



NETLAB+®



Security+ Lab Series

Lab 14: Implementing Common Protocols and Services for Basic Security Practices

Document Version: **2018-08-28**

Copyright © 2018 Network Development Group, Inc.
www.netdevgroup.com

NETLAB Academy Edition, NETLAB Professional Edition, NETLAB+ Virtual Edition, and NETLAB+ are registered trademarks of Network Development Group, Inc.

Contents

Introduction	3
Objectives.....	3
Lab Topology	4
Lab Settings	5
1 Configuring a Proxy Server.....	6
1.1 Configuring Squid	6
1.2 Configuring SquidGuard	11
1.3 Configure & Test Firefox Proxy Settings.....	15
2 Configuring and Enabling SSL for HTTP Services.....	18
2.1 Generating a Server Key and Server Certificate.....	18
2.2 Configure Apache to Utilize SSL	22
2.3 Configuring & Testing HTTPS Test Page	27

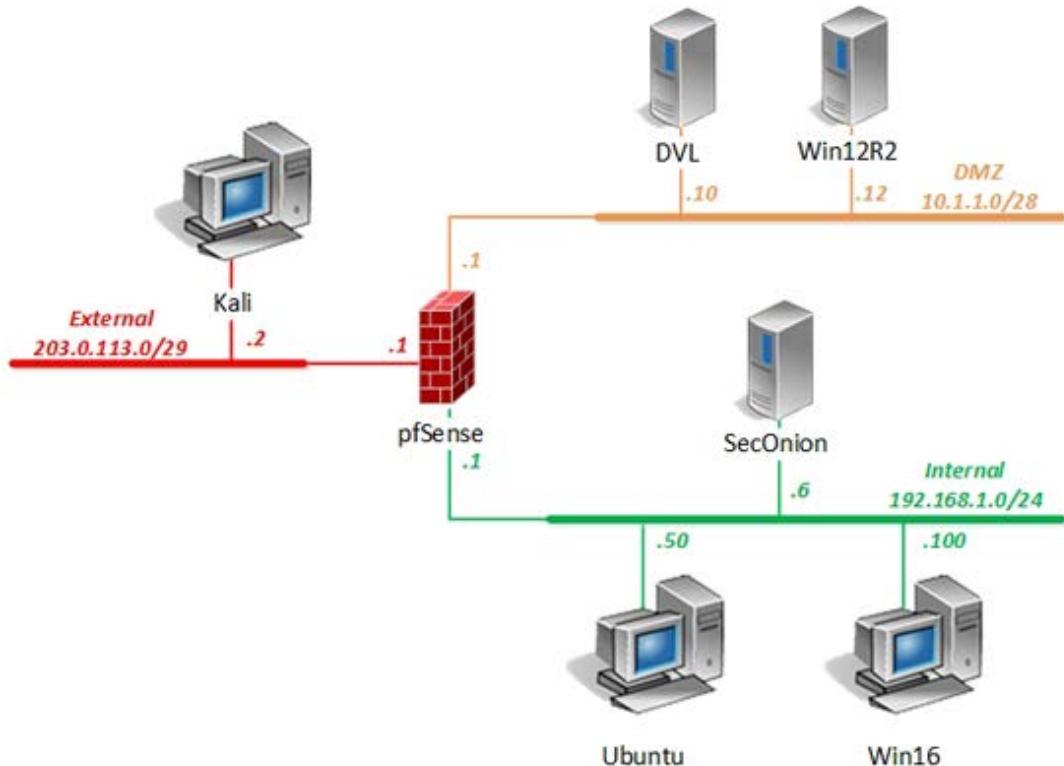
Introduction

In this lab, you will be conducting network security practices by implementing common protocols.

Objectives

- Install and configure network components, both hardware, and software-based, to support organizational security
- Given a scenario implement secure protocols

Lab Topology



Lab Settings

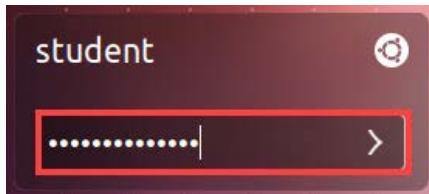
The information in the table below will be needed to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
DVL	10.1.1.10 /28	root	toor
Kali	203.0.113.2 /29	root	toor
pfSense	eth0: 192.168.1.1 /24 eth1: 10.1.1.1 /28 eth2: 203.0.113.1 /29	admin	pfsense
Sec0nion	192.168.1.6 /24	soadmin	mypassword
		root	mypassword
Ubuntu	192.168.1.50 /24	student	securepassword
		root	securepassword
Win12R2	10.1.1.12 /28	administrator	Training\$
Win16	192.168.1.100 /24	lab-user	Training\$
		Administrator	Training\$

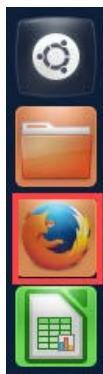
1 Configuring a Proxy Server

1.1 Configuring Squid

1. Launch the **Ubuntu** virtual machine to access the graphical login screen.
2. Log in as **student** with **securepassword** as the password.



3. Open a web browser by clicking on the **Firefox** icon located in the left menu pane.



4. Within the **Firefox** web browser, type **192.168.1.1** into the *address bar*, followed by pressing **Enter**.



- Once presented with the *pfSense* login page, type **admin** as the *username* and **pfSense** as the *password*. Click **Login**.



