KIT304 Server Administration and Security Assurance

Tutorial 12

Goal

In this tutorial you will learn how to configure private DNS servers to share the information about the websites and other services that you create. Why private DNS servers? Running a DNS server on the public internet is a risky endeavour. To be authoritative, you need to go through a registration process with the owner of the parent domain that you're providing lookups for. That's not something that many businesses can or want to do – instead, many use the zone manager services of an existing registry. It's simpler, less prone to failure, and usually means telephone support is close by should you get into trouble. That won't be the case if you're running your own DNS server. So, today, we'll still be running DNS servers, but they'll be for private, internal use only. You'll see how this works for both Windows Server, and for Unix.

> ➢ Setting up a DNS service is fundamentally different to manipulating the **`hosts`** file to map host and domain names to IP addresses. Hosts file entries are checked first, and override anything from DNS, but they're local to the machine they're created on. A private DNS server can play a similar role, but importantly, it can play that role for systems across a local area network. While private DNS servers aren't useful for providing public internet users with the authoritative details of a server you host, they're commonly used in private infrastructure networks and in development environments, since they provide the flexibility of "real-world" domain names without the need to open the private network to a public DNS service.

Introduction

Networked interfaces, such as those that provide Ethernet and Wi-Fi access, have a *physical* network address that identifies each interface globally – this is known as the MAC address. MAC addresses are not routed and are not arranged hierarchically, and so they are not suitable as a way to pass network traffic globally. For that, we use IP addresses, also known as *logical addresses*.

IP addresses are mapped to MAC addresses to enable computers to be found globally, and are organised in a hierarchy that simplifies the process of routing. IP addresses, like MAC addresses, are not suitable for humans to use to navigate to network services. Instead, they are mapped to d*omain*

names that are much more readily understood and remembered. The protocol that enables this mapping to occur is the Domain Name Service (DNS).

When you enter a domain name into the address bar of a web browser, the browser issues a DNS lookup request to find its matching IP address. Once the IP address is known, the request can be made to the server at that address for its content. DNS information is mirrored across the globe on thousands of servers. In the tutorial today you will set up several websites, and then add their details to a private (internal) DNS server. You will then be able to add client computers to the network, and configure them to use your new DNS server. With that information, your clients will then be able to navigate to destination sites by domain name.

Activity – Setting up a DNS Server on Windows

1. Launch a **CentOS** VM and give it an IP address in the **`192.168.1.x`** subnet.

2. Create a new website with a domain name of your choice. Here follows a brief summary of the commands and configuration file entries you'll use, but you'll have to remember where to use them. For detailed steps refer back to tutorial 9:

```
firewall-cmd --permanent --zone=public --add-service=http
firewall-cmd --permanent --zone=public --add-service=https
firewall-cmd --reload

NameVirtualHost *:80
IncludeOptional sites-enabled/*.conf

<VirtualHost localhost:80>
     DocumentRoot /var/www/html
</VirtualHost>
<VirtualHost *:80>
     ServerName www.yourDomain
     DocumentRoot /var/www/yourWebDirectory/html
     ServerAlias yourDomain
     ErrorLog /var/www/yourWebDirectory/error_log
     CustomLog /var/www/yourWebDirectory/access_log combined
</VirtualHost>

cd /var/www/yourWebDirectory
touch error_log
touch access_log
chcon --reference /var/log/httpd/error_log error_log
chcon --reference /var/log/httpd/access_log access_log
```

   Other commands you may need to complete the task: **`service`**, **`mkdir`**, **`chown`**, **`chmod`**.

3. Place an **`index.html`** file in your website's **html** folder, and test that it works by adding the domain to the local **`/etc/hosts`** file. Confirm that it works and is accessible from a browser running on the CentOS VM.

4. Launch the **Windows Server 2016** virtual machine, and give it another IP address in the **`192.168.1.x`** subnet, and ensure its DNS server setting points to itself.

