CSIS 3230 Computer Networking Principles, Spring 2012

Lab 7 – Domain Name System (DNS)

By Michael Olan, Richard Stockton College (last update: March 2012)

Purpose

At this point, all hosts should be communicating with other hosts both inside their LAN and on other LANs the lab. However, the only way to reach any of them is by using IP addresses. The Domain Name System (DNS) allows access by name, which is much easier for a human to remember. Once a DNS server is running, it can easily be attached to the D017 network's DNS hierarchy or in real life, to the Internet.

In this exercise you will configure your local network to use DNS and to attach your DNS to the lab's "global" name service. DNS is one of the most complex services in networking but still uses plain text configuration files. There are many interrelated files, all with arcane and precise syntax requirements. You will be editing several configuration files on your server computer and testing your DNS configuration on both the server and client computers. There are plenty of places to make mistakes, so be careful.

There can be some confusion over what the DNS server is called. The actual DNS server program is called `named` (pronounced "name-dee"). The entire package of server, resolver library, and testing tools is called BIND (Berkeley Internet Name Domain).

1. Checking DHCP and router configuration

There are two essential requirements for the previous lab's routing configuration to work.

First, the DHCP server must be properly configured. Check and double check the `dhcpd.conf` file to be sure the `subnet`, `subnet-mask`, `routers`, and address `range` entries are correct for your subnet addresses. In particular, the `routers` entry should be your router's LAN side IP address. You can check your clients for this by running the command

```
route –n
```

There should be at least two entries in the routing table for the `eth0` interface - one for the local network and a default entry that shows your router's LAN address as the gateway.

The router must be properly configured according to the instructions in Labs 5 and 6.

2. Initial Testing

First, let’s see how DNS behaves when it’s not yet set up. Try the following commands on your server computer and note the results.

```
ping -c 5 172.30.100.50
ping -c 5 www.backbone.lab
```

Repeat these commands on at least one of your client computers and note the results. Take note of what works and what does not. Look at the file `/etc/resolv.conf` on these computers for clues that relate to the problems. For the time being, don't make any changes to this file.

The `host` program queries DNS servers to find an IP address to domain name mapping. Application programs that use names typically need to find a corresponding IP address. However, some programs might give the DNS server an IP address and want to get a name back. In our configurations, we will provide for both types of lookups. Run these commands on your server.

```
host www.example.org
host www.backbone.lab
host 172.30.100.50
```

Try these commands on your client computers and note the results.

```
host www.example.org
host x.y.z.w (your server's IP address)
```

Take note of what works and what does not.

3. Initial DNS Configuration

Pick a name for your server computer. This is the name everyone else in the lab will be using to access your server. The usual name is `ns1`. Use the `hostname` command to assign this name, but the current version of Ubuntu requires a couple other changes first or the all important `sudo` will break:

1) Edit the file `/etc/hostname` and change the name `ubuntu11` stored there to `ns1` (this sets the name on boot up)

2) Edit the file `/etc/hosts` and change the name `ubuntu11` stored there to `ns1` (this is used by `sudo` and must match the value in `hostname`)

3) Run the command (this temporarily changes the name):
```
sudo hostname ns1
```

Because we are running our own independent network, we can pick any domain names that we want. You would need to conform to the existing hierarchical naming scheme if your computers were to be accessible through the Internet. This exercise walks you through a basic DNS server setup that uses the domain name `example.org`. Once that is working, you will erase your work, and configure DNS again using your own domain name.

DNS configuration files are stored in the `/etc/bind/` directory. Note that there are several sample configuration files included here.

In the following steps, you will create a new domain (master zone), add your server's name and IP address to that domain and finally, add an alias for the web server. In the steps that follow, use the domain name `example.org`.

Edit the file `named.conf.local` as described below.

a) Add a master zone with name `example.org`:

```
zone "example.org" {
   type master;
   file "/etc/bind/zones/db.example.org";
};
```

b) The preceding step specifies the location of the configuration file for the zone. Now we need to create the file.

To have a convenient place for the zone files, make a directory called `/etc/bind/zones`. Then make a copy of `db.empty` in the new `zones` directory. Name the copy `db.example.org`. to match the name in the master zone definition.

c) Edit the `db.example.org` file so it looks similar to this, but use the IP address and name of your server. Just the lines in bold need to be modified, as shown on the next page:

```
$TTL 86400
@     IN   SOA  ns1.example.org. root.example.org. (
                    1        ; Serial
               604800        ; Refresh
                86400        ; Retry
              2419200        ; Expire
                86400 ) ; Negative Cache TTL
;
example.org.    IN   NS      ns1.example.org.
ns1             IN   A       192.168.x.y
www             IN   CNAME   ns1
```

(Be careful when editing, every dot, etc. needs to be right.)

Note that item names ending with a dot are used as is, while those that do not will have the zone name appended to them.