Please
Note

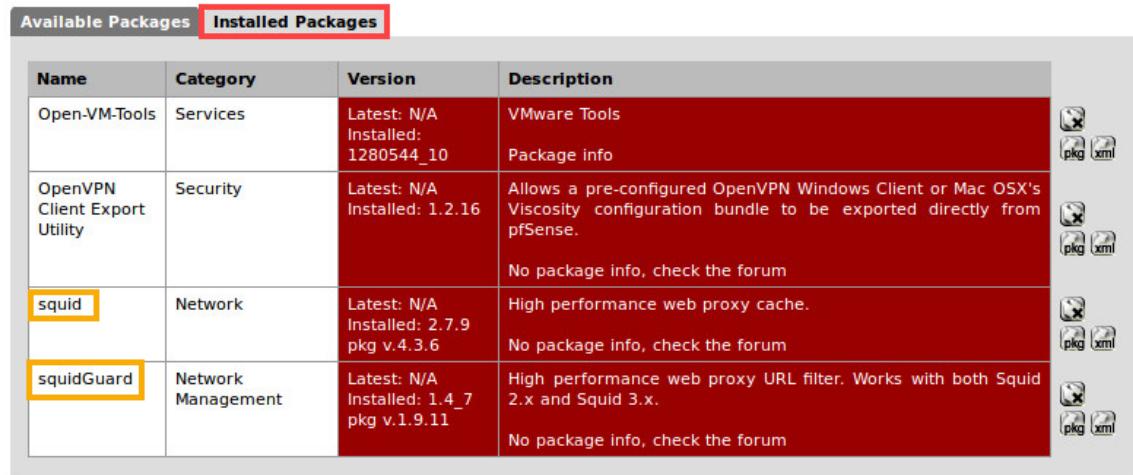
If the *pfSense* graphical user interface does not load immediately, wait an additional 1-2 minutes for the system to finish booting.

- Once logged in, focus on the top menu pane and navigate to **System > Packages**.



7. Make sure to view the **Installed Packages** tab. Verify that both *squid* and *squidguard* packages are installed.

System: Package Manager



Name	Category	Version	Description
Open-VM-Tools	Services	Latest: N/A Installed: 1280544_10	VMware Tools Package info
OpenVPN Client Export Utility	Security	Latest: N/A Installed: 1.2.16	Allows a pre-configured OpenVPN Windows Client or Mac OSX's Viscosity configuration bundle to be exported directly from pfSense. No package info, check the forum
squid	Network	Latest: N/A Installed: 2.7.9 pkg v.4.3.6	High performance web proxy cache. No package info, check the forum
squidGuard	Network Management	Latest: N/A Installed: 1.4_7 pkg v.1.9.11	High performance web proxy URL filter. Works with both Squid 2.x and Squid 3.x. No package info, check the forum

8. Once verified, navigate to **Services > Proxy server**.



9. While on the **General** tab, select **INTERNAL_GW** and **DMZ_GW** for the *Proxy interface*. To do so, hold the **CTRL** key and select each entry until both are highlighted.

Proxy server: General settings

The interface(s) the proxy server will bind to.

10. **Uncheck** the checkbox next to *Allow users on interface*.

If this field is checked, the users connected to the interface selected in the 'Proxy interface' field will be allowed to use the proxy, i.e., there will be no need to add the interface's subnet to the list of allowed subnets. This is just a shortcut.

11. **Check** the checkbox next to *Transparent proxy* to enable this feature.

If transparent mode is enabled, all requests for destination port 80 will be forwarded to the proxy server without any additional configuration necessary.

12. Scroll down until you see the *Enabled logging* entry. **Check** the checkbox to enable.

This will enable the access log. Don't switch this on if you don't have much disk space left.

13. Verify that the *Log store directory* is configured to **/var/squid/logs**.

The directory where the log will be stored (note: do not end with a / mark)

14. Type the number **7** as the value for *Log rotate*.

Defines how many days of logfiles will be kept. Rotation is disabled if left empty.

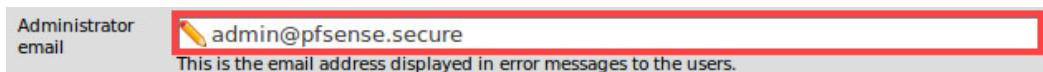
15. Use port number **3128** as the *Proxy port*.

This is the port the proxy server will listen on.

16. For *Visible hostname*, type **proxy.pfsense.secure**.

This is the URL to be displayed in proxy server error messages.

17. For the *Administrator email*, type **admin@pfsense.secure**.



Administrator
email This is the email address displayed in error messages to the users.

18. Scroll to the bottom of the page and click **Save**.

19. Once saved, click on the **Cache Mgmt** tab. Change the value for *Hard disk cache size* to **50**.



Proxy server: Cache management

General Upstream Proxy **Cache Mgmt** Access Control Traffic Mgmt Auth Settings Local Users

Hard disk cache size This is the amount of disk space (in megabytes) to use for cached objects.

20. Scroll to the bottom of the page and click **Save**.

21. Next, click on the **Access Control** tab. Enter the subnets mentioned below into the *Allowed subnets* field:

- a. **192.168.1.0/24**
- b. **10.1.1.0/28**

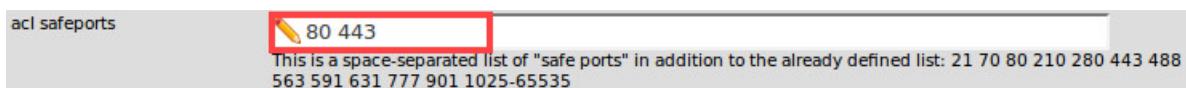


Proxy server: Access control

General Upstream Proxy Cache Mgmt **Access Control** Traffic Mgmt Auth Settings Local Users

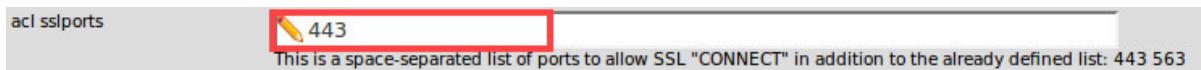
Allowed subnets Enter each subnet on a new line that is allowed to use the proxy. The subnets must be expressed as CIDR ranges (e.g.: 192.168.1.0/24). Note that the proxy interface subnet is already an allowed subnet. All the other subnets won't be able to use the proxy.