5. Your next task is to create a new website on the Windows Server 2016, as you did back on Tutorial 10. It should be located in a folder inside the **`inetpub`** directory of the C: drive. Create an **`index.html`** file for this site with appropriate content so that you can distinguish between it and the CentOS website. (IIS is already set up – you don't need to install any additional components such as ASP for this tutorial).

   Don't forget to add **`IIS_IUSRS`** to the permissions of the website's directory. You should create the web site itself through the GUI using the **Internet Information Services Manager**.

6. This time *don't* edit the Windows local hosts file to create an entry for your new web site. Instead, you are going to configure the Windows Server 2016 to be a **DNS** or **Domain Name Server**. To achieve this, launch the **DNS Manager** (called **DNS** in the list of tools) from the **Server Manager**.

7. In the tree hierarchy on the left-hand pane, open the **CORPDC1** to expose the **Forward Lookup Zones** item. Right click on this and select **New Zone**. Proceed through the wizard selecting the default options and then when you reach the **Zone Name** section, enter the domain you first created on CentOS. Next select the **Allow both nonsecure and secure dynamic updates** option, click **Next**, and then click **Finish**.

8. Expand the **Forward Lookup Zones** branch in the left pane, and right-click on the new zone you just created, and select **New Host**. Leave the name section blank and enter the IP address of the CentOS server in the IP Address field. Click **Add Host**, click **OK**, and then click **Done**.

9. Repeat steps 7 and 8 to add your Windows 2016 web site to the DNS server. Once completed attempt to open both websites in **Internet Explorer** on **Windows 2016** using their domain names.

   If it does not work, attempt to ping the domain name from PowerShell. If you can't ping the domain you may have configured the domain incorrectly. If that is the case you may need to delete the domain (right-click on it in the DNS tree) and then re-add it. Between those two steps you may want to type **`ipconfig /flushdns`** into PowerShell to flush any cached DNS entries that you need to update. If you can ping the CentOS site by its domain name, but you can't load it in the web browser, check that you configured the firewall correctly on CentOS.

10. Finally, now that both websites are loading on Windows Server 2016, launch the **Windows 7** VM, and give it an IP address in the **`192.168.1.x`** subnet – but this time set the DNS server in the network settings to be that of the Windows 2016 server. Next open FireFox and attempt to view each website by domain name. Both should load, and you haven't had to resort to "hacks" like hard-coding IP addresses to domain names in the Windows **`hosts`** file.

Activity – Setting up a DNS Server on CentOS

Next you're going to set up a private DNS server on CentOS. This is a little more complex than doing the same thing in Windows Server, since you'll be using the command line for the entire process.

To start setting up a DNS server on a Linux system, you need to have the **BIND** service and its associated tools installed. BIND is an acronym for Berkeley Internet Name Domain, and it was originally developed at the University of California, Berkely, in the early 1980s. We've already installed BIND and the BIND tools on CentOS for you, but they're not configured.

BIND uses multiple configuration files that are included from the main configuration file **/etc/named.conf**. This is the same approach you've already seen with the Apache web service, which has a main configuration file (**/etc/httpd/conf/httpd.conf**) that in turn includes other configuration files, including those that you place in the **sites-enabled** directory to define your named virtual hosts.

For the BIND service, some of the configuration files begin with the string **named** (short for **name d**aemon) and **named** is also the name of the process that provides the BIND service. If that sounds confusing, remember that Apache doesn't start a process called apache – it starts a process called **httpd**. In the same way, BIND doesn't start a process called bind, it starts a process called **`named`**.

11. To start configuring BIND, open a terminal window on CentOS, and **become the root user**. Open the **`/etc/named.conf`** file in any editor you're comfortable with. There is already configuration information in this file, and you need to be careful not to modify it accidentally. Above the existing **`options`** block, you're going to create an ACL (access control list) that defines the list of clients you'll trust and are prepared to perform recursive DNS queries for. This can be your entire 192.168.1 subnet, in which case it should look as follows:

    ```
    acl "trusted" {
            192.168.1.0/24; localhost;
    };
    ```

    Don't forget the semicolons – BIND will complain and not start if any are missing.

