Setup DNS Server On Debian 7 Wheezy

by SK

What is DNS Server?

As you might know, **DNS**, **Domain Name System**, is a system that resolves Domain names into IP Addresses and vice-versa. For example if you type www.unixmen.com in your browser, the computer doesn't know actually where it is. So your computer will contact the current DNS server to find where www.unixmen.com is. The DNS servers simply reply to your computer with the associated IP address of unixmen.com domain. Since IP address of every domain is hard to remember, DNS Server makes it easy to remember domain names instead of their IP addresses. You can get the DNS server details from your Internet Service Providers.

Types of DNS Server

There are two types of DNS Servers, **Primary** and **Secondary DNS** server. These two DNS servers are computers where the domain names with their respective IP addresses are stored and the information available on both DNS servers is identical. The values for these two DNS's are configured by the Internet Service Provider (ISP). The secondary DNS acts as backup for the Primary DNS if Primary DNS server is down.

Scenario

In this tutorial i am going to setup local Primary and secondary DNS servers. I am using two systems running with **Debian 7** for primary and secondary DNS server and one system running with **Ubuntu 13.04 Desktop** for Client.

Primary DNS server details:

```
Operating System: Debian 7 Wheezy Server
Hostname: master.unixmen.local
IP Address: 192.168.1.200/24
```

Secondary DNS Server Details:

```
Operating System: Debian 7 Wheezy Desktop
Hostname: slave.unixmen.local
IP Address: 192.168.1.201/24
```

Client Details:

```
Operating System: Ubuntu 13.04 Desktop
Hostname: client.unixmen.local
IP Address: 192.168.1.100/24
```

Note: I tested this how-to twice on local area network (LAN). It is worked for me as i expected. If you want to setup external DNS server, use your public ip in zone files and dns configuration files.

1. Setup Primary DNS Server

Install bind9

BIND (Berkely Internet Name Domain) is a software for translating domain names into IP addresses.

Install it using command:

```
apt-get install bind9 bind9utils bind9-doc dnsutils
```

The DNS configuration files are stored in the **/etc/bind** directory. The primary configuration file is **/etc/bind/named.conf**.

Configure Bind9

Open up the file **/etc/bind/named.conf.local** in any editor.

```
nano /etc/bind/named.conf.local
```

Add the following lines to define forward and reverse zone files.

```
zone    "unixmen.local"   {
        type master;
        file    "/etc/bind/for.unixmen.local";
 };

zone   "1.168.192.in-addr.arpa"        {
       type master;
       file    "/etc/bind/rev.unixmen.local";
```

```
};
```

Save and close the file.

Create Zone Files

Now create the forward and reverse zone files which we defined in the above step.

1. Forward Zone file

Copy the existing zone file template **/etc/bind/db.local** to create a new forward zone file.

```
cp /etc/bind/db.local /etc/bind/for.unixmen.local
```

Now edit the file **/etc/bind/for.unixmen.local**,

```
nano /etc/bind/for.unixmen.local
```

Change the contents as shown below and replace the domain name and ip address with your own.

```
;
; BIND data file for forward.unixmen.local zone
;
$TTL    604800
@       IN      SOA     master.unixmen.local. root.unixmen.local. (
2         ; Serial
604800         ; Refresh
86400         ; Retry
2419200         ; Expire
604800 )       ; Negative Cache TTL
         IN      A       192.168.1.200
;
@       IN      NS      master.unixmen.local.
@       IN      NS      slave.unixmen.local.
@       IN      A       192.168.1.200
@       IN      A       192.168.1.201
@       IN      A       192.168.1.100
```

```
@       IN      AAAA    ::1
master  IN      A       192.168.1.200
slave   IN      A       192.168.1.201
client  IN      A       192.168.1.100
```

2. Reverse Zone file

Copy the existing zone file template **/etc/bind/db.127** to create a new reverse zone file.

```
cp /etc/bind/db.127 /etc/bind/rev.unixmen.local
```

Now edit the file **/etc/bind/rev.unixmen.local**,

```
nano /etc/bind/rev.unixmen.local
```

Change the contents as shown below and replace the domain name and ip address with your own.

```
;
; BIND reverse data file for rev.unixmen.local
;
$TTL    604800
@       IN      SOA     master.unixmen.local. root.unixmen.local. (
3         ; Serial
604800         ; Refresh
86400         ; Retry
2419200         ; Expire
604800 )       ; Negative Cache TTL
;
@       IN      NS      master.
@       IN      NS      slave.
@       IN      A       192.168.1.200
@       IN      A       192.168.1.201
@       IN      A       192.168.1.100
200     IN      PTR     master.unixmen.local.
201     IN      PTR     slave.unixmen.local.
```

```
100     IN      PTR     client.unixmen.local.
```

Save and close the file.