22. Scroll towards the bottom of the page until you see *acl safeports*. Type **80** and **443** into the *white space*.



acl safeports This is a space-separated list of "safe ports" in addition to the already defined list: 21 70 80 210 280 443 488 563 591 631 777 901 1025-65535

23. Type **443** for *acl sslports*.



acl sslports This is a space-separated list of ports to allow SSL "CONNECT" in addition to the already defined list: 443 563

24. Click the **Save** button.

25. Click on the **Traffic Mgmt** tab. For *Maximum download size*, enter the value **500000** to represent *500MB* as the maximum download file size.

26. For *Maximum upload size*, enter the value **50000** to represent *50MB* as the maximum upload file size.

27. Scroll towards the bottom and click the **Save** button.

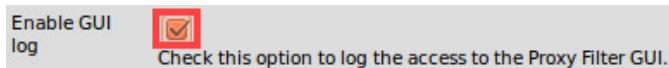
1.2 Configuring SquidGuard

1. While on the *pfSense web configurator*, navigate to **Services > Proxy filter**.

2. On the **General settings** tab, check the checkbox next to **Enable**.

Proxy filter SquidGuard: General settings

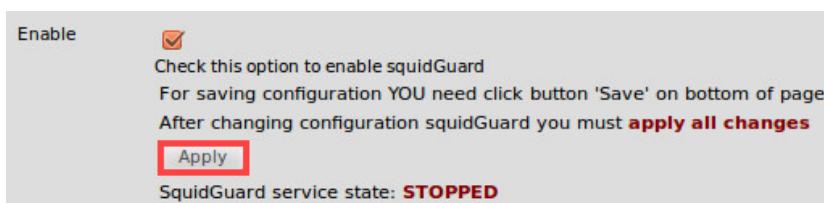
3. Scroll down until you see *Enable GUI log*. Check the checkbox to enable this feature.



4. Check the box for *Enable log*.



5. Scroll to the bottom of the page and click the **Save** button.
 6. Once the page reloads, click the **Apply** button located towards the top of the page.



7. Next, click on the **Target categories** tab. Click the **Add a new item** icon.

Proxy filter SquidGuard: Target categories

General settings | Common ACL | Groups ACL | **Target categories** | Times | Rewrites | Blacklist | Log | XMLRPC Sync

Name	Redirect	Description
		

8. For the *Name*, type **BList1**.

Name	 BList1
Enter a unique name of this rule here. The name must consist between 2 and 15 symbols [a-Z_0-9]. The first one must be a letter.	

9. For *Order*, select the drop-down box and choose --- **Last** ---.

Order	 -- Last --
Select the new position for this target category. Target categories are listed in this order on ALCs and are matched from the top down in sequence.	

10. Type **casino.com** into the whitespace area for *Domain List*.

Domain List

Enter destination domains or IP-addresses here. To separate them use space.
Example: mail.ru e-mail.ru yahoo.com 192.168.1.1

11. Type **casino** into the whitespace area for *Regular Expression*.

Regular Expression

Enter word fragments of the destination URL. To separate them use | . **Example:** mail|casino|game|\|.rsdf\$

12. Select the drop-down box next to *Redirect mode* and choose **int error page (enter error message)**.

Redirect mode

int error page (enter error message)

Select redirect mode here.
 Note: if you use 'transparent proxy', then 'int' redirect mode will not accessible.
 Options: ext url err page , ext url redirect , ext url as 'move' , ext url as 'found'.

13. Check the box for the *Log* entry.

Log

Check this option to enable logging for this ACL.

14. Click the **Save** button.

15. Click on the **Common ACL** tab. Click the **Show rules** icon within the red *Target Rules List* pane.

Proxy filter SquidGuard: Common Access Control List (ACL)

General settings	Common ACL	Groups ACL	Target categories	Times	Rewrites	Blacklist	Log	XMLRPC Sync
<div style="border-bottom: 1px solid #ccc; padding-bottom: 5px;"> Target Rules <div style="border: 1px solid #ccc; height: 10px; margin-top: 5px;"></div> </div> <div style="background-color: red; color: white; text-align: center; padding: 5px;"> Target Rules List (click here) </div>								

16. Notice *BList1* is added onto the list. For this entry, select the access drop-down box and choose **deny**. Click the drop-down box entry for *Default access [all]* and select **allow**.

The screenshot shows the 'Target Rules List' interface. At the top, there's a red bar with the title 'Target Rules List (click here)' and a delete icon. Below it, a message says 'ACCESS: 'whitelist' - always pass; 'deny' - block; 'allow' - pass, if not blocked.' The main area has a table with two rows:

Target Categories	
[BList1]	access deny
Default access [all]	access allow

17. Within the whitespace area for the *Proxy Denied Error*, type **Request denied by the XYZ. corp proxy**.

The screenshot shows the 'Proxy Denied Error' configuration. It contains a text input field with the placeholder 'Request denied by the XYZ.corp proxy'. Below the input field, a note states: 'The first part of the error message displayed to clients when access was denied. Defaults to "Request denied by \${product_name} proxy"'.

18. Select the drop-down box next to *Redirect mode* and choose **int error page (enter error message)**.

The screenshot shows the 'Redirect mode' configuration. It features a dropdown menu with the option 'int error page (enter error message)' selected. A note below the dropdown says: 'Select redirect mode here. Note: if you use 'transparent proxy', then 'int' redirect mode will not accessible. Options: ext url err page , ext url redirect , ext url as 'move' , ext url as 'found'.'

