# Overview

## Goals

The installation of a DNS server that is authoritative for a sub domain of hades.lab. This DNS server should be capable of answering any query of the hades.lab domain and its sub domains. It should also be able to answer any queries made of its sub domain by the name server for the entire domain.

Stage two is taking the working DNS server for the sub domain and making it DNSSEC aware. This lab will not show how integrate DNSSEC with other DNS servers as the lab's DNS server is not capable of speaking DNSSEC.

## Deliverables

* Lab Report Including the following
	+ All the completed information sections throughout the lab
	+ Description of the lab steps completed and the results
	+ Any errors you encountered and the steps you took to resolve them
	+ Any mistakes found or corrections made to the lab steps
	+ Answers to the questions posed at the end of this lab
	+ Text copies of the following in the report:
	+ */etc/bind/named.conf.local*
	+ Zone file (e.g. */var/cache/bind/db.****SUBZONE***)
	+ Text file containing the results of *'dig +dnssec –t soa* ***SUBZONE @DEBIAN\_IP****'*

## Information

Table

|  |  |  |  |
| --- | --- | --- | --- |
| ***Description*** | ***Formula*** | ***Used in Lab*** | ***Answer*** |
| Assigned Subnet Number: | *n* | - |  |
| Your Last Name: | *<last\_name>* | - |  |
| Debian VM IP: | 172.16.*n*.10 | DEBIAN\_IP |  |
| Raspbian Pi IP: | 172.16.n.20 | PI\_IP |  |
| CentOS VM IP: | 172.16.*n*.50 | CENTOS\_IP |  |
| DNS Forwarder: | - | 172.16.1.252 | 172.16.1.252 |
| Subzone: | *<last\_name>*.hades.lab | SUBZONE |  |

If your last name is Jones, your SUBZONE would be **jones.hades.lab**. Use the last name as it appears (no uppercase letters) on the IP subnet assignment page on Moodle. REMEMBER - as of this lab your DEBIAN\_IP needs to end in 10. The DNS glue records on the lab server have been set to point to 172.16.*n*.10. Any number other than 10 will not function properly.

# Stage 1: DNS Server Installation & Configuration

## Getting Started

1. Load Debian VM
2. Verify the VM’s IP Address matches the IP address you calculated in the Information section above
	1. Refer to Lab 1 for information on viewing and modifying your IP address

## Initial Installation and Configuration

1. Install the “bind9” package
	1. aptitude –R install bind9
2. Edit */etc/bind/named.conf.options*
	1. Debian splits the standard named.conf into three separate files
		1. *named.conf* – boiler plate file. Contains some standard configuration and references the two files listed below. This file almost never needs to be changed
		2. *named.conf.options* – contains site-specific configuration options to the name server
		3. *named.conf.local* – contains references to the local zones that this server controls
	2. Uncomment the ‘*forwarders*’ section and replace 0.0.0.0 with IP of the Lab’s DNS forwarder noted in the Information section above
	3. Modify the dnssec\_validatoin line to 0

 forwarders {

 172.16.1.252;

};

…

dnssec\_validation 0;

1. Restart DNS Server
	1. /etc/init.d/bind9 restart
2. Verify server is responding correctly
	1. dig mirror.hades.lab @**DEBIAN\_IP**
	2. Note: You should see a non-zero value for the "Answer" when you do a ***dig***. If you see a zero, there is an error. This is generally true for all digs you do.

## Creating a Sub Zone

1. Edit /etc/bind/named.conf.local
	1. Add a zone for your subzone you calculated in the Information section above

zone "**SUBZONE**" IN {

type master;

file "/var/cache/bind/db.**SUBZONE**";

allow-update { none; };

};

1. Create a zone file for your subzone
	1. e.g. /var/cache/bind/db.**SUBZONE**
	2. The file name must match the file name referenced in named.conf.local
	3. Use the IP addresses you calculated in the Information Section above
	4. $TTL and Negative Cache TTL are both set to 2 minutes so mistakes any mistakes can be corrected quickly. In a real site, you would want $TTL set to a minimum of one day and the Negative Cache TTL set to 15 minutes.
	5. Use standard convention for the Serial, e.g. yyyymmddnn.

$ORIGIN **SUBZONE**.

$TTL 2m

@ IN SOA ns.**SUBZONE. jwatso8.uncc.edu.** (

 **2011091201** ; Serial

 1w ; Refresh

 1d ; Retry

 28d ; Expire

 2m ) ; Negative Cache TTL

;

@ IN NS ns.**SUBZONE**.