As you see in the above configuration, i increased the serial number for reverse zone file. For each change you should increase the reverse zone serial number as well.

Now restart bind9 service.

```
service bind9 restart
```

Test DNS Configuration and Zone Files

You can check the DNS configuration and zone files configuration for any syntax errors.

Check DNS configuration file using command:

```
named-checkconf /etc/bind/named.conf.local
```

If it returns nothing, your configuration file doesn’t have any syntax errors.

Check Forward Zone:

```
named-checkzone unixmen.local /etc/bind/for.unixmen.local
```

Sample Output:

```
zone unixmen.local/IN: loaded serial 2
OK
```

Check Reverse Zone:

```
named-checkzone unixmen.local /etc/bind/rev.unixmen.local
```

Sample output:

```
zone unixmen.local/IN: loaded serial 3
```

```
OK
```

Adjust iptables to allow DNS default port 53

First make sure that BIND9 is running on default port 53.

```
netstat -tulpn | grep :53
```

Sample output:

```
tcp        0      0 192.168.1.200:53        0.0.0.0:*               LISTEN      4893/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      4893/named
tcp6       0      0 :::53                   :::*                    LISTEN      4893/named
udp        0      0 192.168.1.200:53        0.0.0.0:*                           4893/named
udp        0      0 127.0.0.1:53            0.0.0.0:*                           4893/named
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           2582/avahi-daemon:
udp6       0      0 :::53                   :::*                                4893/named
udp6       0      0 :::5353                 :::*                                2582/avahi-daemon:
```

Now let us open port 53 through iptables.

First Install **iptables-persistent** package using command:

```
apt-get install iptables-persistent
```

Open up the file **/etc/iptables/rules.v4** using any editor,

```
nano /etc/iptables/rules.v4,
```

Add the rule to open port 53.

```
# Generated by iptables-save v1.4.14 on Tue Nov  5 13:20:11 2013
*filter
:INPUT ACCEPT [468:43718]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [374:41531]
```

```
:fail2ban-ssh - [0:0]
-A INPUT -p tcp -m multiport --dports 22 -j fail2ban-ssh
-A INPUT -p tcp -m multiport --dports 22 -j fail2ban-ssh
-A INPUT -p tcp -m state --state NEW --dport 53 -j ACCEPT
-A fail2ban-ssh -j RETURN
-A fail2ban-ssh -j RETURN
COMMIT
# Completed on Tue Nov  5 13:20:11 2013
```

Start or restart iptables-persistent service.

```
service iptables-persistent restart
```

Now check port 53 is open or not using command:

```
iptables -L -n
```

Sample output:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
fail2ban-ssh  tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 22
fail2ban-ssh  tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 22
fail2ban-ssh  tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:53

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain fail2ban-ssh (3 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
```

```
RETURN     all  --  0.0.0.0/0           0.0.0.0/0
RETURN     all  --  0.0.0.0/0           0.0.0.0/0
```

As you see in the above command, port 53 is open. Now your local clients will be able to resolve hostnames.

Test Master DNS Server

Edit file **/etc/resolv.conf**,

```
nano /etc/resolv.conf
```

And add your Master DNS server details,

```
search unixmen.local
nameserver 192.168.1.200
```

Reboot the system or restart the networking service.

```
service networking restart
```

Now let us check Master DNS server is working or not using the following commands:

Method 1:

```
dig master.unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> master.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51823
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;master.unixmen.local.        IN    A
```

```
;; ANSWER SECTION:
master.unixmen.local.    604800    IN    A    192.168.1.200

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    master.unixmen.local.
unixmen.local.        604800    IN    NS    slave.unixmen.local.

