the solution is described along with implementation details and operation and maintenance procedures.

## 1.2 Intended Audience

The information in this document is intended for users who are evaluating the benefits of a distributed, high availability solution, as well as for integrators and operational personnel. The components of such a solution, both software and hardware, are also listed in this document, thus ensuring the ability to assess overall associated costs.

## 1.3 Definitions, Terms and Abbreviations

- ***Vertical scalability*** – potential increase in the processing capacity of a machine attainable by hardware upgrades;
- ***Horizontal Scalability*** – potential increase in the processing capacity of a cluster attainable by increasing the number of nodes (machines);
- ***Statefull Services*** – services that provide access to persistent information (i.e. account configuration and mailbox) over multiple sessions. Typically refers to a service in the back-end tier. E.g.: IMAP services for an account;
- ***Stateless Services*** – services that do not store persistent information over multiple sessions. Typically refers to the services in the front-end tier. E.g.: IMAP Proxy;
- ***Front-end Tier*** – Subnet, medium security level, provides proxy services;
- ***Back-end Tier*** – Subnet, high security level, provides data storage and directory services;
- ***Front-end Node*** – Machine residing in the front-end network tier, providing proxy functionality;
- ***Back-end Node*** – Machine residing in the back-end network tier, participating in the high-availability cluster.

# 2 Benefits

## 2.1 Scalability

### 2.1.1 Statefull Services

Non-distributed email solutions, where account information (configuration and messages) is stored on a single machine, allow vertical scalability through hardware upgrades (CPU, RAM, disk). However, due to limitations in a typical machine (i.e. max 2 CPU, max 4 GB RAM etc) an upper limit is eventually reached, where one can no longer upgrade one machine – we shall refer to this as *vertical scalability limit*.

When the vertical scalability limit is reached, the only solution available is to distribute account information (configuration and mailbox) on more than one machine – we shall refer to this as *horizontal scalability*. Since information for one account is atomic and cannot be spread across multiple machines, the solution is to distribute accounts on more than one machine. This way, for a single account, there will be one machine responding to requests (IMAP, POP, SMTP, WebMail) for that specific account. Thus, when the overall capacity (in terms of active accounts) of the messaging solution is reached, adding one more machine to the solution and making sure new accounts are created provides a capacity upgrade, therefore allowing virtually unlimited horizontal scalability.

It must be noted that, since each account of the system is serviced by a specific node, *a centralized location directory* must be available to provide location services. In our case, an LDAP system will store information about which node is able to service requests for a specific account.

### 2.1.2 Stateless Services

Since stateless services do not store information over multiple sessions, we can assume that two different machines are able to service requests for the same account. This way, horizontal scalability can be achieved by simply adding more machines providing the same service in the exact same configuration.

The only remaining requirement is to ensure that requests to a specific service are distributed evenly throughout the machines providing that specific service (i.e. if the system contains two machines providing IMAP proxy services, half of the incoming IMAP connections must reach one of the machines and the rest of the connections must reach the other machine). This functionality is provided by *a load balancer*, be it hardware (dedicated) or software (Linux machine running LVS).

## 2.2 High Availability and Fault Tolerance

### 2.2.1 Statefull Services

Consider the fact that, for statefull services, requests for one specific account are made to a specific machine. If that specific machine experiences a fault and can no longer respond to requests, none of the other machines are able to service the account in question. A mechanism is required to ensure that, in the event of a catastrophic failure on one machine, some other node must take-over the task of servicing requests for that account thus providing high-availability.

