# Overview

## Goals

You will configure a NTP server on your Debian VM that syncs time from the lab time source. Your CentOS VM and Pi will be configured to sync to your Debian VM. Next, you will set up a syslog server on your Debian VM and send logs to it from your CentOS VM and Pi. Finally you will deploy and configuration a secure shell (SSH) environment. You will learn how to install SSH, generate keys, edit the sshd configuration file to specify different types of authentication, and configure denyhosts. You will then use your client VM to log on to the server VM using SSH.

## Deliverables (printed copy with cover and a backup to Canvas)

The printed copy of the lab will be as usual. Include introductory and concluding paragraphs, be sure to document all entities with \*\* and answer all questions. Upload copies of your configuration files and the ***authorized\_keys*** and ***sshd\_config*** files to Canvas.

**Reminder:** all information should be text files except when a screenshot is required!

## Information tables\*\*

This lab will use a Debian Virtual machine as the secure shell server and your host machine as the client. Throughout this document, the terms host and client are used interchangeably to refer to a terminal on your physical host machine. The first information table is to aid documenting the NTP andSyslog setup, the second table is for the SSH setup.

Table 1: NTP and Syslog

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Formula** | **Used in Lab** | **Answer** |
| Assigned Subnet (See 3110 Web) | *n* | - |  |
| Debian VM | 172.16.*(n)*.10 | DEBIAN\_IP |  |
| Raspberry Pi | 172.16.*(n)*.20 | PI\_IP |  |
| CentOS VM | 172.16.*(n)*.50 | CENTOS\_IP |  |
| NTP Source | 172.16.1.253 | 172.16.1.253 |  |

Table 2: SSH Data

|  |  |  |
| --- | --- | --- |
| **Description** | **Used in Lab** | **Value** |
| Debian Username | **DEBIAN\_USER** |  |
| Debian Password | - |  |
| Debian IP Address | **DEBIAN\_IP** |  |

# Lab

Before starting make sure that network configuration for ***eth0*** and ***lo*** interfaces are valid. Also, make sure your Linux has a valid DNS server configuration. Remember ***eth0*** and ***lo*** are typical names, they may vary on your systems.

## NTP

### Check the Lab’s NTP server

1. Install packages “ntp” and “ntpdate”

aptitude –R install ntp ntpdate

1. Run *ntpdate*
 /usr/sbin/ntpdate –q 172.16.1.253
2. An example report from ntpdate is shown below. The stratum is highlighted in the report below.

|  |
| --- |
| prompt$ ntpdate -q clock1.unc.eduserver 152.2.21.1, **stratum 2**, offset 0.002288, delay 0.0514517 Jan 21:47:01 ntpdate[800]: adjust time server 152.2.21.1 offset 0.002288 sec |

### Configuring Debian VM NTP Daemon

1. Edit /etc/ntp.conf
	1. Allow servers on local subnet to query our server
	2. Comment out all lines beginning with ‘server’
	3. Add a new line for the Lab’s NTP source

|  |
| --- |
| # Allow clients on 127.16.1.0/24 to query this serverrestrict 172.16.*(n)*.0 mask 255.255.255.0 nomodify notrap# Commented out these servers#server 0.debian.pool.ntp.org#server 1.debian.pool.ntp.org#server 2.debian.pool.ntp.org#server 3.debian.pool.ntp.org# Added a line for the Lab’s NTP sourceserver 172.16.1.253 |

1. Stop the NTP service
	1. /etc/init.d/ntp stop
2. Set the clock manually
	1. /usr/sbin/ntpdate 172.16.1.253
	2. This step is not necessary but it should help your VM’s clock synchronize with the lab’s time source more quickly
3. Start the NTP service
	1. /etc/init.d/ntp start
4. \*\* Check NTP server
	1. ntpdate –q **DEBIAN\_IP**
	2. Note: As long as you see a non-zero value for “stratum” you’re fine and can ignore any warning. However it may take several minutes for your NTP server to synchronize and report its stratum correctly

### Configuring CentOS VM NTP Daemon

1. You should not configure the CentOS NTP daemon until the Debian NTP daemon has synchronized.
2. Edit /etc/ntp.conf
	1. Comment out all lines beginning with ‘server’
	2. Add a new line for your Debian VM’s NTP source

|  |
| --- |
| # Commented out these serversserver 0.rhel.pool.ntp.orgserver 1.rhel.pool.ntp.orgserver 2.rhel.pool.ntp.org# Added a line for my Debian VM’s NTP sourceserver **DEBIAN\_IP** |