19. Check the box next to *Log*.

The screenshot shows the 'Log' configuration. It features a dropdown menu with the option 'int error page (enter error message)' selected. A note below the dropdown says: 'Select redirect mode here. Note: if you use 'transparent proxy', then 'int' redirect mode will not accessible. Options: ext url err page , ext url redirect , ext url as 'move' , ext url as 'found'.'

20. Click the **Save** button.

21. Once the page refreshes, click the **General settings** tab.

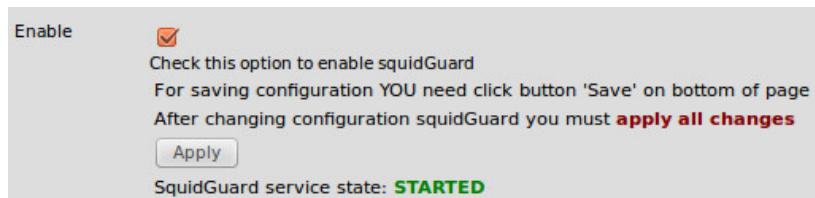
The screenshot shows the 'Proxy filter SquidGuard: General settings' page. The 'General settings' tab is highlighted with a red box. Other tabs include 'Common ACL', 'Groups ACL', 'Target categories', 'Times', 'Rewrites', 'Blacklist', 'Log', and 'XMLRPC Sync'.

22. Scroll to the bottom and click the **Save** button.

23. To apply all configurations, click the **Apply** button.

The screenshot shows the 'SquidGuard' configuration settings. It includes an 'Enable' checkbox which is checked. Below the checkbox, there's a note: 'Check this option to enable squidGuard'. Another note says: 'For saving configuration YOU need click button 'Save' on bottom of page'. A final note at the bottom says: 'After changing configuration squidGuard you must **apply all changes**'. There is a red box around the 'Apply' button. At the very bottom, it says 'SquidGuard service state: STOPPED'.

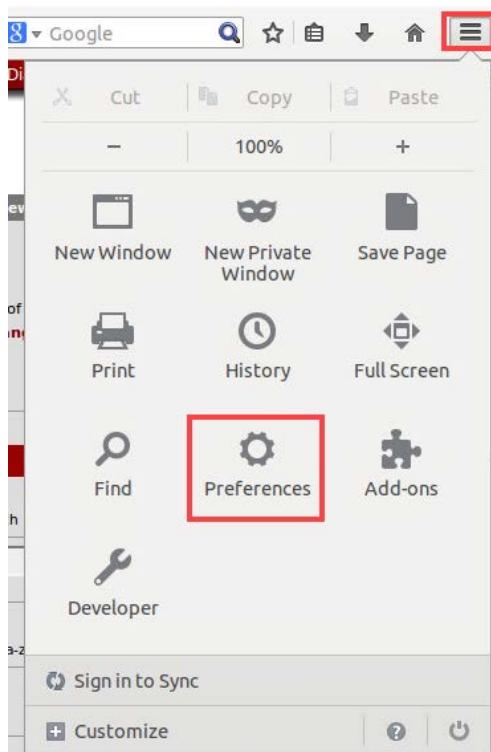
24. Verify that the *SquidGuard* service state is **STARTED**.



25. Leave the *Ubuntu* viewer open to continue with the next task.

1.3 Configure & Test Firefox Proxy Settings

1. While on the *Firefox* web browser, click the **settings** icon located to the top right corner followed by clicking on the **Preferences** icon.



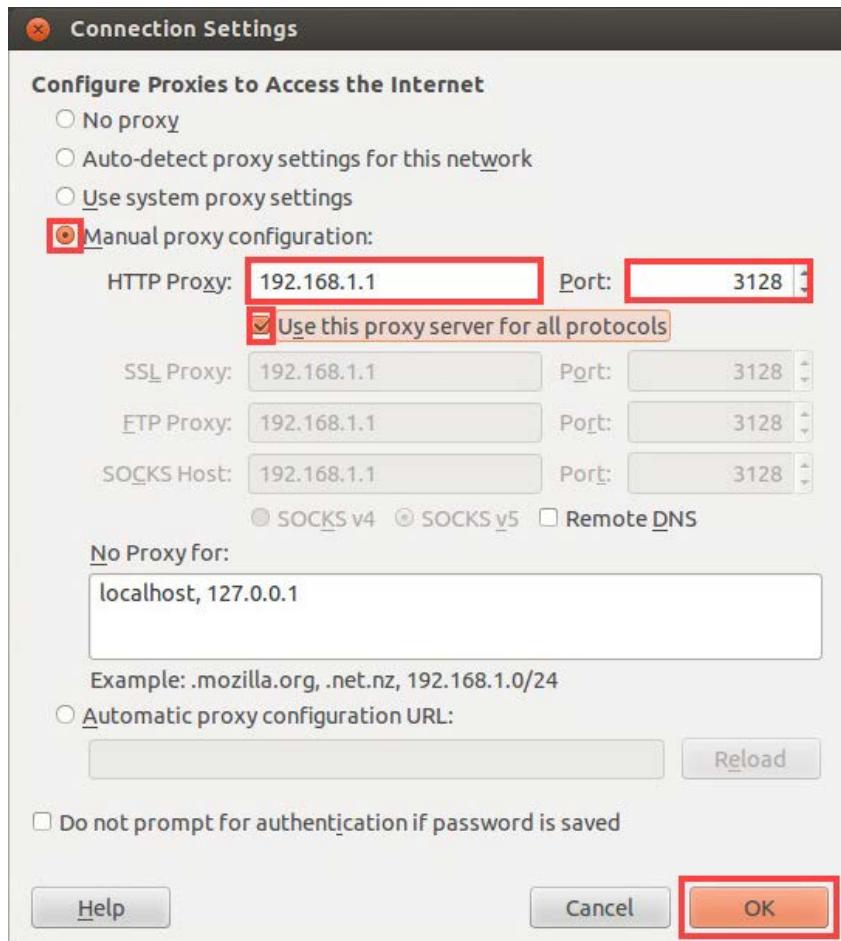
2. A new pop-up window appears. Click the **Advanced** icon at the top.



