

Linuxha.net
Step-by-step Guide:
BIND DNS

Introduction

The purpose of this document is to recommend procedures for creating a clustered BIND DNS configuration with Linuxha.net. These procedures were tested using the following configuration:

- Fedora Core 5
- DRBD version 0.7.20
- Linuxha.net version 1.2
- BIND version 9.3.2

The following conventions are used throughout this document:

# ls	Single-line command entered as root user.
# useradd \ > --home-dir /home2/xyz \ > --gid abc \ > xyz	Multi-line command entered as root user.
\$ ls	Single-line command entered as non-root user
\$ cut \ > --delimiter=":" \ > --fields=1,5 \ / etc/passwd	Multi-line command entered as non-root user.
127.0.0.1 localhost 192.168.1.32 fc5s1 192.168.1.33 fc5s2 192.168.100.32 fc5s1b 192.168.100.33 fc5s2b	Command output or file contents

User Account Creation

The BIND DNS processes will run with the **named** user credentials. The **named** user and group must exist on both nodes and have the same user and groups ids.

All commands in this section are executed as **root** on both nodes, unless otherwise indicated.

1. Check whether the **named** user exists and that the user and group ids are the same on both nodes.

```
# grep named /etc/passwd
```

If there is no result continue to step 2, since the **named** user does not exist. Otherwise, compare the user and group ids on both nodes. These are the third and fourth fields respectively of the user account record as shown below:

```
named:x:25:25:Named:/var/named:/sbin/nologin
```

If the user and group ids are not the same on both nodes, proceed to step 6. Otherwise go to the next section.

2. Identify the next available group id.

```
# expr `cut --delimiter=":" --fields=3 /etc/group | \  
> grep --regexp="^[5-9][0-9][0-9]$" | sort --numeric-sort|tail -1` + 1
```

3. Create **named** group and assign the larger of the group ids returned in step 2, e.g. 508.

```
# groupadd --gid 508 named
```

4. Identify the next available user id.

```
# expr `cut --delimiter=":" --fields=3 /etc/passw | \  
> grep --regexp="^[5-9][0-9][0-9]$" | sort --numeric-sort|tail -1` + 1
```

5. Create **named** user and assign the larger of the user ids returned in step 4, e.g. 508.

```
# useradd --home-dir /var/named -M \  
> --comment "Named" --gid named --uid 508 \  
> named
```

Proceed to the next section.

6. If the **named** group ids are different, change the group id on **one of the nodes** to be the same as the other, e.g. 49. Ensure that the new group id is not already in use.

```
# grep 26 /etc/group # check that group id 26 is not in use  
# groupmod -g 26 named # change group id to 26
```

7. If the **named** user ids are different, change the user id on **one of the nodes** to be the same as the other, e.g. 26. Ensure that the new user id is not already in use.

```
# grep 26 /etc/passwd # check that user id 26 is not in use  
# usermod --uid 26 --gid named named # change user id to 26
```

8. Change the user and group ownership of files owned by **named**.

The following commands will find all files whose user / group ownership is 25 and change their user / group ownership to **named**.

```
# find /usr -gid 25 -exec chgrp named {} \  
# find /usr -uid 25 -exec chown named {} \  
#
```

Installation

This section provides basic instructions for downloading and installing the BIND DNS version 9.3.2 from i386 binary RPM. Instructions for installing or upgrading using other package formats or from source are beyond the scope of this guide.

All steps are to be performed on both nodes as **root** unless otherwise indicated.

1. Check whether BIND DNS is installed, by executing

```
# find /usr -name named -print
```

If a result similar to the following is returned, proceed to step 4. Otherwise, continue to step 2 since BIND DNS is not installed

```
/usr/sbin/named
```

2. Download BIND DNS RPM

To download the Fedora Core 5 i386 RPM, execute the following **single-line** command:

```
# wget http://www.mirror.ac.uk/mirror/fedora.redhat.com/updates/5/i386/bind-9.3.2-20.FC5.i386.rpm
```

3. Install BIND DNS.

```
# rpm --hash --install bind-9.3.2-20.FC5.i386.rpm
```

4. Configure firewall

If the firewall is running, it needs to be configured to allow access to BIND DNS port (default: port 53), by inserting the following into **/etc/syconfig/iptables**:

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 53 -j ACCEPT
```

and restarting the firewall by executing

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

Directories / Files For Replication

The zone database directory will be replicated. Its location is defined by the *directory* parameter in the *options* section of **/etc/named.conf**, as shown in the example below:

```
options {
    directory "/var/named";
    dump-file "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
};
```

Create Replicated File System

All commands are to be executed as **root** on both nodes. Steps 5 to 6 may be omitted on the secondary node.

1. Use **fdisk** to create a partition (*/dev/sda6*) large enough to store all DNS zone files.

2. Initialise the physical volume:

```
# pvcreate /dev/sda6
```

3. Create the volume group (*bindvg*):

```
# vgcreate bindvg /dev/sda6
```

4. Create the mount point (*/named*):

```
# mkdir /named
```

5. Create logical volume (*namedlv*):

```
# lvcreate --size 256M --name namedlv bindvg
```

6. Create file system on logical volume:

```
# mkfs -t ext3 /dev/bindvg/namedlv
```

Populate Replicated File System

All commands are executed as **root** on the primary node unless otherwise specified.

1. Shut down BIND DNS (**both nodes**).

```
# /etc/init.d/named stop
```

2. Mount the replicated file system (*namedlv*).

```
# mount -t ext3 /dev/bindvg/namedlv /named
```

3. Create directory structure

```
# mkdir /named/dns
# chgrp named /named/dns
# mkdir /named/scripts
# touch /named/scripts/startapp /named/scripts/stopapp
# chmod u+x /named/scripts/*
```

4. Copy BIND DNS configuration and zone files, if they exist, to *namedlv*.

```
# cp /etc/named.conf /named
# cd /var/named
# find . -print | cpio --pass-through /named/dns
```

Configuration

These commands are executed as **root** on the primary node.

Existing installation

1. Insert the following into **/named/scripts/startapp**:

```
/usr/sbin/named -u named -c /named/named.conf
```

2. Insert the following into **/named/scripts/stopapp**:

```
kill `cat /var/run/named/named.pid`
```

3. Modify BIND configuration to reflect new directory structure, and to restrict the server to respond to requests on the cluster virtual IP address (*192.168.1.55*)

Change the *options* section of **/named/named.conf** as shown in the table below:

Directive	Value
directory	/named/dns
dump-file	data/cache_dump.db
listen-on	{127.0.0.1; 192.168.1.55; }

New Installation

These instructions describe how to set up a basic BIND DNS configuration for testing purposes.

1. Create **/named/scripts/startapp** as shown:

```
/usr/sbin/named -u named -c /named/named.conf
```

2. Create **/named/scripts/stopapp** as shown:

```
kill `cat /var/run/named/named.pid`
```

3. Create **/named/named.conf** as follows:

```
options {
    directory "/named/dns";
    dump-file "data/cache_dump.db";
    allow-query { any; };
}
```

```

recursion yes;
listen-on { 127.0.0.1; 192.168.1.55; };
};
zone "linuxha.test" {
    type master;
    file "linuxha.test.zone";
};

```

4. Create **/named/dns/linuxha.test.zone** as follows:

```

$TTL      86400
@ in soa localhost. root 2 3H 15M 1W 1D
    ns localhost.
node1 in A 192.168.1.132
node2 in A 192.168.1.133

```

Build Application

The commands in this section are to be executed as **root** on the primary node.

1. Create directory **/etc/cluster/named**.

```
# mkdir /etc/cluster/named
```

2. Create **/etc/cluster/named/appconf.xml** as shown:

```

<appconf>
  <global>
    <version>0.1</version>
    <name>named</name>
    <takeover>normal</takeover>
    <syncrate>2000</syncrate>
    <preferred_node>fc5s1</preferred_node>
  </global>

  <networks>
    <network net="main" ip="192.168.1.55" netmask="255.255.255.0"/>
  </networks>