;; ADDITIONAL SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201

;; Query time: 14 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  5 19:31:18 2013
;; MSG SIZE  rcvd: 104
```

Method 2:

```
dig -x master.unixmen.local
```

Sample Output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -x master.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 48100
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;local.unixmen.master.in-addr.arpa. IN    PTR

;; Query time: 31 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  5 19:30:37 2013
;; MSG SIZE  rcvd: 51
```

Method 3:

```
dig unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65339
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;unixmen.local.            IN    A

;; ANSWER SECTION:
unixmen.local.        604800    IN    A    192.168.1.100
unixmen.local.        604800    IN    A    192.168.1.200
unixmen.local.        604800    IN    A    192.168.1.201

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    master.unixmen.local.
unixmen.local.        604800    IN    NS    slave.unixmen.local.

;; ADDITIONAL SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201
master.unixmen.local.    604800    IN    A    192.168.1.200

;; Query time: 11 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Tue Nov  5 13:31:22 2013
;; MSG SIZE  rcvd: 152
```

Method 4:

```
dig -x unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -x unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 3681
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;local.unixmen.in-addr.arpa.    IN    PTR

;; AUTHORITY SECTION:
in-addr.arpa.        3600    IN    SOA    b.in-addr-servers.arpa. nstld.iana.org. 2011029787 1800 900 604800 3600

;; Query time: 1590 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Tue Nov  5 13:31:36 2013
;; MSG SIZE  rcvd: 112
```

Method 5:

```
nslookup unixmen.local
```

Sample output:

```
Server:        192.168.1.200
Address:    192.168.1.200#53

Name:    unixmen.local
Address: 192.168.1.100
Name:    unixmen.local
Address: 192.168.1.200
```

```
Name:    unixmen.local
Address: 192.168.1.201
```

Method 6:

```
host unixmen.local
```

Sample output:

```
unixmen.local has address 192.168.1.200
unixmen.local has address 192.168.1.201
unixmen.local has address 192.168.1.100
unixmen.local has IPv6 address ::1
```

Method 7:

```
host master.unixmen.local
```

Sample Output:

```
master.unixmen.local has address 192.168.1.200
```

That’s it. Primary DNS server is ready and working.

2. Setup Secondary DNS Server

Install bind9 package with following command:

```
apt-get install bind9 bind9utils bind9-doc dnsutils
```

Configure DNS Server

Before proceeding to configure Secondary DNS server, first you should allow the zone transfer from your Primary DNS server.

To do so, go to Primary DNS server, edit the DNS main configuration file **/etc/bind/named.conf.local**,

```
nano /etc/bind/named.conf.local
```

Add the lines **allow-transfer { 192.168.1.201; };** and **also-notify { 192.168.1.201; };** in the zone section.

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone    "unixmen.local" {
        type master;
        file    "/etc/bind/for.unixmen.local";
        allow-transfer  { 192.168.1.201; };
        also-notify { 192.168.1.201; };
};

zone    "1.168.192.in-addr.arpa"        {
        type master;
        notify no;
        file    "/etc/bind/rev.unixmen.local";
        allow-transfer  { 192.168.1.201; };
        also-notify { 192.168.1.201; };
};
```

Here **192.168.1.201** is Secondary DNS server IP address. **allow-transfer { 192.168.1.201; };** line will transfer the zone files automatically from master dns to secondary dns server. **also-notify { 192.168.1.201; };** line will notify the secondary dns server if any changes in primary dns server zone files. Save and close the file.

Now go to Secondary DNS server, edit file **/etc/bind/named.conf.local**,

```
nano /etc/bind/named.conf.local
```

Add the lines as shown below.

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone    "unixmen.local" {
        type slave;
        file    "/var/cache/bind/for.unixmen.local";
        masters  { 192.168.1.200; };
};

zone    "1.168.192.in-addr.arpa"        {
        type slave;
        file    "/var/cache/bind/rev.unixmen.local";
        masters  { 192.168.1.200; };
};
```

Here **192.168.1.200** is my Primary DNS server IP address. Save and close the file.

