DNS Basics

1 Overview

This exercise introduces some basic functions and protocol elements of the Internet's Domain Name Service (DNS). The student will interact with an example enterprise having a local DNS server and several computers.

This exercise, (and manual), is not intended to replace instruction or independent reading on DNS.

Broadly, a DNS provides a mapping between IP addresses and computer names. DNS allows us to use names such as "google.com" instead of remembering the IP address for google. Computers generate *queries* such as "What is the IP address for google.com?" and send them to the DNS. The DNS generates a *response* to the query, providing the requested information. If the DNS does not itself manage the requested information, the DNS forwards the query to another DNS for resolution.

This exercise is intended to provide students with an environment with which they can observe traffic generated by DNS queries and responses.

This lab and its prerequisite knowlege provide background for other Labtainer networking exercises including a lab on local DNS poisoning attacks.

This lab exercises includes use of the Linux command line (shell), and Wireshark.

2 Lab Environment

This lab runs in the Labtainer framework, available at http://nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer dns
```

A link to this lab manual will be displayed.

3 Network Configuration

The lab includes networked components as illustrated in figure 1. Note that all of the IP addresses are local. The DNS is configured to provide authoritative naming for devices within the domain called *example.com*. In this topology, the DNS is only providing local naming and forwards any non-local requests to the gateway.

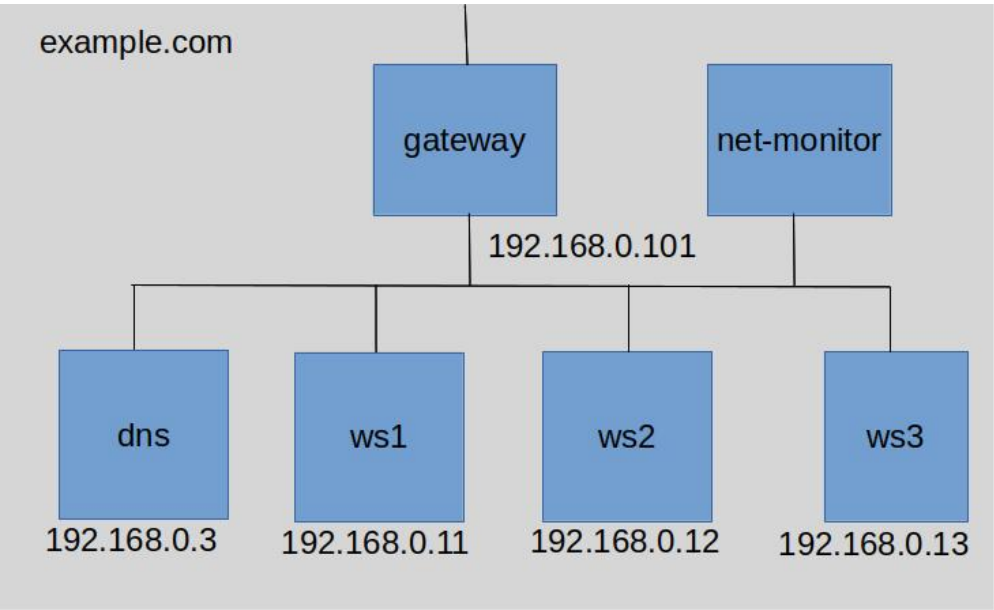

Figure 1: Network topology for the DNS lab

4 Lab Tasks

4.1 Explore

The DNS is configured to provide naming devices within `example.com`. Initially, it is configured to name `ws1` and `ws2`. And the `ws1` and `ws2` computers are configured to use the DNS as their name server. The `ws3` computer is not yet configured. Use the `ping` command see how you can use the names `ws1` and `ws2` in place of IP addresses. For example, on `ws1` type:

```
ping ws2 -c 2
```

Note that if you try to ping `ws3`, if fails. However you can ping `ws3` using its IP address.

You will update the DNS and `ws3` later in this exercise. But first, we will review some of the DNS protocol.

4.2 View DNS traffic

The lab environment includes a `new-monitor` computer that captures all traffic on the local network. It collects all network traffic into a file within its `/taps` directory. Go to the terminal for the `net-monitor` computer and list the content of that directory to see that it is a PCAP file:

```
ls -l /taps
```

You can start Wireshark and view that PCAP file as it is updated by using this command on the `network-monitor` component:

```
sharktap lan.pcap
```

After Wireshark comes up, scroll backward and find the intitial DNS traffic. Note that this conversation occurs between the local DNS and some external DNS, and is used by the local DNS for initialization. Scroll forward to the next DNS protocol conversation, which should be between the `ws1` component and the DNS.

Select the first packet of that exchange, i.e., the one from your `ws1` component and then view the details of the protocol as presented by Wireshark in the middle pane. Find the "queries" block and view the query. Note how the query is asking for the IP address of *ws2.example.com*, yet you only provided the name `ws2` to the ping command. So, where did the `example.com` name come from? Within this Linux system, that information is provided as the *search* directive within the `/etc/resolv.conf` file on the computer that is issuing the DNS query. Go to `ws1` and view that file:

```
cat /etc/resolv.conf
```

That search directive tells the computer to tack the name `example.com` onto names that do not end with a domain suffix such as *.com*, e.g., are just names like `ws2`. Note the `resolv.conf` file also contains a *nameserver* directive. This tells the computer the IP address of its DNS service.

Note how there is a 2nd query from `ws1` to the DNS. That is to retrieve and IPV6 address. Ignore that for this lab. The next packet should be a response from the DNS to ws1. View its protocol elements. Note how it repeats the query, but also has an *answers* section. View that to see the response from the DNS. The next packet is the DNS response to the IPV6 query, which can be ignored for this lab.

4.3 Missing names

The `ws3` computer is not yet defined in the DNS. Test this by trying to ping `ws3` from `ws1`:

```
ping ws3 -c 2
```

Find the reply from the DNS and note the respone flags indicates there is "no such name". Note that `ws1` then tries another query, this time asking for `ws3` instead of `ws3.example.com`. Since this query is not for a name within the `example.com` domain, the local DNS farms out the request to another DNS via the gateway (192.168.0.101). That DNS responds with "No such name".

4.4 Missing DNS

Go to `ws3` and try to ping `ws1`:

```
ping ws1 -c 2
```

Note the lack of traffic in Wireshark. The `ws3` computer has no defined DNS, so it does not know where to send a DNS query. Use `vi` or `nano` to edit the `/etc/resolv.conf` file so that it matches those in `ws1` and `ws2`.

Then try pinging `ws1` again. Now that `ws3` has the address of its `nameserver`, it can use names instead of IP addresses.

4.5 Add missing name

In this section, you will update the DNS so that it can provide the IP address of `ws3`. The DNS is provided by the *bind9* service on the `dns` computer. Information about this service and its configuration files can be found at `https://ubuntu.com/server/docs/service-domain-name-service-dns`. This section refers to the primary files used to configure the DNS.

The starting point is the `/etc/bind/named.conf` file[1]. View that file and note that it simply includes three other files. The second file, `named.conf.local` is where the local names are defined. View that file. As you see, this DNS manages names defined in a single local file, `example.conf`. View that file. At last, a file that does not just include other files! This file includes to *zone* sections. The first defines how the DNS will respond to queries of names within the `example.com` domain, or `zone`. The second entry defines a *reverse name lookup*, which is another function of the DNS via which a computer can query the name associated with an IP address, e.g., "What is the name of the device having IP address 192.168.0.1?".

View the file named in the first section, i.e., `/var/named/example.com.zone`. The first set of lines in that file reflect various configuration options for handing queries for the `example.com` domain, including things like timeout values. Toward the bottom of the file you will see entries that look like a mapping of computer names to IP addresses. There are entries for `ws1` and `ws2`, but not one for `ws3`. Add one for `ws3`.

Then direct your attention to the second section of the `example.conf` file, i.e., the reverse name looking information. Note the funny looking zone name: *0.168.192.in-addr.arpa*, and how the first part of the name includes the `example.com` domain address, but in reverse. This indicates that if the first three octets of an IP address match those three octets, then the remaining octet will name the computer. View the named file, i.e., `/var/named/192.168.0.0`. Observe how the last two entries identify the last octet in the IP addresses of `ws2` and `ws3`. Add an entry for `ws3`.

4.6 Apply and test DNS changes

After modifying the DNS configuration files, the DNS service must be restarted:

```
sudo systemctl restart bind9
```

Then try to ping `ws3` from `ws1`. You should be able to see the resulting DNS query and response in Wireshark.

5 Quiz

Go to the terminal on your Linux system that was used to start the lab and type:

```
quiz -l dns -q post
```

and answer the quiz questions to test your knowledge of what you have done in this lab.

6 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab
```

When you stop the lab, the system will display a path to the zipped lab results on your Linux system. Provide that file to your instructor, e.g., via the Sakai site.

This lab was developed for the Labtainer framework by the Naval Postgraduate School, Center for Cybersecurity and Cyber Operations under National Science Foundation Award No. 1932950.

[1] The word *named* derives from the fact that this service is sometimes referred to as the *name daemon*