  <vg>
    <name>bindvg</name>
    <type>filesystems</type>
  </vg>

  <application>
    <startscript>/named/scripts/startapp</startscript>
    <stopscript>/named/scripts/stopapp</stopscript>
    <maxstoptime>10</maxstoptime>
    <maxstarttime>20</maxstarttime>
  </application>
</appconf>

```

The bolded values are configuration-specific, as described in the following table:

Entry	Value
global/preferred_node	Host name of one of the nodes in the cluster, or LEAST_CPU_LOAD
networks/network.net	Same as one of <i>node/network.name</i> in /etc/cluster/clconf.xml
networks/network.ip	Virtual IP address of application

Entry	Value
networks/network.netmask	Netmask corresponding to virtual IP address
vg/name	Name of application volume group

3. Create **/etc/cluster/named/lems.local.xml** as shown:

```
<?xml version="1.0"?>
<lems_config>
  <globals modules="/sbin/cluster/lems/modules"
    programs="/sbin/cluster/lems/programs"
    logs="/var/log/cluster/lems"
  />

  <check>
    <name>flag_check</name>
    <type>internal</type>
    <module>flag_check named</module>
    <interval>5</interval>
    <action_list>
      <action rc="0" action="NOP"/>
      <action rc="1" action="%RCDATA%"/>
      <action rc="2" action="ABORT"/>
    </action_list>
  </check>
  <check>
    <name>named</name>
    <type>internal</type>
    <module>procmon /etc/cluster/named/named.xml</module>
    <interval>10</interval>
    <action_list>
      <action rc="0" action="NOP"/>
      <action rc="1" action="STOP"/>
      <action rc="2" action="FAILOVER"/>
    </action_list>
  </check>
  <check>
    <name>ip</name>
    <type>internal</type>
    <module>ip_module named</module>
    <interval>10</interval>
    <action_list>
      <action rc="0" action="NOP"/>
      <action rc="1" action="RUN move_ip"/>
      <action rc="2" action="STOP"/>
    </action_list>
  </check>
  <check>
    <name>fsmonitor</name>
    <type>internal</type>
    <module>fsmon named</module>
    <interval>10</interval>
    <action_list>
      <action rc="0" action="NOP"/>
      <action rc="1" action="PAUSE 30"/>
      <action rc="2" action="STOP"/>
      <action rc="3" action="FAILOVER"/>
      <action rc="10" action="PAUSE 60"/>
    </action_list>
  </check>
</lems_config>
```

4. Create `/etc/cluster/named/named.xml` as shown:

```
<?xml version="1.0"?>
<procmon>
  <global>
    <logdir>/var/log/cluster</logdir>
    <restarts>5</restarts>
    <resetwindow>3600</resetwindow>
    <restartcmd>
      /named/scripts/stopapp ; /named/scripts/startapp
    </restartcmd>
  </global>
  <process>
    <label>BIND DNS</label>
    <user>named</user>
    <process_string>
      /usr/sbin/named -c /named/named.conf *
    </process_string>
    <min_count>1</min_count>
    <max_count>1</max_count>
  </process>
</procmon>
```

5. Verify LEMS configuration by executing:

```
# lems.pl --config /etc/cluster/named/lems.local.xml \
> --application named --verbose --check --file /dev/tty
```

If successful, a result similar to the following will be displayed.

```
INFO 07/10/2006 17:22:30 Using modules from : /sbin/cluster/lems/modules
INFO 07/10/2006 17:22:30 Using programs from : /sbin/cluster/lems/programs
INFO 07/10/2006 17:22:30 Writing logs to : /var/log/cluster/lems
INFO 07/10/2006 17:22:30 Listening on port : 9904
INFO 07/10/2006 17:22:30 Global initialisation complete.
INFO 07/10/2006 17:22:30 Started local server on port 9904
INFO 07/10/2006 17:22:30 Validating monitor entry named...
INFO 07/10/2006 17:22:30 Validated monitor entry named successfully.
INFO 07/10/2006 17:22:30 Validating monitor entry ip...
INFO 07/10/2006 17:22:30 Validated monitor entry ip successfully.
INFO 07/10/2006 17:22:30 Validating monitor entry fsmonitor...
INFO 07/10/2006 17:22:30 Validated monitor entry fsmonitor successfully.
INFO 07/10/2006 17:22:30 Validating monitor entry flag_check...
INFO 07/10/2006 17:22:30 Validated monitor entry flag_check successfully.
INFO 07/10/2006 17:22:30 Check mode - transferring validated config to remote node.
INFO 07/10/2006 17:22:31 Configuration transferred successfully.
INFO 07/10/2006 17:22:31 Calculated a check interval of 2.5 seconds.
```

6. Build **named**.