3. Select the **Network** tab and then click on the **Settings** button in the *Connection* pane.



4. Another pop-up window appears. Select the radio button for **Manual proxy configuration**. Type **192.168.1.1** as the *HTTP Proxy* and **3128** as the *Port*. Check the checkbox for **Use this proxy server for all protocols**. Click **OK**.



5. Back on the *Firefox Preferences* window, click the **Close** button.
6. Open a new tab in Firefox by clicking the "+" icon located at the top.

7. Type **casino.com** into the *address field* followed by pressing **Enter**.



A screenshot of a web browser window. The address bar shows "casino.com". A red box highlights the address bar. The main content area displays the following message:
Request denied by the XYZ.corp proxy: 403 Forbidden
Reason:
Client address: 192.168.1.50
Client group: default
Target group: BList1
URL: http://casino.com/



Notice the request has been denied with a message expressing the “blocked” statement with a *403 Forbidden* error.

8. **Close** the web browser.
9. Leave the *Ubuntu* viewer open to continue with the next task.

2 Configuring and Enabling SSL for HTTP Services

2.1 Generating a Server Key and Server Certificate

1. While on the *Ubuntu* system, open a new terminal window by clicking on the **terminal** icon located on the left menu pane.



2. Create a new directory by typing the command followed by pressing the **Enter** key.

```
student@Ubuntu: ~$ mkdir /tmp/ssl certs
```

```
student@Ubuntu:~$ mkdir /tmp/sslcerts
student@Ubuntu:~$
```

3. Change to the newly made directory.

```
student@Ubuntu: ~$ cd /tmp/ssl certs
```

```
student@Ubuntu:/tmp/sslcerts$ cd /tmp/sslcerts
student@Ubuntu:/tmp/sslcerts$
```

4. Verify that *OpenSSL* is installed on the system.

```
student@Ubuntu: /tmp/sslcerts$ openssl version
```

```
student@Ubuntu:/tmp/sslcerts$ openssl version
OpenSSL 1.0.1 14 Mar 2012
student@Ubuntu:/tmp/sslcerts$
```

5. Generate an *RSA server key*. When prompted for a *pass phrase*, type **secret** followed by pressing the **Enter** key. When prompted once more, type **secret** again. Press **Enter**.

```
student@Ubuntu:/tmp/sslcerts$ openssl genrsa -des3 -out server.key 1024
```

```
student@Ubuntu:/tmp/sslcerts$ openssl genrsa -des3 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
student@Ubuntu:/tmp/sslcerts$
```

6. Verify that the *server.key* has been generated.

```
student@Ubuntu:/tmp/sslcerts$ ls -l
```

```
student@Ubuntu:/tmp/sslcerts$ ls -l
total 4
-rw-rw-r-- 1 student student 963 Aug  8 11:01 server.key
student@Ubuntu:/tmp/sslcerts$
```

7. Generate the *Certificate Signing Request (CSR)* with the new **server.key**.

```
student@Ubuntu:/tmp/sslcerts$ openssl req -new -key server.key -out server.csr
```

- a. When prompted for the *server.key pass phrase*, type **secret**. Press **Enter**.
- b. During the signing request process, a series of questions will be asked. Type the information given below for each step followed by pressing **Enter**.
 - i. *Country Name*: **US**
 - ii. *State Name*: **TX**
 - iii. *Locality Name*: **Austin**
 - iv. *Organization Name*: **XYZ.corp**
 - v. *Organizational Unit Name*: Press **Enter**
 - vi. *Common Name*: **XYZ**
 - vii. *E-mail*: Press **Enter**
 - viii. *Challenge Password*: Press **Enter**
 - ix. *Company Name*: Press **Enter**

```
student@Ubuntu:/tmp/sslcerts$ openssl req -new -key server.key -out server.csr
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:TX
Locality Name (eg, city) []:Austin
Organization Name (eg, company) [Internet Widgits Pty Ltd]:XYZ.corp
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:XYZ
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
student@Ubuntu:/tmp/sslcerts$
```

8. Once completed with the wizard, verify that the *server.csr* has been created.

```
student@Ubuntu:/tmp/sslcerts$ ls -l
```

```
student@Ubuntu:/tmp/sslcerts$ ls -l
total 8
-rw-rw-r-- 1 student student 611 Aug  8 11:04 server.csr
-rw-rw-r-- 1 student student 963 Aug  8 11:01 server.key
student@Ubuntu:/tmp/sslcerts$
```

9. Sign the **server.csr** to create a **server.crt** file. When prompted for the *pass phrase*, type **secret** followed by pressing **Enter**.

```
student@Ubuntu:/tmp/sslcerts$ openssl x509 -req -days 365 -in server.csr
-siginkey server.key -out server.crt
```

```
student@Ubuntu:/tmp/sslcerts$ openssl x509 -req -days 365 -in server.csr -signkey server.key
-out server.crt
Signature ok
subject=/C=US/ST=TX/L=Austin/O=XYZ.corp/CN=XYZ
Getting Private key
Enter pass phrase for server.key:
student@Ubuntu:/tmp/sslcerts$
```

10. Verify that the new **server.crt** has been created.

```
student@Ubuntu:/tmp/sslcerts$ ls -l
```

```
student@Ubuntu:/tmp/sslcerts$ ls -l
total 12
-rw-rw-r-- 1 student student 774 Aug  8 11:07 server.crt
-rw-rw-r-- 1 student student 611 Aug  8 11:04 server.csr
-rw-rw-r-- 1 student student 963 Aug  8 11:01 server.key
student@Ubuntu:/tmp/sslcerts$
```

