Exprement Name: Install and configure DNS server using Windows or Linux platform.

Theory and Configuration:

A DNS server, or name server, is used to resolve an IP address to a hostname or vice versa.

You can set up four different types of DNS servers:

- A **master DNS server for your domain(s),** which stores authoritative records for your domain.
- A **slave DNS server,** which relies on a master DNS server for data.
- A **caching-only DNS server,** which stores recent requests like a proxy server. It otherwise refers to other DNS servers.
- A **forwarding-only DNS server,** which refers all requests to other DNS servers.

Before configuring BIND to create a DNS server, you must understand some basic DNS concepts.

The entire hostname with its domain such as **server.example.com** is called a fully qualified domain name (FQDN). The right-most part of the FQDN such as .com or .net is called the **top level domain,** with the remaining parts of the FQDN, which are separated by periods, being sub-domains.

These sub-domains are used to divide FQDNs into zones, with the DNS information for each zone being maintained by at least one **authoritative name server.**

The authoritative server that contains the master zone file, which can be modified to update DNS information about the zone, is called the **primary master server,** or just **master server.**

The additional name servers for the zone are called **secondary servers** or **slave servers.** Secondary servers retrieve information about the zone through a zone transfer from the master server or from another secondary server. DNS information about a zone is never modified directly on the secondary server

chroot features

chroot feature is run named as user **named**, and it also limit the files named can see. When installed, **named** is fooled into thinking that the directory **/var/named/chroot** is actually the **root or /** directory. Therefore, named files normally found in the **/etc** directory are found in **/var/named/chroot/etc** directory instead, and those you would expect to find in **/var/named** are actually located in **/var/named/chroot/var/named.**

The advantage of the chroot feature is that if a hacker enters your system via a BIND exploit, the hacker's access to the rest of your system is isolated to the files under the chroot directory and nothing else. This type of security is also known as a chroot jail.

Implementation and Testing:

Configure dns server

In this example we will configure a dns server and will test from client side.

For this example we are using three systems one linux server one linux clients and one window clients.

bind and **caching-nameserver** rpm is required to configure dns. check them for install if not found install them.

```
[root@Server ~]# rpm -qa bind*
bind-libs-9.3.3-10.el5
bind-chroot-9.3.3-10.el5
bind-devel-9.3.3-10.el5
bind-utils-9.3.3-10.el5
bind-libbind-devel-9.3.3-10.el5
bind-9.3.3-10.el5
bind-sdb-9.3.3-10.el5
[root@Server ~]# rpm -qa cach*
caching-nameserver-9.3.3-10.el5
cachefilesd-0.8-2.el5
[root@Server ~]# _
```

set hostname to **server.example.com** and ip address to **192.168.0.254**

```
[root@Server ~]# cat /etc/sysconfig/network
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=Server.example.com

[root@Server ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:11:AD:E1
          inet addr:192.168.0.254  Bcast:192.168.0.255  Mask:
          inet6 addr: fe80::20c:29ff:fe11:ade1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:17981 (17.5 KiB)
          Interrupt:67 Base address:0x2000
```

main configuration file for dns server is **named.conf.** By default this file is not created in **/var/named/chroot/etc/** directory. Instead of named.conf a sample file **/var/named/chroot/etc/named.caching-nameserver.conf** is created. This file is use to make a caching only name server. You can also do editing in this file after changing its name to **named.conf** to configure master dns server or you can manually create a new **named.conf** file.

In our example we are creating a new named.conf file

```
[root@Server etc]# vi /var/named/chroot/etc/named.conf _
```

We are using bind's **chroot** features so all our necessary files will be located in chroot directory. Set directory location to **/var/named.** Further we will set the location of **forward zone** and **reverse lookup zone** files.

Do editing exactly as shown here in image

```
options{
        directory  "/var/named/";
};

zone "example.com" {
       type master;
       file "example.com.zone";
       allow-transfer {192.168.0.1;};
};
zone "0.168.192.in-addr.arpa" {
      type master;
      file "0.168.192.in-addr.arpa.zone";
};
```

save this file with **:wq** and exit

Configure zone file

We have defined two zone files **example.com.zone** for forward zone and **0.168.192.in-addr.arpa** for reverse zone. These files will be store in **/var/named/chroot/var/named/** location. We will use two sample files for creating these files.

Change directory to **/var/named/chroot/var/named** and copy the sample files to name which we have set in named.conf

```
[root@Server named]# cd /var/named/chroot/var/named
[root@Server named]# cp localhost.zone example.com.zone
[root@Server named]# cp named.local 0.168.192.in-addr.arpa.zone
[root@Server named]# _
```

Now open forward zone file **example.com.zone**

```
[root@Server named]# vi example.com.zone _
```

By default this file will look like this

```
$TTL    86400
@               IN SOA  @       root (
                                        42              ; serial
                                        3H              ; refresh
                                        15M             ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                IN NS           @
                IN A            127.0.0.1
                IN AAAA         ::1
```