1. Stop NTP service
	1. ~~/etc/init.d/ntpd stop~~
	2. service ntpd stop
	3. Note: stop, start, restart and status are all options that can be used by service
2. Set the clock manually
	1. /usr/sbin/ntpdate **DEBIAN\_IP**
	2. This step is not normally necessary but it should help your VM’s clock synchronize to your Debian time source more quickly
3. Start NTP service
	1. /etc/init.d/ntpd start
4. \*\* Check NTP
	1. /usr/sbin/ntpdate –q **CENTOS\_IP**
	2. Note: As long as you see a non-zero value for “stratum” you’re fine and can ignore any warning. However it may take several minutes for your NTP server to synchronize and report its stratum correctly
	3. If your CentOS VM is having trouble synchronizing try commenting out the dummy stratum 10 server from the ntp.conf file.

### Configuring the Raspberry Pi NTP Daemon

By this time you should be an NTP pro. Find the Raspbian NTP configuration file and configure it to use the Debian NTP service.

After restarting the daemon, check that the Pi is getting the time from the Debian server.

**Note:** if ntpdate is not found on your Pi iInstall ntpdate from <http://lab302-repo.hades.lab>. If you are at home download “ntpdate deb file” from an appropriate source and install it using following command:

dpkg –i ntpdate\_4.2.6.p5+dfsg-3\_armhf.deb

## Syslog

### Configuring Debian VM Syslog Daemon

Debian uses *rsyslogd*, an enhanced replacement for the venerable syslog package. We will not be using all of its advanced features for this lab.

1. Edit */etc/rsyslog.conf*
	1. Enable Logging of remote syslog messages
	2. Send all log messages to a remote syslog server

|  |
| --- |
| # Uncommented the following lines. These enable# Logging of incoming UDP messages from the network.# This is normally achieved with the ‘-r’ option to syslog$ModLoad imudp$UDPServerRun 514 |

1. Restart syslog server
	1. /etc/init.d/rsyslog restart

### Configuring CentOS VM Syslog Daemon

1. Edit */etc/rsyslog.conf*
	1. Send all log messages to your Debian VM. Ensure the following is in the file.

|  |
| --- |
| # Transmit all logs to my Debian VM\*.\* @**DEBIAN\_IP** |

1. Restart syslog server
	1. ~~/etc/init.d/syslog restart~~
	2. service rsyslog restart
2. \*\* Test syslog
	1. logger “hello from userid on CentOS VM”
	2. Substitute your userid in the above command
	3. The above message should appear in /var/log/syslog on your Debian VM.
	4. \*\* Document files in report

### Configure the Pi Syslog Daemon

Follow the steps for the CentOS Syslog configuration. Don’t forget to test with *logger*.

## SSH

### Installing Secure Shell

Note: Hopefully, you have used ssh server from both your Debian server VM and your CentOS client VM. In case you have not completed the openssh server install:

To install the secure shell daemon on your server, you will use aptitude. As root on your Debian VM, execute:

# aptitude -R install openssh-server

This installs the daemon, generates server keys, and starts sshd automatically.

### Determining Server IP Address

With your Debian VM, verify your IP address with the following command:

$ /sbin/ifconfig

Record this IP address in the Information table at the beginning of this document. You will need to use this IP address when using SSH to log in to your server.

### Testing Login with SSH

With the default sshd configuration file, password authentication will be enabled. From your host-machine (client), try to log in to your server VM using SSH.

$ ssh **DEBIAN\_USER**@**DEBIAN\_IP**

You can type a few commands to verify you have access to the server. Type exit to close the session.

### Installing DenyHosts

DenyHosts monitors the secure log (/var/log/auth.log on Debian) for failed login attempts. Its configuration file is in */etc/denyhosts.conf*, but the default configuration is acceptable for our uses.

# aptitude –R install denyhosts

This installs the daemon and starts denyhosts automatically.

### Testing DenyHosts

DenyHosts has different policies depending on whether a failed attempt is for a non-existent account, a normal account, or the root account. Access to the root account is the strictest and therefore the easiest to test.

Run the SSH command twice and simply hit enter at the password prompts.