11. View the contents of the newly created **server.crt** certificate.

```
student@Ubuntu:/tmp/sslcerts$ openssl x509 -in server.crt -noout -text
```

```
student@Ubuntu:/tmp/sslcerts$ openssl x509 -in server.crt -noout -text
Certificate:
Data:
    Version: 1 (0x0)
    Serial Number:
        ce:71:e0:2c:2d:79:6c:fa
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=US, ST=TX, L=Austin, O=XYZ.corp, CN=XYZ
        Validity
            Not Before: Aug  8 15:07:53 2018 GMT
            Not After : Aug  8 15:07:53 2019 GMT
        Subject: C=US, ST=TX, L=Austin, O=XYZ.corp, CN=XYZ
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (1024 bit)
                    Modulus:
                        00:c1:87:68:05:86:b1:8b:cf:e6:eb:d1:9c:a5:b8:
                        65:e4:7c:ee:98:e4:2e:d7:86:0c:0a:23:90:1b:04:
                        ee:75:5d:89:f8:f6:27:4d:ed:36:95:b6:62:c1:95:
                        dc:02:1c:13:75:2a:79:6d:d9:67:19:af:23:02:7f:
                        c7:70:e9:eb:38:c4:6b:56:63:4a:f1:a6:9e:b5:f8:
                        e7:ef:a7:7b:ee:a1:36:6a:4b:18:17:a3:3d:3d:70:
                        0c:92:8b:1a:12:06:5f:48:16:e0:99:26:a5:76:1f:
                        20:c0:0f:53:c9:60:11:a7:24:2c:20:6d:79:d7:8a:
                        3d:07:1c:d1:16:d8:e5:c6:b1
                    Exponent: 65537 (0x10001)
            Signature Algorithm: sha1WithRSAEncryption
                82:74:22:b8:ad:b6:50:c8:e0:a2:52:33:d7:02:74:b1:b4:15:
                de:23:b9:f0:67:f5:b6:74:b9:ed:f4:dc:a4:32:aa:7f:9e:54:
                2d:d7:14:95:4a:6d:62:40:id:7c:17:58:ca:f2:8b:1c:b6:53:
                47:58:e1:f2:6c:55:14:9c:61:b9:08:c0:1e:a0:18:ea:2b:f6:
                02:f3:81:2a:b9:6e:e4:a4:d0:9a:9d:94:8d:10:6a:3d:6a:e8:
                df:8c:02:31:ab:3e:66:ec:56:cb:50:b3:74:e8:ae:e5:6a:f2:
                a1:dc:2d:7c:4b:07:f1:f3:c2:91:1d:f4:d7:9a:9b:d2:2e:6e:
                07:2b
student@Ubuntu:/tmp/sslcerts$
```

2.2 Configure Apache to Utilize SSL

1. Create a new directory that will act as a placeholder for the **SSL** objects. If prompted for a password, enter **securepassword**.

```
student@Ubuntu: /tmp/sslcerts$ sudo mkdir /etc/apache2/ssl_certs
```

```
student@Ubuntu:/tmp/sslcerts$ sudo mkdir /etc/apache2/ssl_certs
[sudo] password for student:
student@Ubuntu:/tmp/sslcerts$
```

2. While in the **/tmp/sslcerts** directory, generate the same **server.key** but with no pass phrase requirement. If prompted for a password, enter **secret**.

```
student@Ubuntu: /tmp/sslcerts$ openssl rsa -in server.key -out server.key.nopass
```

```
student@Ubuntu:/tmp/sslcerts$ openssl rsa -in server.key -out server.key.nopass
Enter pass phrase for server.key:
writing RSA key
student@Ubuntu:/tmp/sslcerts$
```

3. List the current files in the directory. You should now have *four* different files.

```
student@Ubuntu: /tmp/sslcerts$ ls -l
```

```
student@Ubuntu:/tmp/sslcerts$ ls -l
total 16
-rw-rw-r-- 1 student student 774 Aug  8 11:07 server.crt
-rw-rw-r-- 1 student student 611 Aug  8 11:04 server.csr
-rw-rw-r-- 1 student student 963 Aug  8 11:01 server.key
-rw-rw-r-- 1 student student 887 Aug  8 11:13 server.key.nopass
student@Ubuntu:/tmp/sslcerts$
```

4. Copy the **server.key.nopass** to the **/etc/apache2/ssl_certs** directory. If prompted for a password, enter **securepassword**.

```
student@Ubuntu: /tmp/sslcerts$ sudo cp server.key.nopass /etc/apache2/ssl_certs
```

```
student@Ubuntu:/tmp/sslcerts$ sudo cp server.key.nopass /etc/apache2/ssl_certs
student@Ubuntu:/tmp/sslcerts$
```

5. Copy the **server.crt** file to the **/etc/apache2/ssl_certs** directory. If prompted for a password, enter **securepassword**.

```
student@Ubuntu: /tmp/sslcerts$ sudo cp server.crt /etc/apache2/ssl_certs
```

```
student@Ubuntu:/tmp/sslcerts$ sudo cp server.crt /etc/apache2/ssl_certs
student@Ubuntu:/tmp/sslcerts$
```

6. Change to the **/etc/apache2/ssl_certs** directory.

```
student@Ubuntu: /tmp/sslcerts$ cd /etc/apache2/ssl_certs
```

```
student@Ubuntu:/tmp/sslcerts$ cd /etc/apache2/ssl_certs
student@Ubuntu:/etc/apache2/ssl_certs$
```