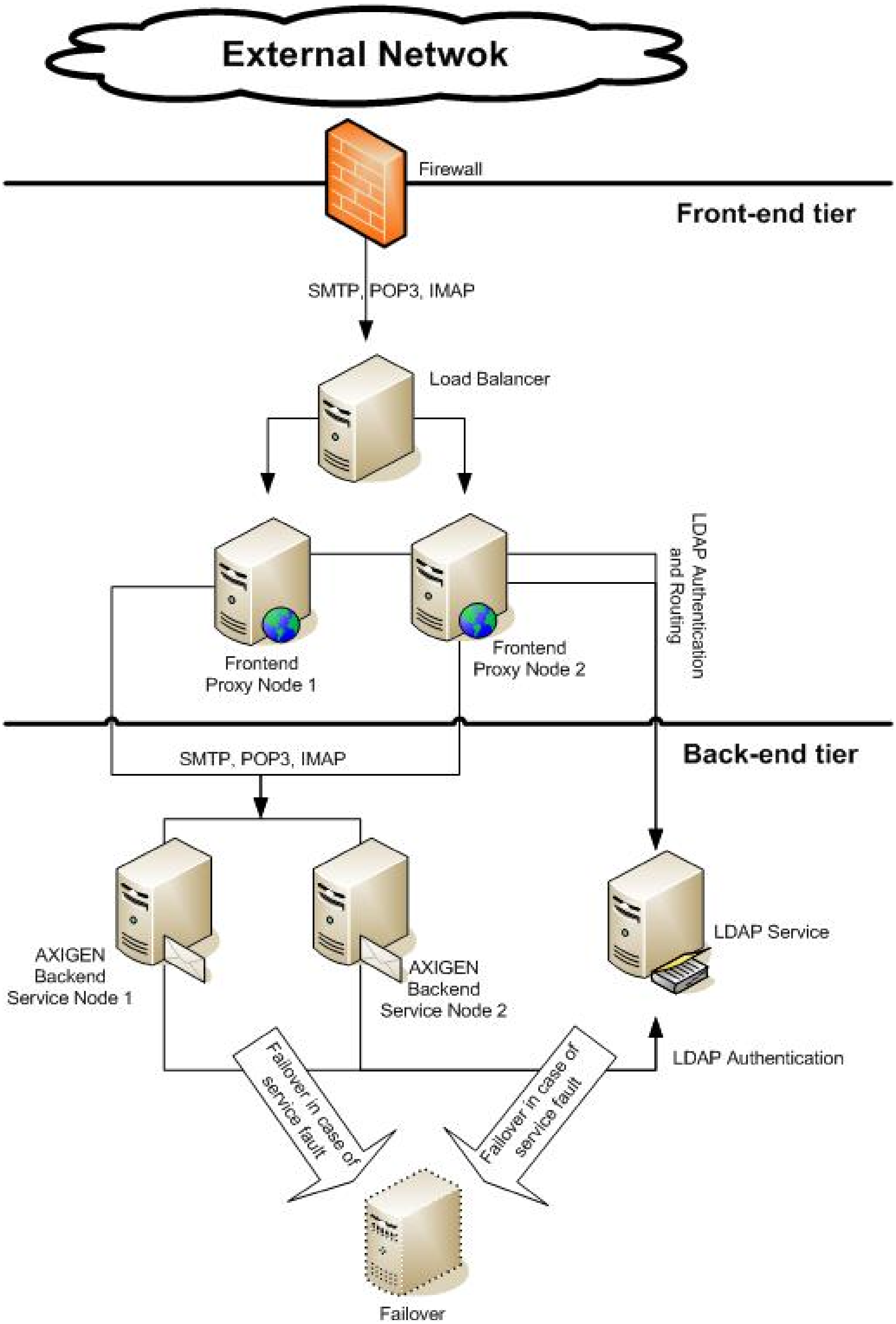
RedHat Clustering Suit provides this exact functionality; it ensures that, if one node running a statefull service fails, another node will automatically detect the fault and start the required service in-place of the failed node, providing minimal downtime to that service.

### 2.2.2 Stateless Services

In the case of stateless services, since any of the nodes providing the same service is able to respond to requests for any account, the only requirement is to make sure that the request distribution mechanism (load balancer) can detect when one of the nodes no longer responds to requests and ceases to direct service requests to that specific node. The total request processing capacity is decreased (the system will respond slower, since one node no longer processes requests), but all service requests can still be processed.

# 3 Solution Architecture

A global description of the architecture of the messaging solution is provided below.



## *3.1 Multi-tier*

The solution uses three tiers to provide the required functionality. The load balancer tier provides services for network layer 4 (transport), TCP connections, and is completely unaware of account information; it only provides distribution of connections to the nodes in the front-end tier.

The front-end tier comprises of nodes running proxy services and SMTP routing services. Its task is to ensure that messages and connections are routed to the appropriate node (depending on the account for which a request is being performed) in the back-end tier.

Finally, the back-end tier provides access to persistent data (such as account configuration and mailbox data); each node in the back-end tier is capable of responding to requests for a set of accounts. No node in the back-end tier is capable of servicing requests for an account that is serviced by a different node.

## *3.2 Service-level High Availability*

Depending on the service type (statefull or stateless), high availability is achieved differently.

### 3.2.1 High Availability for Statefull Services

We shall define, for the back-end tier, the following terms: *service-instance* and *cluster-node.* A service instance comprises of a running service and storage for a specific set of accounts. A cluster node is a machine. Each machine is capable of running one (typically) or more service instances.

High-availability in this tier is achieved by making sure that, *in the event of failure of a physical node, the service instance is automatically started on a different cluster node*. Currently, there are a number of software packages providing this functionality; AXIGEN was tested with Red Hat Clustering Suite.

A disadvantage of this high-availability mechanism is the delay that is induced between the time a service on one node fails (or the node fails completely) and the time the service is restarted on a different node. This delay is caused by the time required for the cluster software (RHCS, in our case) to detect the failure and the time required to remount the data partitions on a different node, activate the virtual network interfaces and start the service. During this period (which may vary from 10 seconds to a couple of minutes) the service is not available for the users.

### 3.2.2 High Availability for Stateless Services

In the case of stateless services (services in the front-end tier), the load distribution system also provides high availability capabilities. The load balancer automatically distributes requests (at TCP level – network layer 4) to all the nodes in the front-end tier, based on the configured algorithm. If one node in the front-end tier fails, the load balancer will automatically detect the failed node and will no longer distribute connections to it. A major advantage over the statefull services high availability mechanism is that, due to its ‘active-active’ nature, a node failure causes no service downtime.

## 3.3 I/O High Availability

The high-availability used at service level provides a full ‘no-single-point-of-failure’. However, any faulty hardware component in a node causes that node to be unusable thus diminishing the total processing power (hence the total transaction capacity) the solution provides. There is a mechanism which makes sure that, even in the case of an I/O controller failure, the node can continue to provide the service; this relies on having duplicate I/O controllers on each node and a software method of failover (rerouting I/O traffic from the faulty controller to the healthy one).

The I/O high availability can be used for disk I/O and network I/O fault tolerance, provided that duplicate controllers are available on the nodes. This reduces the occurrence of service downtime in the case of statefull services; if an I/O controller fails, the service would need to be restarted on a different node.