$ ssh root@**DEBIAN\_IP**

Wait a moment, as blocking IP addresses is not instantaneous and then try to log in again using SSH. You should see the error below.

$ **ssh root@DEBIAN\_IP**

ssh\_exchange\_identification: Connection closed by remote host

You can also see the blocked IP address *in /etc/hosts.deny* on the server.

### White-Listing an IP Address or Network

Occasionally you will need to white-list an IP address or network to prevent DenyHosts from blocking it. This can be achieved by editing */etc/hosts.allow*. In this lab, it makes sense to white-list the entire lab’s network, as below.

# White-list the Lab’s network

sshd: 172.16.0.0/255.255.0.0

After white-listing the network, you should be able to log in from your client again. **This step is required to continue the lab.**

### Generating Keys

We will next create a public/private key pair so we can connect more securely. On your client, execute the following command.

$ ssh-keygen -t rsa -b 4096

This step requests a password that will be used to encrypt your private key on disk. It is not essential to set a password, but it adds an extra layer of security. For this lab, DO NOT enter a password here.

This command will create two keys: *id\_rsa* and *id\_rsa.pub* *in ~/.ssh/.*

### Copying Public Key to Server

We need to copy our public key to the server. The simplest method is to use secure shell to transfer the key. Run the following command on the client to copy your public key to the server.

$ scp ~/.ssh/id\_rsa.pub **DEBIAN\_USER**@**DEBIAN\_IP:.**

The above command transfers your public key to your home directory on the server. In UNIX parlance, ‘.’ is the current directory, which defaults to your home directory when doing a secure copy.

### Authorizing Public Key on the Server

Next, we will authorize your public key to authenticate your user on the server. We first need to create an empty authorized\_keys file and configure its permissions correctly.

$ mkdir ~/.ssh

$ touch ~/.ssh/authorized\_keys

$ chmod 700 ~/.ssh

$ chmod 600 ~/.ssh/authorized\_keys

**Reminder:** The chmod command configures permissions for files and directories in Linux. The above example is using the octal notation. 7=RWX, 6-RW-, 5=R-X, etc. First group is owner, then group, then world.

Once the *authorized\_keys* file has been created, we need to ***append*** our public key to it.

$ cat id\_rsa.pub >> ~/.ssh/authorized\_keys

Keys should always be appended to the *authorized\_keys* file as you may have multiple keys for multiple machines.

### Modifying sshd\_config

The configuration file for the SSH daemon is located in */etc/ssh/sshd\_config* on your server. We will be modifying lines relating to authentication. Using your editor of choice you will need to locate and modify certain lines as follows:

# Make sure that we can authenticate using our public/private key pair

PubkeyAuthentication yes

# Disable logins using standard passwords

PasswordAuthentication no

You will need to restart sshd after your modifications to enable these changes. Execute:

# /etc/init.d/ssh restart

### Testing Key-Based Login

From your client, try to log on to your server using SSH. You can also log on using the normal method:

$ ssh **DEBIAN\_USER**@**DEBIAN\_IP**

Assuming that you did not set a password on your private key, you should now have a shell on your Debian VM.

If your private key were not located *in ~/.ssh/id\_dsa* or *~/.ssh/id\_rsa*, you would need to specify its location with as follows:

$ ssh **USER**@**HOST** -i **KEYFILE**

### SSH and the Pi

An ssh client should be part of the Raspbian Linux. Try logging onto the Debian SSH Server with ssh on the PI.

* What error occurs?
* What change to the Debain SSH server configuration do you need to make to allow the Pi to ssh to the Debian server using standard passwords?

Make the change to the Debian SSH configuration file and test. Document it working.\*\*

Depending on your preference you may leave the change when done or remove the change.

## Questions

1. What does nano’s ‘-w’ option do?
2. What does aptitude’s ‘-R’ option do? Use the man page for aptitude to answer this question.
3. What is the stratum of the Lab’s NTP source?
4. What is the stratum of your Debian’s NTP server?
5. What is the stratum of your CentOS client?
6. What does the “\*.\*” stand for when configuring the syslog destination?
7. If you do not create a passphrase for your ssh private key, what would happen if someone were able to copy your private key file?
8. Using key based authentication with SSH and having no passphrases for the private key is very helpful in some circumstances. What is one of these circumstances? (Think about using scp)
9. What, if any, are the security concerns of someone stealing your public key?