Because it's easy to make typos in these configuration files, there are some commands to do simple checks for blatant errors. Run the following and correct any errors.

```
named-checkconf
```

`named-checkzone` *zonename zonefilename* (use values from `named.conf.local`)

d) One more file needs to be modified: `/etc/resolv.conf`. Comment out any lines that are there, and add the following so that your DNS server will refer to itself for all DNS queries

```
nameserver 127.0.0.1
```

There may already be a DNS server running. Check with the command:

```
ps aux | grep named
```

If it is running, there will be an output line printed that indicates this (in addition to the one that shows the `grep` command)

Put the new configuration into effect with:

```
service bind9 start
```

(Note: Use `restart` if it's already running)

4. Testing your DNS server

Check that your system is configured with the correct DNS server address by doing the following.

Test your zones with these commands on the server:

```
dig ns1.example.org
dig www.example.org
```

These should resolve to your DNS server's IP address.

Make sure all of the dig results show status: `NOERROR`. Anything else indicates a problem with the DNS configuration (status: NXDOMAIN is a type of error).

Try these:

```
ping –c 5 172.25.100.50
ping –c 5 www.backbone.lab
```

Which of these work?

Repeat the all of the preceding `dig` and `ping` commands on the clients. Note which ones work and which don't. If there are problems, see if you can fix some of them by making changes in `/etc/resolv.conf` on the client).

You should now have a working DNS server, but it wont be known by any other DNS servers. We will fix this in the next step.

5. Let's get real – sort of

Using the domain name `example.org` gets an initial DNS configuration working, but it is totally useless in practice, especially since all the networks in the lab are using the same name. In general your domain will be part of a larger domain, like `.com` or `.edu`. For the rest of the semester, the top level domain will be `.lab`.

You will need to let your instructor know the actual name you use for your new subdomain. You can also select your own host name for your server.

Next wipe out your practice work and then create it again with your new domain name. Refer to the instructions in step 3 and make appropriate changes to the master zone by replacing `example.org` with your new domain name. Make sure the file name in `named.conf.local` match the actual names of your zone files.

Before repeating the tests in step 4, configure your DHCP server to give the IP address of your DNS server to clients. Restart the DHCP server and the network interfaces on the clients to have this change take effect.

Check the client’s `/etc/resolv.conf` after finishing these changes.

Try these commands on your client computers and note the results

```
host www.yourdomain.lab
host ns1.yourdomain.lab
host your_server's_IP_address
```

None of these should return an error. If any do, your DNS server isn't quite right.

Try the tests in step 4 again, but with your new domain name in place of `example.org`. Make note of any that fail.

The following steps should fix any remaining problems.

6. Connect your DNS server to the backbone (upstream)

Your local DNS should now be working well, but if any client machine on your network tries to find `www.backbone`.lab it fails. This is because your DNS server needs to ask a TLD server but it can't find one. Configure it to forward queries that are outside your own domain to the main DNS server, by editing the `/etc/bind/named.conf.options` file.

Remove the comment markers from the example forwarders entry and put `172.30.100.50` in the list.

Your LAN should be able to access names in the `backbone.lab` domain now.

Occasionally configuration errors don't show up when restarting the DNS server. The check commands used earlier just check syntax. Correctly formatted configuration files can still have logic errors. You can check `/var/log/messages` for run-time error messages.If problems exist, try running **System → Administration → Log File Viewer** and check the bottom of `syslog` for errors messages.

7. Connect your DNS server to the backbone (downstream)

While your clients may be able to access names in the `backbone.lab` domain, they can't get to any computers on other subdomains, like say `www.footbone.lab`. This is because the lab's TLD name server doesn't know how to get to these individual networks. To fix this, everyone needs to register their domain names. This entails telling the proper authority your domain name and the name/IP address of your DNS server, along with paying a nominal fee. In the lab, this authority is ICANT, with the instructor as the local agent. Fees are currently being waived. The root authority will update its list of mappings and once that is done, your server will be accessible to everyone else. Once the instructor gives the ok, check this by having another group ping the server in your domain.

8. Going in reverse

Now we need to provide configuration for mapping addresses to names. This requires adding a reverse master zone. Add the following to `named.conf.local` (replacing *x* as appropriate):

```
zone "X0.168.192.in-addr.arpa" {
   type master;
   file "/etc/bind/zones/db.192";
};
```

Copy the file `db.127` into the zones directory and call it `db.192`.

Edit `db.192`, replacing *y* with the *host number* of your server and using your own domain name as shown.

```
$TTL 86400
@     IN   SOA  ns1.yourdomain.lab. root.yourdomain.lab. (
                      1        ; Serial
                 604800        ; Refresh
                  86400        ; Retry
                2419200        ; Expire
                  86400 ) ; Negative Cache TTL
;
yourdomain.lab.  IN   NS   ns1.yourdomain.lab.
ns1              IN   A    192.168.x.y
www       IN   CNAME   ns1
```

Restart the DNS server and watch for errors. Test the reverse lookup with

```
dig -x your_server's_IP_address
```

This should return `ns1.`*yourdomain*`.lab` as the hostname associated with the IP address.

Note: The Webmin GUI tool can also be used for configuring the BIND DNS server—and just about everything else!

It is accessed with a web browser at the url: `https://localhost:10000`.

The latest versions of Webmin and Ubuntu don’t share user names and passwords. In order to log in to Webmin you need to set this up as follows:

```
    cd /usr/share/webmin
    sudo perl changepass.pl /etc/webmin d17user d17u$er
```