# 4 Requirements

## 4.1 Software

### 4.1.1 OS

- **RedHat Enterprise Linux (Standard or Advanced Platform) version 5.x**

Or

- **CentOS version 5.x**

### 4.1.2 AXIGEN

- **AXIGEN SP Edition version 7.x or later**

### 4.1.3 Directory

- **OpenLDAP 2.4.x**. Typically, the OpenLDAP package in the Linux distribution should be used.

### 4.1.4 Cluster software

- **Red Hat Cluster Suite 5 (conga – part of RHEL5 Advanced Platform).** If a hardware load balancer is not available, the Linux Virtual Server (LVS) component of the RedHat Cluster Suite can be used to achieve the same purpose.

## 4.2 Hardware

### 4.2.1 Load balancer

Any Layer 3-7 compatible hardware load balancer can be used to provide request balancing. Alternatively, the Virtual Server component of the RedHat Cluster Suite can be used for balancing.

### 4.2.2 Servers

- **Balancer.** If a hardware load balancer is not employed, one machine must be available for running Virtual Server (component of RHCS).

- **Front-end**

One server must be available for each node in the front-end tier; in order to achieve high availability, at least two nodes must be used. The hardware configuration of the machines and the number of nodes depend on the solution's performance requirements.
It is recommended that RAID1 controllers are used in the front-end nodes to ensure fault tolerance for disk I/O.

- **Back-end**

One server must be used for each of the AXIGEN service nodes in the back-end and one for the Directory. It is recommended that a standby node is also available; it will be used by the clustering software in the event of failure of one of the active nodes.
Each back-end node must have an (or two, if disk I/O fault tolerance is required) external SCSI or FiberChannel ports (depending on the interfaces of the shared storage) in order to connect to the shared storage. Operating system files (root partition) must reside on a local disk, using a RAID1 controller and two disks to provide I/O fault tolerance.

### 4.2.3 Shared Storage

The clustering software (RHCS) requires all nodes in the cluster to access the same storage system. A wide variety of directly-attached storage systems (SAN) accessible via SCSI or FibreChannel is available and the selection will probably be influenced by a number of factors such as scalability and performance requirements, price and/or preferred vendor. An important issue to be considered when making this choice if the limitation of using an SCSI-attached storage: typically, solutions provide only two or four SCSI ports, thus limiting the number of nodes that can be used. FibreChannel solutions provide numerous ports, allowing the solution to scale better, at the expense of the overall solution cost.

### 4.2.4 Fence Devices

Fence devices allow a failed node to be isolated from the storage so that, at no time, two nodes may write on the same partition on the shared storage. There are two types of fence devices:

- **Remote power switches** (allow the cluster software to remotely power down/reboot a node that failed);
- **I/O barriers** (allow the cluster software to block access to the shared storage for a node that failed)

These components are required for the back-end tier of the solution; one fence device port is required for each node in the back-end tier. Example: if using fence devices with 4 ports in a solution which has 8 nodes in the back-end tier, 2 fence devices are required.

## *4.3 Licenses*

### 4.3.1 AXIGEN

The AXIGEN SP Edition is licensed on a per-mailbox model. Thus, no matter how many nodes running AXIGEN are contained in the solution (both in the back-end tier and in the front-end tier), only the total number of mailboxes that is hosted in the solution affects the license price.

### 4.3.2 OS

Red Hat Enterprise Linux is licensed on a per-host model, so a license is required for each node in the cluster (both for back-end, front-end and, if a software load balancer is used, in the load balancer tier).

### 4.3.3 Cluster

Red Hat Clustering Suite (part of Red Hat Enterprise Linux Advanced Platform) is licensed on a per-cluster-node model. Each node in the back-end tier requires a separate RHCS license.

