# Overview

## Goals

This lab aims to show you how to create a working SMTP server using Postfix. This includes the creation of a TLS key and certificate pair that can be used to encrypt communication to and from the SMTP server. This lab will also cover creating a Sender Policy Framework (SPF).

## Deliverables

* Lab Report including the following
  + 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
  + Appendix A: Properly titled “Debian main.cf”
    - /etc/postfix/main.cf (from Debian)
  + Appendix B: Properly titled “SPF Dig Output”
    - Output of the 'dig –t txt **SUBZONE**' command

## Information

The IP addresses, networks, and sub domains used throughout this lab document are for example only. Fill out the table below and refer to it when following this lab’s directions.

|  |  |  |  |
| --- | --- | --- | --- |
| ***Description*** | ***Formula*** | ***Used in Lab*** | ***Answer*** |
| Assigned Subnet Number | *n* | - |  |
| Your Last Name | *<last\_name>* | **LAST\_NAME** |  |
| Debian VM IP | 172.16.*n*.10 | **DEBIAN\_IP** |  |
| CentOS VM IP | 172.16.*n.*50 | **CENTOS\_IP** |  |
| DNS Subzone | *<last\_name>*.hades.lab  (e.g. watson.hades.lab) | **SUBZONE** |  |

Transfer these key names from your DNS lab

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

# Installing Postfix (Debian)

We need to remove exim4 at the same time we install postfix to prevent the system from breaking. (SMTP daemons are important in the UNIX/Linux world.)

When using aptitude, placing an underscore after a package name will mark it as 'purged'. Purging a package will remove both the package and its configuration from the system.

1. Install postfix
   1. aptitude -R install postfix postfix-policyd-spf-perl exim4\_ exim4-base\_ exim4-config\_ exim4-daemon-light\_
   2. Leave all questions set to their default value
   3. We will correct the mail name later
   4. This removes exim4 at the same time
2. Install mailx
   1. aptitude –R install mailx
   2. This may already be installed in your system
3. Configure the DNS MX record for your domain
   1. Add the following line to your zone file (e.g. /var/cache/bind/db.**SUBZONE**)
   2. Do not forget to update the zone’s serial number

@ IN MX 10 debian.**SUBZONE**.

1. Update zone’s DNSSEC information
   1. **Remember to run this command in /etc/bind/keys/**
   2. dnssec-signzone -r /dev/urandom -k KEY\_SIGNING\_KEY -o SUBZONE -t –A /var/cache/bind/db.SUBZONE ZONE\_SIGNING\_KEY
2. Reload the DNS server
   1. rndc reload
3. Verify the MX record
   1. dig -t mx **SUBZONE**
   2. Should return the MX record you configured above

**NOTE**: If you do not get a SUCCESSFUL dig response, do not continue with the lab. You will not be able to successfully implement email without a working DNS subzone.

# Generating SSL Key (Debian)

By default, Debian creates a 'snake oil' certificate for SSL traffic. It is referred to as ‘snake oil’ because it has no benefit (e.g. no security). You will need to create your own certificate to replace the snake oil certificate. In real life, you would create a Certificate Signing Request (CSR) to send to a Certificate Authority (CA) who would generate your certificate. In this example we will skip the CA and generate our certificate directly.

1. Create the directory /etc/postfix/keys and cd into it.
   1. mkdir /etc/postfix/keys
   2. cd /etc/postfix/keys
2. Use the OpenSSL command line client to generate an RSA private key
   1. openssl genrsa -out debian.**SUBZONE**.key 1024
   2. Use your Fully Qualified Domain Name (FQDN) for the key name.
3. Generate a certificate using the private key you just created
   1. openssl req -new -key debian.SUBZONE.key -x509 -out debian.SUBZONE.cert
   2. **For the "Common Name (eg, YOUR name)" prompt, enter your server's FQDN** (e.g. debian.**SUBZONE**)
   3. The argument to the ‘-key’ option is the key you created in the step above
   4. You would normally create a CSR here instead of creating the key directly

# Configuring Postfix (Debian)

1. Edit /etc/mailname
   1. File is only one line
   2. Only contents in the system’s hostname
   3. E.g. debian.**SUBZONE**
2. Edit /etc/postfix/main.cf
   1. Find and edit the lines shown in the example below