    Next, just inside the **`options`** block, you'll see the **`listen-on`** directive. This specifies that the service must listen on port 53, but only on the localhost interface (127.0.0.1). You want it to also listen on the IP address you've set up for the CentOS system (so that other hosts can use the service too), so change it to match this:

```
listen-on port 53 { 127.0.0.1; CentOSIPAddress; };
```

Again, note that there's a semicolon *after* the IP address.

The next line under this specifies the IP addresses to listen on for IPv6 connections. Since you're not using IPv6, you can comment this line out by placing a **//** comment marker at the start of the line.

Further down in the file, locate the **allow-query** directive. This specifies which hosts you'll support queries for. This is predefined to be only localhost, but you need to change this to the name of the ACL you created earlier that defines the trusted hosts, like this:

```
allow-query { trusted; };
```

Now move to the bottom of the file You'll see some existing **include** directives. Add one more at the bottom of the file to include a new configuration file that you're going to use:

```
include "/etc/named/named.conf.local";
```

Save your changes to the file, and exit. In summary, the changes you've made to the main config file are to declare which hosts BIND will accept DNS queries from, and on which IP address it will listen for those connections.

12. Next, you're going to create the new file **named.conf.local** in **/etc/named** (this is the file that you've just referenced via the new **include** directive at the bottom of the main configuration file). This new local config file is going to define a **zone** for *forward lookups* on your domain. Note that this file won't include the DNS records themselves – they will be stored in yet other file that this definition refers to.

    Before you set up the zone file definition, you need to decide on the domain name that your hosts are going to reside in (each Zone in DNS represents a single domain). For example, it could be **example.com**, **kit304.com.au**, or anything else you wish to use. In the example below, we'll use **networks.com**, but you can replace this with your own domain.

    When you've selected a domain name, edit the file **/etc/named/named.conf.local** with your preferred editor, and enter the text below, substituting your domain name as appropriate:

```
zone "networks.com" {
    type master;
    file "/etc/named/zones/db.networks.com"; # forward zone file path
};
```

Save and close the file. Note the path to, and name of, the forward zone file in the content above. You'll be creating this file in the next step.

13. You're almost ready to create your forward zone file. According to your **named.conf.local** file, this will be in **/etc/named/zones**, but that directory doesn't yet exist. Create it with the following command:

```
mkdir /etc/named/zones
```

Now you're ready to create the forward zone file. Create this with your preferred editor, remembering that it needs to be stored (and named appropriately) as specified in the full path in the config file you just created (**/etc/named/zones/db.networks.com** in the example we're using, but that depends on the domain name you've chosen to use). Add the following text to the file, replacing the domain name *networks.com* with your own as appropriate, and adding some A records of your own at the end that will map to real IP addresses:

```
$TTL    604800
@       IN      SOA     ns.networks.com. admin.networks.com. (
                  1     ; serial number
             604800     ; refresh
              86400     ; retry
            2419200     ; expire
             604800 )   ; minimum time to live

; name server (NS) records
    IN      NS      ns.networks.com.

; name servers - A records
ns.networks.com.          3600      A      ipAddresssofCentOS

; A records – add your own additional hostnames/IP addresses here
@       3600   A   192.168.1.1      ; default IP for 'networks.com'
win7    3600   A   192.168.1.7      ; host IP
win10   3600   A   192.168.1.10     ; host IP
```

You should be familiar with all of these records, as they were covered in the week 6 lecture. If you're copying-and-pasting the content above from a PDF of this tutorial sheet, check everything carefully, as PDFs sometimes don't store text as plainly as you would need it. Save and close the file when you're done.

14. The **bind tools** come with a utility to check that your configuration files don't contain any errors. Run it by entering the following command:

```
named-checkconf
```

If any errors are reported, correct them and check again until you have a configuration with no errors.