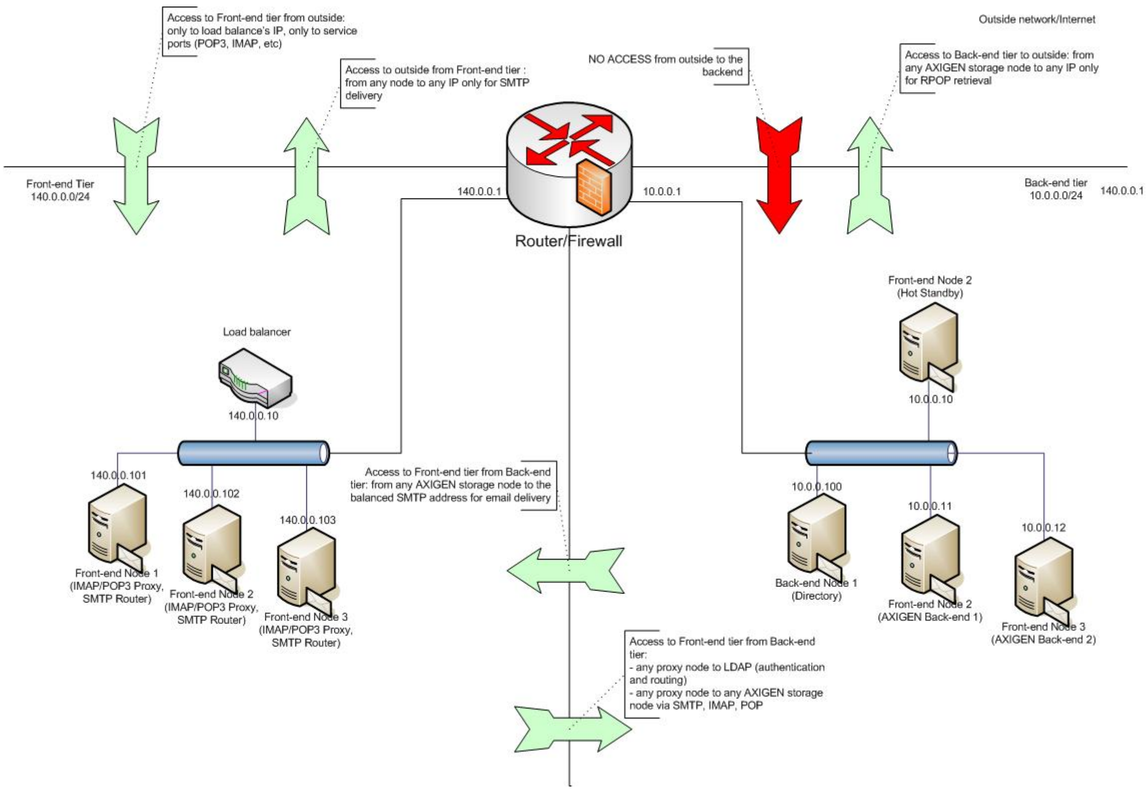
# 5 Setup and Configuration

## 5.1 Network Planning

The front-end and back-end tiers are separated in different subnets and constitute different security zones. The load balancer resides in the same subnet as the front-end tier.

In our example, the back-end layer uses subnet 10.0.0.0/24 and the front-end layer uses a valid, internet routable subnet 140.0.0.0/24 (the 140.0.0.0 IP class is an example and should not be used in a real scenario). A router must exist to connect the outside network, the front-end network and the back-end network, also providing firewall and address translation services.

The image below depicts the network topology for this solution.



## 5.2 DNS Configuration

In our scenario, a DNS service will be configured on the Router/Firewall machine to allow visibility from both front-end tier and back-end tier.
One forward and one reverse lookup zones are required for each network tier (back-end and front-end). The Router/Firewall has one network interface in each zone. The zones for the scenario in this document are described below:

- Front-end
  - DNS zone name: front-end.cluster
  - Zone subnet: 140.0.0.0/24
  - Nodes:
    - Router: router.front-end.cluster = 140.0.0.1
    - Load Balancer Node: loadbalancer.front-end.cluster = 140.0.0.10
    - Front-end Node 1: proxy1.front-end.cluster = 140.0.0.101
    - Front-end Node 2: proxy1.front-end.cluster = 140.0.0.102
    - Front-end Node 3: proxy1.front-end.cluster = 140.0.0.103
  - Virtual (balanced) service addresses (on the load balancer)
    - SMTP: smtp.front-end.cluster = 140.0.0.200
    - POP3: pop3.front-end.cluster = 140.0.0.201
    - IMAP: imap.front-end.cluster = 140.0.0.202
- Back-end
  - DNS zone name: back-end.cluster
  - Zone subnet: 10.0.0.0/24
  - Nodes
    - Router: router.back-end.cluster = 10.0.0.1
    - LDAP: ldapnode.back-end.cluster = 10.0.0.100
    - AXIGEN node 1: axigenode1.back-end.cluster = 10.0.0.11
    - AXIGEN node 1: axigenode2.back-end.cluster = 10.0.0.12
  - Clustered service addresses
    - LDAP: ldap.back-end.cluster = 10.0.0.200
    - AXIGEN1: axigen1.back-end.cluster = 10.0.0.211
    - AXIGEN2: axigen1.back-end.cluster = 10.0.0.211

All the machines in the solution must use the same DNS server to avoid confusions.

## 5.3 Load Balancer

The load balancer resides in the front-end tier subnet and provides the following functionality: it accepts TCP connections on service ports (SMTP – 25, IMAP – 143, POP3 – 110, etc.) and redirects them, based on a scheduling algorithm, to a front-end node. In our example, the load balancer's address is 140.0.0.10 and the front-end nodes are assigned with 140.0.0.101, 140.0.0.102, 140.0.0.103. The following service routing policy should be implemented:

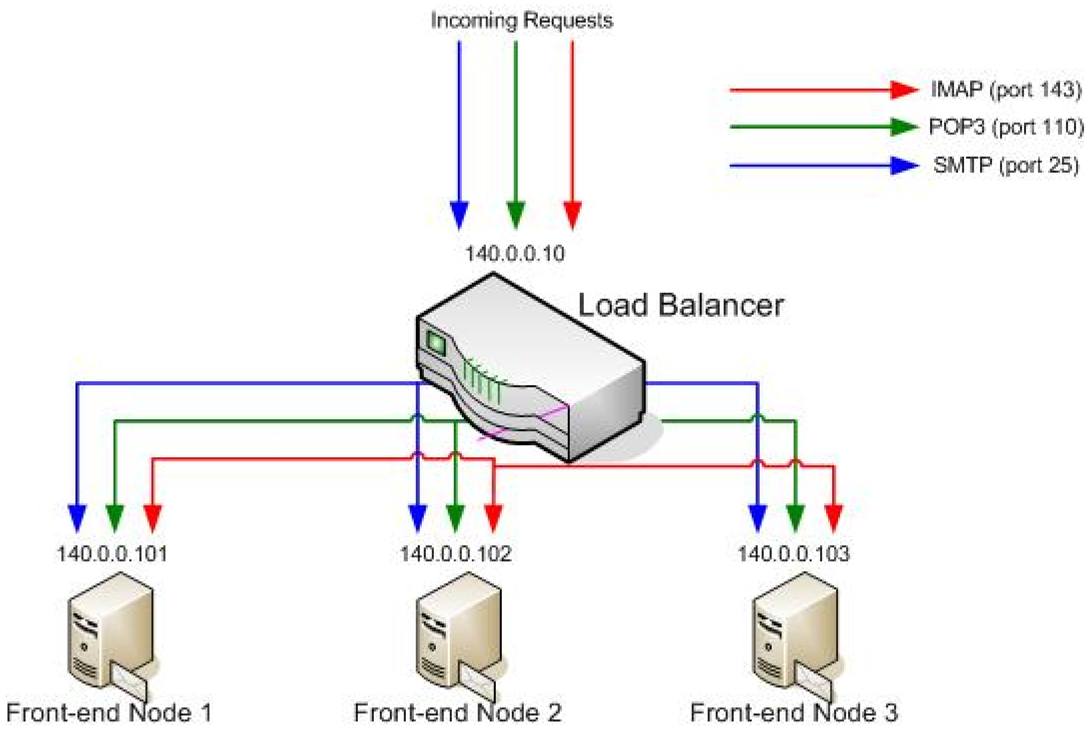
- Service IMAP
  - Virtual server: 140.0.0.10
  - Virtual port: 143
  - Real servers: 140.0.0.101:143, 140.0.0.102:143, 140.0.0.103:143
  - Scheduling algorithm: Least-connection
  - Fault-detection: enabled