7. Verify that two files are present in the directory.

```
student@Ubuntu: /etc/apache2/ssl_certs$ ls -l
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ ls -l
total 8
-rw-r--r-- 1 root root 774 Aug  8 11:17 server.crt
-rw-r--r-- 1 root root 887 Aug  8 11:15 server.key.nopass
student@Ubuntu:/etc/apache2/ssl_certs$
```

8. Rename the **server.key.nopass** file to **server.key**. If prompted for a password, enter **securepassword**.

```
student@Ubuntu: /etc/apache2/ssl_certs$ sudo mv server.key.nopass server.key
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo mv server.key.nopass server.key
student@Ubuntu:/etc/apache2/ssl_certs$
```

9. Start the *Apache* web service. If prompted for a password, type **securepassword**.

```
student@Ubuntu: /etc/apache2/ssl_certs$ sudo service apache2 start
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo service apache2 start
 * Starting web server apache2
apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.
0.1 for ServerName
httpd (pid 1407) already running
[ OK ]
student@Ubuntu:/etc/apache2/ssl_certs$
```

10. Initiate the *a2enmod* module for *SSL*. If prompted for a password, type **securepassword**.

```
student@Ubuntu: /etc/apache2/ssl_certs$ sudo a2enmod ssl
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo a2enmod ssl
Enabling module ssl.
See /usr/share/doc/apache2.2-common/README.Debian.gz on how to configure SSL and create self
-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
student@Ubuntu:/etc/apache2/ssl_certs$
```

11. Create a symbolic link. If prompted for a password, type **securepassword**.

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo ln -s /etc/apache2/sites-available/default-ssl /etc/apache2/sites-enabled/000-default-ssl
student@Ubuntu:/etc/apache2/ssl_certs$
```

12. Verify that a *Virtual Host* is configured in the default *sites-available* file. Type the command below to open the file with the **nano** text editor. If prompted for a password, type **securepassword**.

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo nano /etc/apache2/sites-available/default
```

13. When using the *nano* editor, confirm that the **ServerName** is set to **192.168.1.50:80**. Once confirmed, press **CTRL+X** to exit.

```
<VirtualHost *:80>
    ServerName 192.168.1.50:80
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
```

14. Next, edit the contents of the *default-ssl* file. Type the command below followed by pressing **Enter**. If prompted for a password, type **securepassword**.

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo nano /etc/apache2/sites-available/default-ssl
```

15. Add the following line after the *<VirtualHost _default_:443>* line. Use the **arrow keys** to position the cursor.

```
ServerName 192.168.1.50:443
```

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerName 192.168.1.50:443
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www
    <Directory />
        Options FollowSymLinks
        AllowOverride None
```

16. Next change the line that reads *DocumentRoot /var/www* to **DocumentRoot /var/www_ssl**.

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerName 192.168.1.50:443
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www_ssl
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
```

17. Scroll down with the **arrow keys** until you see an entry for *SSLCertificateFile*. Change this, so it reflects the line below.

```
SSLCertificateFile /etc/apache2/ssl_certs/server.crt
```

```
# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
SSLEngine on

# A self-signed (snakeoil) certificate can be created by installing
# the ssl-cert package. See
# /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
# If both key and certificate are stored in the same file, only the
# SSLCertificateFile directive is needed.
SSLCertificateFile /etc/apache2/ssl_certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

# Server Certificate Chain:
# Point SSLCertificateChainFile at a file containing the
# concatenation of PEM encoded CA certificates which form the
# certificate chain for the server certificate. Alternatively
# the referenced file can be the same as SSLCertificateFile
# when the CA certificates are directly appended to the server
# certificate for convinience.
#SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt
```

18. One line down, edit the `SSLCertificateKeyFile` line to match the information below.

```
SSLCertificateKeyFile /etc/apache2/ssl_certs/server.key
```

```
</Directory>

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

#   A self-signed (snakeoil) certificate can be created by installing
#   the ssl-cert package. See
#   /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
#   If both key and certificate are stored in the same file, only the
#   SSLCertificateFile directive is needed.
SSLCertificateFile /etc/apache2/ssl_certs/server.crt
SSLCertificateKeyFile /etc/apache2/ssl_certs/server.key

#   Server Certificate Chain:
#   Point SSLCertificateChainFile at a file containing the
#   concatenation of PEM encoded CA certificates which form the
#   certificate chain for the server certificate. Alternatively
#   the referenced file can be the same as SSLCertificateFile
#   when the CA certificates are directly appended to the server
#   certificate for convinience.
#SSLCertificateChainFile /etc/apache2/ssl_crt/server.ca.crt
```

19. Press **CTRL+X** to exit. When prompted to *Save modified buffer*, type **Y**.

Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
 Y Yes
 N No ^C Cancel

20. When prompted to confirm filename, verify the output is set to `/etc/apache2/sites-available/default-ssl`. Press **Enter**.

File Name to Write: /etc/apache2/sites-available/default-ssl
 ^G Get Help M-D DOS Format M-A Append M-B Backup File
 ^C Cancel M-M Mac Format M-P Prepend

21. Create a new directory.

```
student@Ubuntu: /etc/apache2/ssl_certs$ sudo mkdir /var/www_ssl
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ sudo mkdir /var/www_ssl
student@Ubuntu:/etc/apache2/ssl_certs$
```

22. Leave the *terminal* open for the next section.

2.3 Configuring & Testing HTTPS Test Page

1. While on the *terminal*, navigate to the **/var/www_ssl** directory.

```
student@Ubuntu: /etc/apache2/ssl_certs$ cd /var/www_ssl
```

```
student@Ubuntu:/etc/apache2/ssl_certs$ cd /var/www_ssl
student@Ubuntu:/var/www_ssl$
```

2. Create a new **index.html** file. If prompted for a password, enter **securepassword**.

```
student@Ubuntu: /var/www_ssl $ sudo nano index.html
```

3. Within the *nano* text editor, type *HTML* code below.