Adjust iptables to allow DNS default port 53

First make sure that BIND9 is running on default port 53.

```
netstat -tulpn | grep :53
```

Sample output:

```
tcp        0      0 192.168.1.200:53        0.0.0.0:*               LISTEN      4893/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      4893/named
tcp6       0      0 :::53                   :::*                    LISTEN      4893/named
udp        0      0 192.168.1.200:53        0.0.0.0:*                           4893/named
```

```
udp        0      0 127.0.0.1:53            0.0.0.0:*                   4893/named
udp        0      0 0.0.0.0:5353            0.0.0.0:*                   2582/avahi-daemon:
udp6       0      0 :::53                   :::*                        4893/named
udp6       0      0 :::5353                 :::*                        2582/avahi-daemon:
```

Check the port 53 is open using command:

```
iptables -L -n
```

If port 53 is not open, do the following steps.

First Install **iptables-persistent** package using command:

```
apt-get install iptables-persistent
```

Open up the file **/etc/iptables/rules.v4** using any editor,

```
nano /etc/iptables/rules.v4,
```

Add the rule to open port 53.

```
# Generated by iptables-save v1.4.14 on Tue Nov  5 14:23:49 2013
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

-A INPUT -p tcp -m state --state NEW --dport 53 -j ACCEPT

COMMIT
# Completed on Tue Nov  5 14:23:49 2013
```

Start or restart iptables-persistent service.

```
service iptables-persistent restart
```

Now check port 53 is open or not using command:

```
iptables -L -n
```

Sample output:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:53

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

As you see in the above command, port 53 is open. Now your local clients will be able to resolve hostnames.

Now restart the bind9 service with command:

```
service bind9 restart
```

Now the zone files from master server will be replicated to secondary server automatically. The zones files will be created in **/var/cache/bind** folder of secondary server.

```
ls /var/cache/bind/
```

Sample output:

```
for.unixmen.local  managed-keys.bind  managed-keys.bind.jnl  rev.unixmen.local
```

Check Forward zone file:

```
cat /var/cache/bind/for.unixmen.local
```

Sample output:

```
$ORIGIN .
$TTL 604800    ; 1 week
unixmen.local        IN SOA    master.unixmen.local. root.unixmen.local. (
2          ; serial
604800     ; refresh (1 week)
86400      ; retry (1 day)
2419200    ; expire (4 weeks)
604800     ; minimum (1 week)
)
NS    slave.unixmen.local.
NS    master.unixmen.local.
A    192.168.1.100
A    192.168.1.200
A    192.168.1.201
AAAA    ::1
$ORIGIN unixmen.local.
client            A    192.168.1.100
master            A    192.168.1.200
slave            A    192.168.1.201
```

Check Reverse Zone file:

```
cat /var/cache/bind/rev.unixmen.local
```

Sample output:

```
$ORIGIN .
$TTL 604800    ; 1 week
1.168.192.in-addr.arpa    IN SOA    master.unixmen.local. root.unixmen.local. (
3          ; serial
604800     ; refresh (1 week)
86400      ; retry (1 day)
2419200    ; expire (4 weeks)
604800     ; minimum (1 week)
```

```
)
NS    slave.
NS    master.
$ORIGIN 1.168.192.in-addr.arpa.
100            PTR    client.unixmen.local.
200            PTR    master.unixmen.local.
201            PTR    slave.unixmen.local.
```

Test Secondary DNS Server

Edit file **/etc/resolv.conf**,

```
nano /etc/resolv.conf
```

And add your Master DNS server details,

```
search unixmen.local
nameserver 192.168.1.200
nameserver 192.168.1.201
```

Reboot the system or restart the networking service.

```
service networking restart
```

Now let us check Master DNS server is working or not using the following commands:

Method 1:

```
dig slave.unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> slave.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53146
```

```
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;slave.unixmen.local.        IN    A

;; ANSWER SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    slave.unixmen.local.
unixmen.local.        604800    IN    NS    master.unixmen.local.

;; ADDITIONAL SECTION:
master.unixmen.local.    604800    IN    A    192.168.1.200

;; Query time: 11 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  4 20:04:35 2013
;; MSG SIZE  rcvd: 104
```

Method 2:

```
dig -x slave.unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -x slave.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 8652
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;local.unixmen.slave.in-addr.arpa. IN    PTR
```

```
;; Query time: 4010 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  4 20:05:34 2013
;; MSG SIZE  rcvd: 50
```

Method 3:

```
dig unixmen.local
```

Sample Output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9642
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;unixmen.local.            IN    A

;; ANSWER SECTION:
unixmen.local.        604800    IN    A    192.168.1.200
unixmen.local.        604800    IN    A    192.168.1.201
unixmen.local.        604800    IN    A    192.168.1.100

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    slave.unixmen.local.
unixmen.local.        604800    IN    NS    master.unixmen.local.