- Service POP3
  - Virtual server: 140.0.0.10
  - Virtual port: 110
  - Real servers: 140.0.0.101:110, 140.0.0.102:110, 140.0.0.103:110
  - Scheduling algorithm: Least-connection
  - Fault-detection: enabled
- Service SMTP
  - Virtual server: 140.0.0.10
  - Virtual port: 25
  - Real servers: 140.0.0.101:25, 140.0.0.102:25, 140.0.0.103:25
  - Scheduling algorithm: Least-connection
  - Fault-detection: enabled

If all the nodes in the front-end layer have identical hardware performances, the 'least-connection' scheduling algorithm will suffice. If, however, the hardware differs, a 'weighted least-connection' scheduling algorithm must be used to ensure a uniform load on the front-end nodes.

The figure below depicts the functionality of the load balancer:




Depending on the load balancer that is used (either hardware or software), a dual 'active-active' load balancer setup may be used to ensure no-single-point-of-failure at this tier. Please consult the specific load balancer documentation for details on how to implement such a setup.

## 5.4 Front-end Tier

### 5.4.1 Generic Nodes Configuration

#### (1) Operating System

The operating system on the front-end tier nodes requires no special configuration other than the specific network settings, according to the planned network topology. In our example, each node has one network interface connected to the router in the front-end tier subnet, configured with the specific IP, 140.0.0.x, network mask 255.255.255.0 and gateway 140.0.0.1.

Additional configuration:

- Kernel parameters
    - Disable routing: *net.ipv4.ip_forward = 0*
    - Increase the maximum number of open descriptors: *fs.file-max = 32786* (this also covers open TCP sockets; consider the fact that each service connection may consume up to 3 filedescriptors: one for front-end node to client, one for front-end node to back-end node and one for LDAP query)
    - Tune the virtual memory manager: *vm.bdflush* (this should be configured depending on the specific scenario)
    - Tune the kernel swapping algorithm: *vm.kswapd* (this should be configured depending on the specific scenario)
    - Increase the maximum number of local tcp ports: *net.ipv4.ip_local_port_range = 1024 65000* (this is required so that the front-end node is able to handle many simultaneous TCP connections)
    - Configure the TCP connection backlog: *net.ipv4.tcp_max_syn_backlog* (according to the required setup – a larger backlog consumes more memory but allows a queue for new request in peaks; a smaller backlog is more economical, but connections may be refused in peak periods)
    - Reduce the TCP keep-alive timeout: *net.ipv4.tcp_keepalive_time = 600*
- DNS Resolver
    - Configure the */etc/resolv.conf* file according to your specific network settings
    - In our scenario, the */etc/resolv.conf* file contains

```
search backend.cluster
nameserver 10.0.0.1
```

#### (2) AXIGEN Configuration

The AXIGEN package must be installed according to the instruction manual. After installation, the following configurations must be performed:

- Enabled services: IMAP-Proxy, POP3-Proxy, SMTPIn, SMTPOut, DNR. Other services may be also enabled if required.
- Listeners: IMAP-Proxy, POP3-Proxy, SMTPIn must be configured with one listener each, on the IP address allocated to the front-end node (in our example, front-end node 1's IP is 140.0.0.101)
- LDAP Connector. One LDAP connector is required for both proxies and SMTP routing.
    - LDAP connector parameters:

        - Name: LDAP_Master
        - IP: specific LDAP IP (in our example, *ldap://10.0.0.100*)
        - Port: specific LDAP port (in our example 389)
        - Server type: OpenLDAP
        - Make sure the “Enable Clustered Operations” box is checked
        - Enable “Administrative DN”
        - AdminDN: LDAP Administrator’s DN
          (in our example: *cn=admin,dc=example,dc=com*)
        - Admin DN Password: LDAP Administrator’s password
          (as defined in the LDAP configuration file)
        - Account Base DN: LDAP base of the user entries
          (in our example: *dc=example,dc=com*)
        - Hostname attribute: LDAP attribute holding the hostname of the back-end node where the account resides (in our case *“mailHost”*)
    - The values for the LDAP connector settings must be according to the LDAP schema used in the specific scenario
- UserMap. One UserMap is required to allow the proxies and SMTP router to locate home back-ends for each account.
    - UserMap parameters:
        - Name: LDAP_Master_UserMap
        - Type: LDAP Bind
        - LocalFile: Not used, since this is an LDAP, not a local map
        - userDBConnectorName: LDAP_Master
- Domain Name Resolver
    - Add nameserver:
        - Priority: 5
        - Address: 140.0.0.1 (if the DNS is located on the router, as in our example scenario)
        - Timeout: 2
        - No. of retries: 3