@ IN A **DEBIAN\_IP**

ns.**SUBZONE**. IN A **DEBIAN\_IP**

debian IN A **DEBIAN\_IP**

**raspbian IN A PI-IP**

centos IN A **CENTOS\_IP**

test.**SUBZONE**. IN CNAME Debian

Figure . If you make any mistakes in this file, update the Serial before you reload it.

1. Verify the zone file you just wrote
	1. named-checkzone **SUBZONE** /var/cache/bind/db.**SUBZONE**
	2. Should return a statement saying everything is well. Correct any errors in your zone file before continuing.
2. Reload the DNS server
	1. rndc reload
3. Verify that the server is responding correctly
	1. dig debian.**SUBZONE** @**DEBIAN\_IP**
	2. dig debian.**SUBZONE** @172.16.1.252
	3. Both dig commands should return the same information

## Troubleshooting

* The system log may contain useful error messages
	+ /var/log/syslog
	+ ‘less’ is a useful command to view the log file
		- <shift>-G jumps to the last line of the log file
		- <shift>-F will monitor the log file
			* <ctrl>-C to stop monitor mode
		- q to quit

# Stage 2: DNSSEC

1. Edit /etc/bind/named.conf.options
	1. Enable DNSSEC

// Add this inside the options {} statement

dnssec-enable true;

1. Create /etc/bind/keys and cd to it
	1. mkdir /etc/bind/keys
	2. cd /etc/bind/keys
2. Generate DNSSEC Zone Signing Key for your domain
	1. **This step needs to be run from the /etc/bind/keys directory**
	2. dnssec-keygen –r /dev/urandom –a NSEC3RSASHA1 –b 1024 –n ZONE **SUBZONE**
		1. Never use the ‘-r /dev/urandom’ on a real server. It is included here for speed reasons
	3. Note the name of your Zone Signing Key in the table below. Be aware that dnssec-keygen returns the key name without the ‘.key’ extension. **Add the ‘.key’ extension when recording the key file name**
3. Generate DNSSEC Key Signing Key
	1. **This step needs to be run from the /etc/bind/keys directory**
	2. dnssec-keygen –r /dev/urandom –f KSK –a NSEC3RSASHA1 –b 2048 –n ZONE **SUBZONE**
	3. Note the name of your Key Signing Key in the table below. Be aware that dnssec-keygen returns the key name without the ‘.key’ extension. **Add the ‘.key’ extension when recording the key file name**

|  |  |  |
| --- | --- | --- |
| ***Key Name*** | ***Used in Lab*** | ***File Name*** |
| Zone Signing Key | ZONE\_SIGNING\_KEY |  |
| Key Signing Key | KEY\_SIGNING\_KEY |  |

1. Edit your zone file and $INCLUDE the two keys you noted in the table above
	1. e.g. /var/cache/bind/db.**SUBZONE**
	2. Remember to increment the zone’s serial number

; Append these two lines to your zone file.

$INCLUDE /etc/bind/keys/ZONE\_SIGNING\_KEY

1. Create a signed zone file
	1. **This step needs to be run from the /etc/bind/keys directory. It is all on one line.**dnssec-signzone –r /dev/urandom -k **KEY\_SIGNING\_KEY** -o **SUBZONE** -t -A /var/cache/bind/db.**SUBZONE** **ZONE\_SIGNING\_KEY**
2. Verify the zone file is signed correctly
	1. named-checkzone **SUBZONE** /var/cache/bind/db.**SUBZONE**.signed
	2. Should mention that the zone is signed

zone "**SUBZONE**" IN {

type master;

file "/var/cache/bind/db.**SUBZONE.*signed***";

allow-update { none; };

};

Figure 2: Update /etc/bind/named.conf.local to use signed zone file

1. Reload the DNS server
	1. rndc reload
2. Verify DNSSEC is working correctly
	1. dig +dnssec centos.**SUBZONE** @**DEBIAN\_IP**
	2. **The dig output should contain a number of RRSIGS that contain long alphanumeric strings. These are the signatures for each record**

# Questions

1. Nowhere in this lab exercise did we modify the /etc/resolv.conf file to use the DNS server we created. Why does this work?
2. Can you look up a host name in your sub domain from your host workstation? Please explain your answer and why you think it does or does not work.
	1. E.g. does ‘dig debian.**SUBZONE**’ work on your host workstation.
3. Why is using /dev/urandom in the real world a bad idea? The manual page for random will likely help with answering this question. (man 4 random) Explain in your own words how this could be exploited in practice.