;; ADDITIONAL SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201
master.unixmen.local.    604800    IN    A    192.168.1.200

;; Query time: 11 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
```

```
;; WHEN: Mon Nov  4 20:03:08 2013
;; MSG SIZE  rcvd: 152
```

Method 4:

```
dig -x unixmen.local
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -x unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 34729
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;local.unixmen.in-addr.arpa.    IN    PTR

;; Query time: 4006 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  4 20:03:58 2013
;; MSG SIZE  rcvd: 44
```

Method 5:

```
nslookup unixmen.local
```

Sample output:

```
Server:        192.168.1.200
Address:    192.168.1.200#53

Name:    unixmen.local
Address: 192.168.1.201
Name:    unixmen.local
```

```
Address: 192.168.1.100
Name:    unixmen.local
Address: 192.168.1.200
```

Method 6:

```
host unixmen.local
```

Sample output:

```
unixmen.local has address 192.168.1.100
unixmen.local has address 192.168.1.200
unixmen.local has address 192.168.1.201
unixmen.local has IPv6 address ::1
```

Method 7:

```
host slave.unixmen.local
```

Sample output:

```
slave.unixmen.local has address 192.168.1.201
```

Method 8:

```
host master.unixmen.local
```

Sample output:

```
master.unixmen.local has address 192.168.1.200
```

That’s it. Secondary DNS server is ready and up now.

Client Side Configuration

In your client system, edit file **/etc/resolv.conf**,

```
sudo nano /etc/resolv.conf
```

Comment out all existing DNS servers and add the your Primary and Secondary DNS servers IP addresses.

```
search unixmen.local
nameserver 192.168.1.200
nameserver 192.168.1.201
```

Test DNS Server

Run the following command to test Primary DNS server.

```
dig master.unixmen.local
```

Sample output:

```
; <<>> DiG 9.9.2-P1 <<>> master.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47844
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;master.unixmen.local.        IN    A

;; ANSWER SECTION:
master.unixmen.local.    604800    IN    A    192.168.1.200

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    master.unixmen.local.
unixmen.local.        604800    IN    NS    slave.unixmen.local.
```

```
;; ADDITIONAL SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201

;; Query time: 5 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  4 22:59:52 2013
;; MSG SIZE  rcvd: 115
```

Check Secondary server using command:

```
dig slave.unixmen.local
```

Sample output:

```
; <<>> DiG 9.9.2-P1 <<>> slave.unixmen.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16506
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;slave.unixmen.local.        IN    A

;; ANSWER SECTION:
slave.unixmen.local.    604800    IN    A    192.168.1.201

;; AUTHORITY SECTION:
unixmen.local.        604800    IN    NS    slave.unixmen.local.
unixmen.local.        604800    IN    NS    master.unixmen.local.

;; ADDITIONAL SECTION:
master.unixmen.local.    604800    IN    A    192.168.1.200
```

```
;; Query time: 3 msec
;; SERVER: 192.168.1.200#53(192.168.1.200)
;; WHEN: Mon Nov  4 23:00:25 2013
;; MSG SIZE  rcvd: 115
```

That’s it. Now you’ve successfully installed Primary and Secondary DNS servers.

Cheers!

SK

A Linux enthusiast, FOSS Supporter & Linux Consultant from Tamilnadu, India.

Setting Up DNS Server On CentOS 7

Comments for this thread are now closed.

3 Comments unixmen Login

Recommend 1 Share Sort by Newest

SAB • 2 years ago
Inside resolv.conf I prefer
nameserver 127.0.0.1

then you won't depend on the IP address
• Share ›

Ertan ERBEK • 2 years ago
You should use notify option other ways when you update first zone file but nut chnage soa record than second server can't understand and not update salve file so you can get two diffrent message from your server.
• Share ›

SK Ertan ERBEK • 2 years ago
Yes you're right. I forgot to mention it. I updated the article now. Thanks.
• Share ›

Subscribe Add Disqus to your site Privacy