### 5.4.2 IMAP, WebMail and POP Proxies

- Configure the proxies to route connections via a user map, for IMAP, WebMail and POP3
    - In the Mapping Data section of the proxies
        - userMap: the name of the usermap defined in the step above (in our case, "*LDAP_Master_UserMap*")
        - mappingHost: default back-end node hostname to use if the user is not found in LDAP
        - mappingPort: default back-end node port to use if the user is not found in LDAP
- Configure authentication via LDAP
    - For both IMAP and POP3 proxies, set:
        - userdbConnectorType: ldapBind
        - userdbConnectorName: LDAP_Master
        - authenticateOnProxy: yes

### 5.4.3 SMTP Routing

- Enable Routing
    - In the SMTPIn service general context
        - Set the 'enableSMTPRouting' to yes
    - In the Mapping Data section of the SMTPIn service
        - UserMap: LDAP_Master_UserMap
        - mappingHost: default back-end node hostname to use if the user is not found in LDAP
        - mappingPort: default back-end node port to use if the user is not found in LDAP
- Configure authentication via LDAP
    - In the SMTPIn service general context
        - userdbConnectorType: ldapBind
        - userdbConnectorName: LDAP_Master

## *5.5 Back-end Tier*

### 5.5.1 Configuring the Storage

The external storage (SAN) must be connected to all the back-end nodes via SCSI cables or FiberChannel.

The next step is to configure the virtual disks (LUNs) on the storage that will be accessible from the back-end nodes. Depending on the storage or the desired result, the following storage configurations can be used:

- One virtual disk for each service instance (each AXIGEN service instance and the LDAP service). In this case, on each virtual disk, partitions will be created for each data folder required for each service. In our example, let's assume the disk is accessible from the back-end nodes with the device name: "/dev/sda" for LDAP, "/dev/sdb" for AXIGEN instance 1 and "/dev/sdc" for AXIGEN instance 2. The following partitions must be created (using fdisk on any of the back-end nodes)
    - LDAP

- /dev/sda1 – LDAP data partition (will be mounted in /var/lib/ldap)
  - AXIGEN instance 1
    - /dev/sdb1 – AXIGEN1 Storage (will be mounted in /var/opt/AXIGEN1/domains)
    - /dev/sdb2 – AXIGEN1 Queue (will be mounted in /var/opt/AXIGEN1/queue)
    - /dev/sdb3 – AXIGEN1 RunDir (will be mounted in /var/opt/AXIGEN1/run)
  - AXIGEN instance 2
    - /dev/sdc1 – AXIGEN2 Storage (will be mounted in /var/opt/AXIGEN2/domains)
    - /dev/sdc2 – AXIGEN2 Queue (will be mounted in /var/opt/AXIGEN2/queue)
    - /dev/sdc3 – AXIGEN2 RunDir (will be mounted in /var/opt/AXIGEN2/run)
- One virtual disk for all services. In this case, partitions are required for each service and for each data folder. In our example, let's assume that the single virtual disk on the storage is available on the back-end nodes as device name "/dev/sda":
  - LDAP
    - /dev/sda1 – LDAP data partition (will be mounted in /var/lib/ldap)
  - AXIGEN instance 1
    - /dev/sda2 – AXIGEN1 Storage (will be mounted in /var/opt/axigen1/domains)
    - /dev/sda3 – AXIGEN1 Queue (will be mounted in /var/opt/axigen1/queue)
    - /dev/sda4 – AXIGEN1 RunDir (will be mounted in /var/opt/axigen1/run)
  - AXIGEN instance 2
    - /dev/sda5 – AXIGEN2 Storage (will be mounted in /var/opt/AXIGEN2/domains)
    - /dev/sda6 – AXIGEN2 Queue (will be mounted in /var/opt/AXIGEN2/queue)
    - /dev/sda7 – AXIGEN2 RunDir (will be mounted in /var/opt/AXIGEN2/run)
- One virtual disk for each data folder required by the services. This is not a recommended scenario since it adds a lot of overload to the subsequent administration.

The first scenario is typically the best solution (also the one used in this document), because it allows isolation of each service to a storage virtual disk and, if the storage supports it, may allow selective availability of the disk on particular nodes; this feature can be used if some services will only be allowed to run on specific nodes (for instance, LDAP will run only on its home back-end node and on the hot-standby node).

### 5.5.2 Operating System Configuration

#### (1) Kernel Configuration

The same tuning must be performed for the back-end tier nodes as for the front-end tier nodes.

#### (2) DNS Resolver

Configure the system DNS resolver to use the DNS server that contains the appropriate zones; in our case, the DNS is on the router. The /etc/resolv.conf file contains, in our scenario:

```
Search backend.cluster
nameserver 10.0.0.1
```

### 5.5.3 Configuring the Network

Each node in the back-end tier must be configured according to the appropriate network setup. In our scenario:

- LDAP: 10.0.0.100
- AXIGEN node 1: 10.0.0.11
- AXIGEN node 2: 10.0.0.12

The gateway for all nodes is 10.0.0.1 (the back-end tier interface of the router).