Change this file exactly as shown in image below

```
$TTL    86400
@               SOA         example.com.      root (
                                        42              ; serial
                                        3H              ; refresh
                                        15M             ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

@               NS              server.example.com.
@               NS              client1.client.com.
server          A               192.168.0.254
client1         A               192.168.0.1
client2         A               192.168.0.2
```

Now open reverse lookup zone file **0.168.192.in-addr.arpa**

```
[root@Server named]# vi 0.168.192.in-addr.arpa.zone _
```

By default this file will look like this

```
$TTL    86400
@       IN      SOA     localhost. root.localhost.  (
                                        1997022700 ; Serial
                                        28800      ; Refresh
                                        14400      ; Retry
                                        3600000    ; Expire
                                        86400 )    ; Minimum
        IN      NS      localhost.
1       IN      PTR     localhost.
```

Change this file exactly as shown in image below

```
$TTL    86400
@             SOA     example.com. root.server.example.com.  (
                                        1997022700 ; Serial
                                        28800      ; Refresh
                                        14400      ; Retry
                                        3600000    ; Expire
                                        86400 )    ; Minimum

          IN      NS      server.example.com
254       IN      PTR     server.example.com.
1         IN      PTR     client1.example.com.
2         IN      PTR     client2.
```

Now changed the ownership of these zone files to **named** group

```
[root@Server named]# chgrp named example.com.zone
[root@Server named]# chgrp named 0.168.192.in-addr.arpa.zone
[root@Server named]# _
```

Now start the named service

```
[root@Server named]# chkconfig named on
[root@Server named]# service named restart
Stopping named:                                            [  OK  ]
Starting named:                                            [  OK  ]
[root@Server named]# _
```

If service restart without any error means you have successfully configured master name server.

Configure dns slave server

For this example we are using three systems one linux server one linux clients and one window clients.

We have configured master DNS server with ip address of **192.168.0.254** and hostname **server.example.com** on linux server. Now we will configure **slave DNS server** on linux clients

To configure slave DNS server go on client1 system.

First test connectivity from dns server by ping commands and check necessary rpm. **bind** and **caching-nameserver** rpm is required to configure dns. check them for install if not found install them.

```
[root@Server ~]# rpm -qa bind*
bind-libs-9.3.3-10.el5
bind-chroot-9.3.3-10.el5
bind-devel-9.3.3-10.el5
bind-utils-9.3.3-10.el5
bind-libbind-devel-9.3.3-10.el5
bind-9.3.3-10.el5
bind-sdb-9.3.3-10.el5
[root@Server ~]# rpm -qa cach*
caching-nameserver-9.3.3-10.el5
cachefilesd-0.8-2.el5
[root@Server ~]# _
```

set hostname to **client1** and ip address to **192.168.0.1** And create a new **named.conf** file

```
[root@Client1 ~]# vi /var/named/chroot/etc/named.conf_
```

We are using bind's **chroot** features so all our necessary files will be located in chroot directory. Set directory location to **/var/named.** As we are configuring **slave server** so we need not to define the location of zone database files. Zone database file can be created and modified only on master server. A **slave server** only copied it's from master server.

Do editing exactly as shown here in image in **named.conf**

```
options{
        directory "/var/named/";
};
zone "example.com" IN {
      type slave;
      masters {192.168.0.254;};
      file "slave/example.com.zone";
};
```

save this file with **:wq** and exit

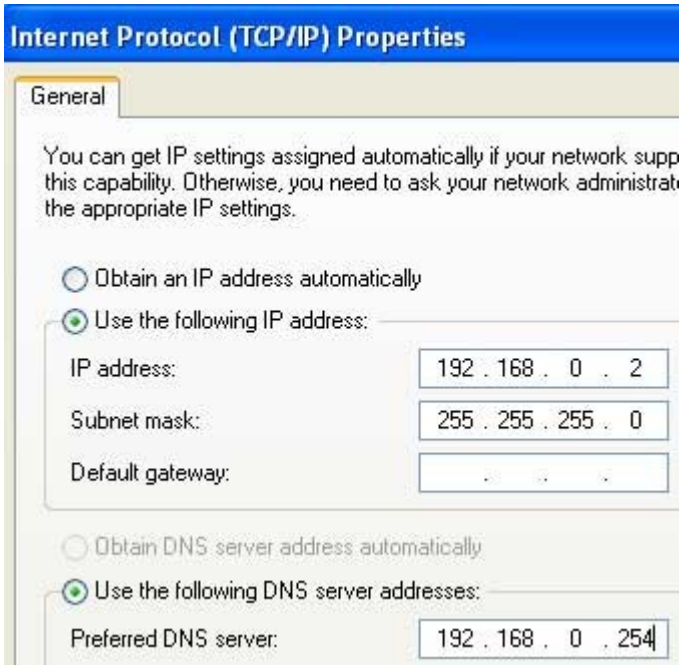
Now restart the named service. It should be start without any error.

```
[root@Client1 ~]# service named restart
Stopping named:                                            [FAILED]
Starting named:                                            [  OK  ]
[root@Client1 ~]# _
```