Finally, before you enable your nameserver, give your CentOS host its own hostname on the domain. The way you do this is as follows:

```
hostnamectl set-hostname hostname.domainname
```

Thus, if you wanted to name the host **centos**, and your chosen domain name was **networks.com**, you'd enter the command:

```
hostnamectl set-hostname centos.networks.com
```

15. You're now ready to start the BIND service. Do that as follows:

```
service named start
```

It's still possible that errors will occur here. If any are reported, attempt to fix them and try to restart the service again. If you need assistance, be sure to ask your tutor.

Since CentOS is running a built-in firewall, you also need to configure it to allow incoming DNS queries. You can do that with the following commands:

```
firewall-cmd --permanent --zone=public --add-port=53/tcp
firewall-cmd --permanent --zone=public --add-port=53/udp
firewall-cmd --reload
```

16. With the BIND service now running, you can try some hostname lookups. You can do this with the **nslookup** command. For example, using the example zone file on the previous page, you should be able to enter this command:

```
nslookup win7
```

and you'd get the returned value **192.168.1.7** (or whatever value you've defined). This lookup works because:

- even though you haven't yet set a DNS server setting in your CentOS network settings, the **named** server is also listening on **localhost**, and that's what **nslookup** will default to if it doesn't have a specified DNS server to talk to
- since you've declared with the **hostnamectl** command that your CentOS system is on the same domain that the **named** server is authoritative for, you don't need to enter the full domain to lookup – it will assume **networks.com** (or whatever you've used)

Try looking up the IP address of all of the hosts you've added in your zone file.

17. Before progressing further, configure your CentOS server use its public IP address (rather than **localhost**) for DNS lookups. To do this, go to the same network configuration settings that you used to configure its IP address back at task 1, and set the DNS server address in the section below where the IP address is set.

18. Next, you're going to modify the zone file to add an entry for the CentOS system itself (assuming you haven't already done this when you set the zone up initially.

    Using your preferred editor, open the zone file you created at step 14, and add a new **A** record at the end that maps a hostname for the CentOS server to its IP address. For example, that might be **centos.*yourdomain***, or any other hostname you want to use on the domain you're using. While you're editing this file, you should increment the serial number near the top of the file. You should always increment or modify this value when editing the zone file so that other secondary nameservers know when the file has been updated and they need to transfer the updated version (we're not covering how to configure a secondary nameserver today).

    After saving your changes, you need to check the file syntax, and then (assuming the syntax is correct) tell the BIND service to reload the configuration information, as follows:

    ```
    named-checkconf
    service named reload
    ```

19. Now switch to your Windows 7 virtual machine, and change its DNS server setting so that it's pointing to the CentOS DNS server, rather than the Windows 2016 server. While you're there, ensure its IP address matches a value you put in your **named** zone file back at step 14.

    Now, you should be able to open the Windows 7 command line, and use either **nslookup** or **ping** to verify that the name server is reachable, and that you can use fully qualified domain names in ping commands (rather than IP addresses).

20. Back at task 2, you created a web site on the CentOS host, but you most likely chose a different domain name for that host than the one you've currently set for your DNS server.

    Edit the web site configuration file (in **/etc/httpd/sites-enabled**) and change the **server name** and **server alias** to match the fully qualified domain name for your CentOS server (e.g., **centos.networks.com)**. You *don't* need to change any other references in this file to match the new domain name, since those are pointing to files and directories on the local file system, and have no bearing on how Apache sees the server's domain name.

    After making this change, reload the web server (**service httpd reload**) and you should then be able to access this by host and domain name from your Windows 7 browser.

21. Start Windows 10, give it an IP address, and configure it to use the CentOS system as its DNS server. Verify that it too can load the CentOS web site by name – without you having to modify the Windows 10 **hosts** file.