### 5.5.4 Setting-up the Cluster Software

#### (1) Install RHCS packages

Install the RedHat Cluster packages according to RHCS user's manual.

#### (2) Configure the cluster

Use the "system-config-cluster" tool to create a new cluster configuration, with the following parameters (the information below is based on the example scenario defined in this document):

- Cluster settings:
  - Locking system: GULM
- Nodes:
  - In our setup, a physical node exists for each service instance plus one for LDAP and another one as hot standby
- Shared resources. The AXIGEN service script must be defined as a shared resource
  - Type: Script
  - Name: AXIGEN Script
  - Path: /etc/rc.d/init.d/axigen
- Services. One service must be created for each AXIGEN back-end service instance and one for the LDAP service (in our example, AXI1, AXI2 and LDAP)
  - AXIGEN Service instances
    - Service parameters

        - Service name: *AXIGEN<instance_number>*. Instance number may be 1, 2, aso. In our case, AXIGEN1 and AXIGEN2
    - Service Resources
        - Service IP Address: virtual IP for the service instance (in our example: 10.0.0.201 for AXI1, 10.0.0.202 for AXI2)
        - Service File Systems (associated with the partitions on the shared storage)
            - Storage partition: mounted in "/var/opt/<service_name>/domains"
            - Queue partition: mounted in "/var/opt/<service_name>/queue"
        - AXIGEN Service script (shared resource)
- LDAP Service
    - Service parameters
        - Service name: *LDAP*
    - Service resources
        - Service IP Address: virtual IP for the service instance (in our example: 10.0.0.210 for LDAP)
        - Service File Systems (associated with the partitions on the shared storage)
            - LDAP Data partition: mounted in "/var/lib/ldap"
        - LDAP Service script: /etc/rc.d/init.d/ldap

After configuring the cluster, copy the /etc/cluster/cluster.conf file on all the back-end nodes, preserving the permissions.

#### (3) Start the cluster services

On all the back-end nodes, start the cluster services according to the RHCS user manual.

### 5.5.5 Setting-up the LDAP Directory

#### (1) Install the LDAP packages

The following packages must be installed on the back-end nodes that will be used to run the LDAP service:

- openldap
- openldap-clients
- openldap-servers

#### (2) Configure the LDAP service

The *objectClass* that will be used to identify user accounts in LDAP will be **inetLocalMailRecipient**. The default LDAP configuration does not include the schema for this objectClass; it has to be explicitly included. Add the following line to the "slapd.conf" LDAP configuration file:

```
include /etc/openldap/schema/misc.schema
```

Configure the LDAP base, the administrative DN and the administrative DN's password. In our example, the base is "dc=example,dc=com" and the administrative DN is "cn=admin,dc=example,dc=com".

```
suffix "dc=example,dc=com"
rootdn "cn=admin,dc=example,dc=com"
rootpw secret
```

This example uses a simple plaintext password for the administrative DN; it is recommended to:

- Use a more complex password
- Define it encrypted in the configuration file (use the *saslpasswd* utility)

Copy the LDAP configuration file to all the back-end nodes where the LDAP service will be allowed to run (in our case, all the back-end nodes).

### 5.5.6 Setting-up AXIGEN

#### (1) Install the AXIGEN package

Install the appropriate AXIGEN package for the platform (*AXIGEN for RPM-based distros for GCC3*) on all the back-end nodes that will be used to run AXIGEN service instances. Depending on the version of the distribution, the "compat-libstdc++-33" package may also be required prior to installing the AXIGEN package.

After installation, disable the AXIGEN automatic startup at boot time by using the "chkconfig" command or the "setup" utility.

Do not run the AXIGEN install wizard as it will perform unnecessary tasks and also attempt to start AXIGEN which are not needed for this type of setup.

Due to the fact that each service instance may 'float' on different nodes (the cluster software will relocate the service on a different node in the event of a failure), some instance-specific files (such as the storage, queue, and rundir – configuration and pidfile) must reside on partitions of the shared storage.

#### (2) Configure AXIGEN

For **each AXIGEN service instance**, temporarily mount the "RunDir" service partition and create a copy of the default configuration file on it. Modify the following parameter:

```
queuePath = /var/opt/AXIGEN<instance>/run
```

Replace <instance> with the actual service instance number (i.e. /var/opt/AXIGEN1/run). Each AXIGEN service instance will now use its queue directory on the shared storage. Change the ownership of the mounted RunDir directory to user 'axigen', group 'axigen':

```
chown axigen.axigen <rundir_partition_mountpoint>
```

Remember to unmount the RunDir partition – it will be automatically mounted by the cluster software when a service is started.

For **each AXIGEN service instance**, temporarily mount (for instance, on one back-end node in the /var/opt/<service_name>/queue and /var/opt/<service_name>/queue respectively) the "Queue" and "Storage" service partition and change their ownership to user 'axigen' group 'axigen':

```
chown axigen.axigen <queue_partition_mountpoint>
chown axigen.axigen <storage_partition_mountpoint>
```

Remember to unmount both partitions after the ownership change – they will be automatically mounted by the cluster software when a service is started.