```
# clbuildapp --application named --sync
```

The output of a successful build is shown below:

```
INFO 07/10/2006 15:57:17 Backups directory defaulted to /clbackup
INFO 07/10/2006 15:57:17
INFO 07/10/2006 15:57:17 Validation of Application 'named' started.
INFO 07/10/2006 15:57:17 ['/var/log/cluster/build/named-check-300610071557.log']
INFO 07/10/2006 15:57:19 Initial Validation of Application successful.
INFO 07/10/2006 15:57:19
INFO 07/10/2006 15:57:19 NOTE: Build of new application is being performed.
INFO 07/10/2006 15:57:19
INFO 07/10/2006 15:57:19 Host Environment Validation started.
INFO 07/10/2006 15:57:19 ['/var/log/cluster/build/named-envcheck-300610071557.log']
INFO 07/10/2006 15:57:20 Host Environment Validation successful.
INFO 07/10/2006 15:57:20
```

```

INFO 07/10/2006 15:57:20 Cluster state      : DOWN
INFO 07/10/2006 15:57:20 Application state: UNDEFINED
INFO 07/10/2006 15:57:20
INFO 07/10/2006 15:57:20 Volume Group Configuration started.
INFO 07/10/2006 15:57:20   ['/var/log/cluster/build/named-lvm-300610071557.log']
INFO 07/10/2006 15:57:25 Volume Group Configuration successful.
INFO 07/10/2006 15:57:25
INFO 07/10/2006 15:57:25 Application Resource Allocation started.
INFO 07/10/2006 15:57:25   ['/var/log/cluster/build/named-build-300610071557.log']
INFO 07/10/2006 15:57:31 Application Resource Allocation successful.
INFO 07/10/2006 15:57:31
INFO 07/10/2006 15:57:31 Application Data Synchronisation started.
INFO 07/10/2006 15:57:31   ['/var/log/cluster/build/named-syncdata-
300610071557.log']
Storage Syncing:      256Mb/          0Mb [0.0 % Complete]
Storage Syncing:      0Mb/           0Mb [100 % Complete]
INFO 07/10/2006 15:59:54 Application Data Synchronisation successful.

```

Run Application

The commands in this section are executed as **root** on either node except where indicated.

1. If necessary, form the cluster

```
# clform
```

2. Start **named** on the primary node (*fc5s1*).

```
# clrunapp --application named --node fc5s1
```

3. Verify the application state using **clstat**.

```
# clstat --application named
```

A result similar to the following will be returned if **named** is running:

```

Cluster: cluster1 - UP

Application      Node      State  Runnig  Monitor  Stale  Fail-over?
   named        fc5s1   STARTED  0:00:00  Running    0      Yes

File Systems

Mount Point      Valid  Type    State  % Complete  Completion
/named           both   drbd    Sync

Process Monitors

   Name  Status  Restarts  Current  Reset at
   named  Running    5         0        N/A

General Monitors

   Type      Name      Status
Flag Check  flag_check  Running
FS Monitor  fsmonitor  Running
IP Monitor  ip         Running

```

4. On the node where **named** is running, list the **named** processes by executing:

```
# ps -u named -f
```

An output similar to the following process list should be returned.

```
named 67721 0 17:23 ? 00:00:00 /usr/sbin/named -u named -c /named/named.conf
```

5. Verify that the BIND server is running by using **nslookup** to resolve a network name, for example:

```
# nslookup node1.linuxha.test 192.168.1.55
```

should return

```
Server:          192.168.1.55
Address:         192.168.1.55#53

Name:   node1.linuxha.test
Address: 192.168.1.132
```

6. On the node where **named** is running, stop it by executing:

```
# clhaltapp --application named
```

7. Start **named** on the secondary node (*fc5s2*).

```
# clrunapp --application named --node fc5s2
```

8. Repeat steps 3 to 6 to test **named** on the secondary node.