# SSL Certificate and Private key created in steps above

smtpd\_tls\_cert\_file=/etc/postfix/keys/debian.**SUBZONE**.cert

smtpd\_tls\_key\_file=/etc/postfix/keys/debian.**SUBZONE**.key

# hostname of this machine, should match contents

# of /etc/mailname

myhostname = debian.**SUBZONE**

# These are valid mail destinations for this computer.

# This server will accept mail addressed to these domains

# from anywhere on the Internet

mydestination = **SUBZONE**, localhost

# These are trusted networks. They are permitted to send

# mail addressed to anywhere on the Internet through this

# server.

#

# Trusted networks are: loopback, Debian IP (me), and

# CentOS IP

mynetworks = 127.0.0.0/8, **DEBIAN\_IP**/32, **CENTOS\_IP**/32

1. Restart postfix
   1. /etc/init.d/postfix restart
   2. Check /var/log/mail.log to see if any errors were reported
2. Verify mail server works
   1. **Run these tests from your CentOS system**
   2. If an error occurs, either system may generate a mail message
      1. Use ‘mail’ command to read messages
      2. Watch files /var/spool/mail/ for mail delivered to other users
   3. Watch log files on both systems
      1. /var/log/mail.log (Debian)
      2. /var/log/maillog (CentOS)
   4. Attempt to connect to your SMTP server using SSL
      1. openssl s\_client -connect debian.**SUBZONE**:25 -starttls smtp
      2. Copy the output and paste into your lab report
      3. We do not need to do anything with this connection so hit <CTRL>-C to quit
   5. Try sending a message to yourself using mail
      1. Send mail to your Debian username at your subzone
      2. E.g. jwatso8@**SUBZONE**
      3. Reference the example below (**bold** is text entered)
      4. You should be able to read the message on your Debian system (log in using your normal user account)

centos# **mail jwatso8@*SUBZONE***

**Subject: Testing mail from CentOS**

**This is just a test. Remember to end your mail with a period on an empty line.**

**.**

CC:

centos#

## mail Primer

The mail command is an ancient method of reading mail on a command line. It is useful as a last resort. It takes simple commands followed by the enter key at the ‘&’ prompt. Below is a short list of mail commands that will help you use it.

* h – show headers (subject lines)
* # – read Nth message (# is a number of a message in the header list)
* d# – delete the Nth message
* q – quit mail

# Configure Sender Policy Framework

This only configures the DNS-side of SPF. This SPF record would be used by mail servers at other domains trying to verify incoming email that has an envelope sender from your domain.

1. Configure the DNS SPF record for your domain
   1. Add the following line to your zone file (e.g. /var/cache/bind/db.**SUBZONE**)
   2. Do not forget to update the zone’s serial number

; SPF record

@ IN TXT “v=spf1 ip4:**DEBIAN\_IP** –all”

;

; Empty Sender ID PRA record to prevent Sender ID from

; evaluating our SPF record to determine PRA

@ IN TXT “spf2.0/pra ?all”

1. Update zone’s DNSSEC information
   1. **Remember to run this command in /etc/bind/keys/**
   2. dnssec-signzone -r /dev/urandom -k **KEY\_SIGNING\_KEY** -o SUBZONE -t –A /var/cache/bind/db.SUBZONE **ZONE\_SIGNING\_KEY**
2. Reload the DNS server
   1. rndc reload
3. Verify the SPF record
   1. dig –t txt **SUBZONE**
   2. Should return the SPF record you configured above
   3. Paste the output from this step into your lab report

# Configure SPF in Postfix (Debian)

These steps configure your postfix installation to honor SPF records set by other domains. This will allow your mail server to verify the envelope sender address on inbound email.

1. Edit /etc/postfix/main.cf, adding the lines below

# Add SPF checking to the recipient restrictions.

# default: permit\_mynetworks, reject\_unauth\_destination

#

# This should be entered as one line in your main.cf

smtpd\_recipient\_restrictions = permit\_mynetworks,

reject\_unauth\_destination, check\_policy\_service

unix:private/spfcheck

policy\_time\_limit = 3600

1. Edit /etc/postfix/master.cf, adding the line below
   1. **The text below should be entered on one line**

spfcheck unix - n n - 0 spawn user=policyd-spf argv=/usr/sbin/postfix-policyd-spf-perl

1. Restart postfix
   1. /etc/init.d/postfix restart
   2. Check /var/log/mail.log to see if any errors were reported
2. Repeat Step 4 of “Configuring Postfix (Debian)”, “Verify mail server works”
   1. Check the mail you sent and note if it has a Received-SPF header or not

# Questions

1. What is the MX for the email domain 'hades.lab'. Please explain exactly how you found this information and how finding such information might be useful.
2. What is the SPF policy for the email domain 'hades.lab'. Describe what each part of the policy means.
3. Describe the SPF policy that you configured for your domain. For reference, it was “v=spf1 ip4:DEBIAN\_IP –all”
4. Describe the following SPF policies and what situations they might be useful. If a policy does not appear useful, explain why.
   1. “v=spf1 –all”
   2. “v=spf1 all”
   3. “v=spf1 ip4:10.10.10.0/24 A:google.com ~all”
5. There is a mistake in the SPF record below. Point out the mistake and explain why you feel it is a mistake.
   1. “v=spf1 ip4:10.10.10.0/24 ip4:10.10.20.0/24 ip4:10.10.30.0/24 ip4:10.10.40.0/24 all”
6. Write *one* SPF rule that meets the following conditions:
   1. 192.168.18.0/255.255.254.0 is permitted to send mail
   2. 192.168.20.0/255.255.254.0 is prohibited from sending mail
   3. No information on if 192.168.22.0/255.255.254.0 should send mail
   4. All other systems should never send mail