22. Next, you're going to modify your DNS server configuration so that you can do *reverse lookups* – that is, to start with an IP address, and look up the registered domain name for that address (if one should exist). To do this you need to modify **/etc/named/named.conf.local** and add a new section at the bottom that defines a new zone for reverse lookups. This looks a little different to the forward zone definition that's already in the file – for example, for the 192.168.1.0/24 network it would look as follows:

```
zone "1.168.192.in-addr.arpa" {
    type master;
    file "/etc/named/zones/db.1.168.192";  # 192.168.1.0/24 subnet
};
```

Notice the green digits that represent the network (**192.168.1.x**) are listed in *reverse* order, and that the last octet is not included. That final octet represents hosts within the network, and those host octets are expressed as records in the zone file.

Add your reverse zone to the configuration file, and after saving and closing it, create and edit the new file that will store the reverse zone records **/etc/named/zones/db.1.168.192** and add content similar to the following but that represents your own registrations. Note that the top part of the file is identical to the forward zone file you created earlier. The major difference is at the bottom, where PTR records are now used instead of A records. Replace the hostnames and IP addresses there to suit your own current settings.

```
$TTL    604800
@       IN      SOA     ns.networks.com. admin.networks.com. (
                  1     ; serial number
             604800     ; refresh
              86400     ; retry
            2419200     ; expire
             604800 )   ; minimum time to live

; name server (NS) records
    IN      NS      ns.networks.com.

; PTR records for reverse lookups
1     IN      PTR     centos      ; 192.168.1.1
7     IN      PTR     win7        ; 192.168.1.7
10    IN      PTR     win10       ; 192.168.1.10
```

Note that the digit at the start of the line is the lowest octet of the actual IP address

Again, if you're copying-and-pasting the content above from a PDF of this tutorial sheet, check everything carefully before saving and exiting the editor.

23. After saving your changes to the reverse zone file, check the file syntax, and have the BIND service reload the configuration information from the updated files as follows:

```
named-checkconf
service named reload
```

24. Now try to do a reverse lookup. For example:

```
nslookup 192.168.1.1
```

if the IP address you've specified is actually listed in the reverse zone file, that system's domain name should be reported, like this:

```
[root@localhost]# nslookup 192.168.1.10
10.1.168.192.in-addr.arpa     name = win10.1.168.192.in-addr.arpa.
```

Not all services on the internet go to the trouble of registering reverse lookups for their IP addresses, but being able to lookup such information can be useful when fault finding and when trying to trace the owner of an IP address that might be the source of an attack.

WHOIS

The **whois** command line utility allows you to view specific information about a domain, including the IP addresses and/or host names of the domain's DNS servers, and any associated contact information, which often (though not always) includes an address and phone number. This information doesn't come from the DNS system directly, but rather from the registry through which the domain owner registered and paid for the domain.

whois is built into Linux and macOS operating systems and can be used by typing the command:

```
whois domain
```

On your lab computer, open the macOS **Terminal** (which should be in the dock at the bottom of the screen) and enter the above command, choosing a well-known domain name. If you are using a Windows computer (which does not support the **whois** command) there are a number of websites that allow you to access **whois** information, such as **http://who.is/.**
It is worth noting that there is often some effort expended in hiding who owns a domain, and there are companies that are offer registrant privacy as a for-fee service.

You won't be able to use **whois** to determine anything related to private DNS servers, since they don't need to be registered through a registry in the first place.

Conclusion

DNS facilitates the basic usability of the Internet, mapping IPs to domain names (and vice versa when configured to do so). For this to occur the information needs to be public, and cached globally across thousands of DNS servers. The internet is built on, and relies on the public domain name service for the majority of user-initiated sessions.

In this tutorial you've set up private Windows and Linux-based DNS servers that respond to DNS requests, translating domain names to IP addresses so that multiple hosts can be referenced by name on client systems.

Skills to Master

To successfully complete this tutorial, you must have mastered the following skills:

- creating forward DNS zones on Windows 2016 Server
- configuring and starting the BIND service on CentOS
- creating forward and reverse zones on the CentOS BIND service
- using **`whois`** to look up information about publicly registered domains

You could be required to demonstrate any of these skills in this module's practical exam.