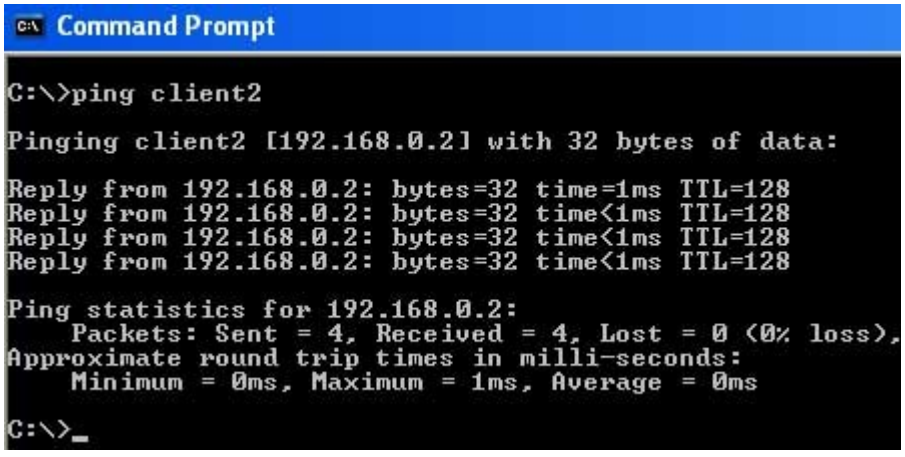
Congratulation you have configured both Master and client DNS server. Now we will configure dns client and test it with dns server.

Configure Window DNS Client

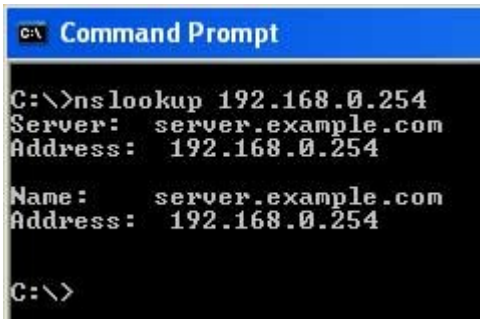
Now go on windows xp system and test connectivity from DNS server. And set **DNS ip address** in LAN card properties.



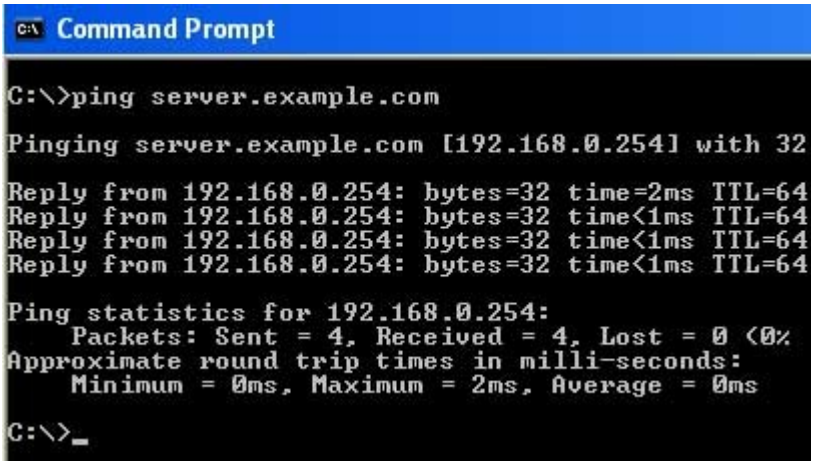
Now go on **commands prompt** and ping from other client by name to test **dns**.



Alternately You can also verify **DNS server** by **nslookup** command



Test also by **pinging server** from name



Configure Linux DNS clients

RHCE Exam question Dig Server.example.com, Resolve to successfully through DNS Where DNS server is 192.168.0.254.

RHCE Exam question2

Your System is configured in 192.168.0.0/24 Network and your nameserver is 192.168.0.254. Make successfully resolve to server.example.com.

On command line interface you don't have any options to set DNS ip in network configuration window. IP of DNS server can be set from **/etc/resolv.conf** file. Each nameserver line represents a DNS server, and the search line specifies domain names to try if only the first part of a hostname is used. For example, if just the name client1 is used as a hostname, **client1. example.com** will also be tried if the /etc/resolv.conf file is configured as shown in image below on the system.

To set DNS ip open **/etc/resolv.conf** file

set **nameserver** ip to **192.168.0.254** and **search** option to **example.com**

After saving **/etc/resolv.conf** file restart the network service

```
[root@Client1 ~]# service network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
[root@Client1 ~]# _
```

dig **server.example.com** to test dns server

```
[root@Client1 ~]# dig server.example.com

; <<>> DiG 9.3.3rc2 <<>> server.example.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERR
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1,

;; QUESTION SECTION:
;server.example.com.            IN      A
```

now verify by pinging to other client from name

```
[root@Client1 ~]# ping client2
PING client2.example.com (192.168.0.2) 56(84) bytes of data.
64 bytes from client2 (192.168.0.2): icmp_seq=1 ttl=128 time=16.3 ms
64 bytes from client2 (192.168.0.2): icmp_seq=2 ttl=128 time=0.383 ms

--- client2.example.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.383/8.348/16.313/7.965 ms
[root@Client1 ~]# _
```