For **each back-end node** that will run AXIGEN, modify the /etc/sysconfig/axigen file changing the following parameters:

```
PIDFILE ="/var/opt/$OCF_RESKEY_service_name/run/axigen.pid"
AXIOPT="-C  /var/opt/$OCF_RESKEY_service_name/run/axigen.cfg -W
/var/opt/$OCF_RESKEY_service_name/"
```

The "$OCF_RESKEY_service_name" environment variable will be filled-in by the cluster software with the actual name of the service (AXIGEN1, AXIGEN2, etc).

### 5.5.7 Starting the Services

Use the *clusvcadm* utility to start the service instances on the specific back-end nodes:

```
clusvcadm –E LDAP –m backend_ldap_node
clusvcadm –E AXIGEN2 –m backend_axigen1_node
clusvcadm –E AXIGEN2 –m backend_axigen2_node
```

### 5.5.8 Configuring the AXIGEN Service Instances

#### (1) Listeners

On each AXIGEN service instance, the SMTP, POP3, WebMail and IMAP services must be configured to bind on the specific service instance IP.

In our case, the configuration for each service instance is the following:

- AXIGEN1
  - SMTP – one listener, 10.0.0.201 on port 25
  - POP3 – one listener, 10.0.0.201 on port 110
  - IMAP – one listener, 10.0.0.201 on port 143
  - WebMail – one listener, 10.0.0.201 on port 80
  - WebAdmin – one listener, 10.0.0.201 on port 9000
  - CLI – one listener, 10.0.0.201 on port 7000
  - FTP Backup – one listener, 10.0.0.201 on port 21
- AXIGEN2
  - SMTP – one listener, 10.0.0.202 on port 25
  - POP3 – one listener, 10.0.0.202 on port 110
  - IMAP – one listener, 10.0.0.202 on port 143
  - WebMail – one listener, 10.0.0.202 on port 80
  - WebAdmin – one listener, 10.0.0.202 on port 9000
  - CLI – one listener, 10.0.0.202 on port 7000
  - FTP Backup – one listener, 10.0.0.202 on port 21

#### (2) Domain Name Resolver

On all service instances, add the following nameserver entry (in the DNR configuration)

- Priority: 5
- Address: 10.0.0.1 (if the DNS is located on the router, as in our example scenario)

- Timeout: 2
- No. of retries: 3

### (3) LDAP authentication

- Define LDAP Connector
  - In the UserDB section, create an LDAP Connector with the following parameters:
    - Name: LDAP_Master
    - IP: specific LDAP IP (in our example, *ldap://10.0.0.100*)
    - Port: specific LDAP port (in our example 389)
    - Server type: OpenLDAP
    - Make sure the “Enable Clustered Operations” box is checked
    - Synchronization direction: AXIGEN to LDAP
    - Conflict resolution: AXIGEN wins
    - Enable “Administrative DN”
    - AdminDN: LDAP Administrator’s DN
      (in our example: *cn=admin,dc=example,dc=com*)
    - Admin DN Password: LDAP Administrator’s password
      (as defined in the LDAP configuration file)
    - Account Base DN: LDAP base of the user entries
      (in our example: *dc=example,dc=com*)
    - Hostname field: LDAP attribute holding the hostname of the back-end node where the account resides (in our case *“mailHost”*)
- Enable LDAP authentication for POP3 and IMAP
  - Configure, for both the POP3 and IMAP services:
    - userdbConnectorType: ldapBind
    - userdbConnectorName: LDAP_Master

Make sure that after the domains are created, they are also configured for sync with the LDAP connector defined in the above section. The LDAP synchronization is only performed in the back-end (where the storage is located) because this is where the data is saved. Do not configure domains and LDAP syncs in the front-end layer.

### (4) Logging

Ideally, a separate log server must be used and all AXIGEN services must send log entries through the log service, via the network. In our example, we will log locally (on each back-end node), making sure that the log files’ names are unique for each AXIGEN service instance.

- Make sure that, for each instance, the log files’ names contain the AXIGEN service instance name (so that they will not mix). The default log files are:
  - **everything.txt** – we shall rename it to *everything_<service instance name>.txt* (i.e. everything_AXIGEN1.txt and everything_AXIGEN2.txt)
  - **default.txt** – we shall rename it to *default_<service instance name>.txt* (i.e. default_AXIGEN1.txt and default_AXIGEN2.txt)

# 6 Provisioning

When creating a new account, one back-end AXIGEN service instance must be selected. The provisioning utility must be implemented to select the back-end service instance based on one of the following algorithms:

- Random
    - Each new account is created on one of the back-end service instances, picked randomly;
    - As an enhancement, a weighted-random distribution algorithm may be used to allow creating more accounts on some of the back-end service instances than on others.
- Least used
    - The provisioning interface must be aware of the number of accounts that exist on each service instance, so that, each time a new account is created, the back-end service instance that has the least number of accounts is used.
- Domain based
    - Each domain is placed on one of the back-end service instances; the provisioning interface must have configured a domain/back-end service instance table in order to be able to select a specific back-end service instance when creating a new account;
    - Each domain will have a 'home' back-end service instance.

The first and second distribution algorithms have the advantage of a better spread of the accounts to the back-end service instances. The disadvantage resides in the fact that each domain must be created on all the back-end service instances and that domain-wide settings for each domain must be kept in sync on all the back-end service instances.

The third distribution algorithm simplifies the management of the accounts (the domain is only created on the specific back-end service instance that will host that domain; changes to domain configuration are performed only on the domain-home back-end service instance); moreover, routing can be performed with a much simpler LDAP configuration (i.e. one entry per domain instead of one entry per account).