```
<html>
<body>
<h1>Testing HTTPS Service</h1>
</body>
</html>
```

```
<html>
<body>
<h1>Testing HTTPS Service</h1>
</body>
</html>
```

4. Press **CTRL+X** to exit and save the file. When prompted to save modified buffer, type **Y**.

Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?

Y Yes

N No ^C Cancel

5. When prompted for the file name, verify that **index.html** is populated and press **Enter**.

File Name to Write: index.html	^G Get Help	M-D DOS Format	M-A Append	M-B Backup File
	^C Cancel	M-M Mac Format	M-P Prepend	

- Restart the *Apache* web service to apply all the configurations changes made. If prompted for a password, enter **securepassword**.

```
student@Ubuntu:/var/www_ssl$ sudo service apache2 restart
```

```
student@Ubuntu:/var/www_ssl$ sudo service apache2 restart
 * Restarting web server apache2
apache2: Could not reliably determine the server's fully qualified domain name, using 127.0
.0.1 for ServerName
... waiting apache2: Could not reliably determine the server's fully qualified domain name
, using 127.0.0.1 for ServerName
 [ OK ]
```

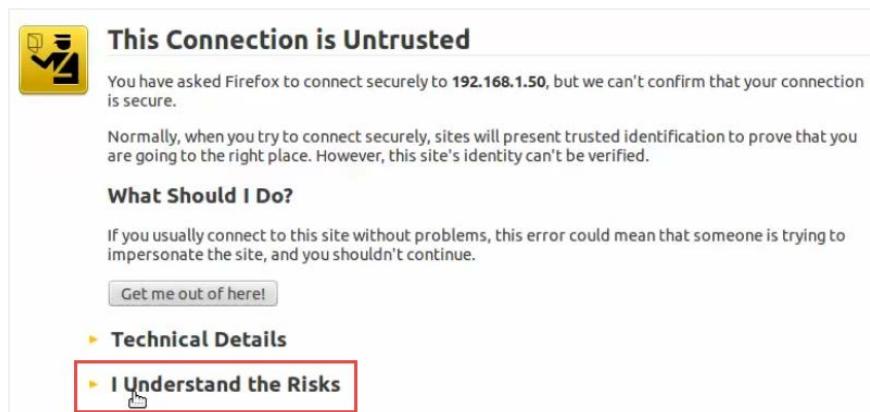
- Open a new *Firefox* web browser by clicking the **Firefox** icon located on the left menu pane.



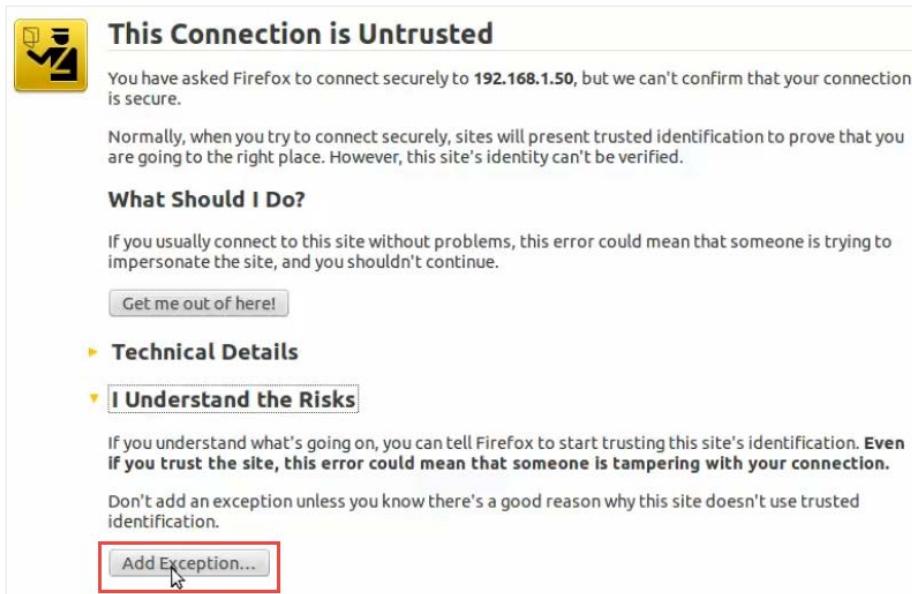
- Within the *address bar*, type **https://192.168.1.50**. Press **Enter**.



- When presented with the *Connection is Untrusted* page, click on **I Understand the Risks**.



10. When expanded, click on the **Add Exception** button.



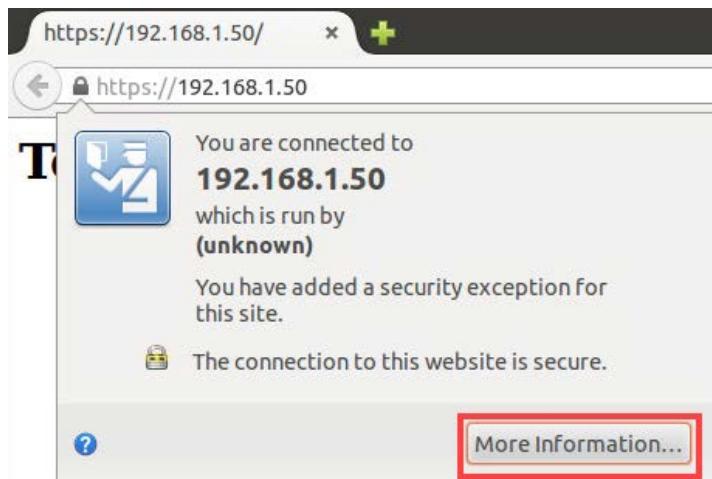
11. A new pop-up window appears, click on **Confirm Security Exception**.



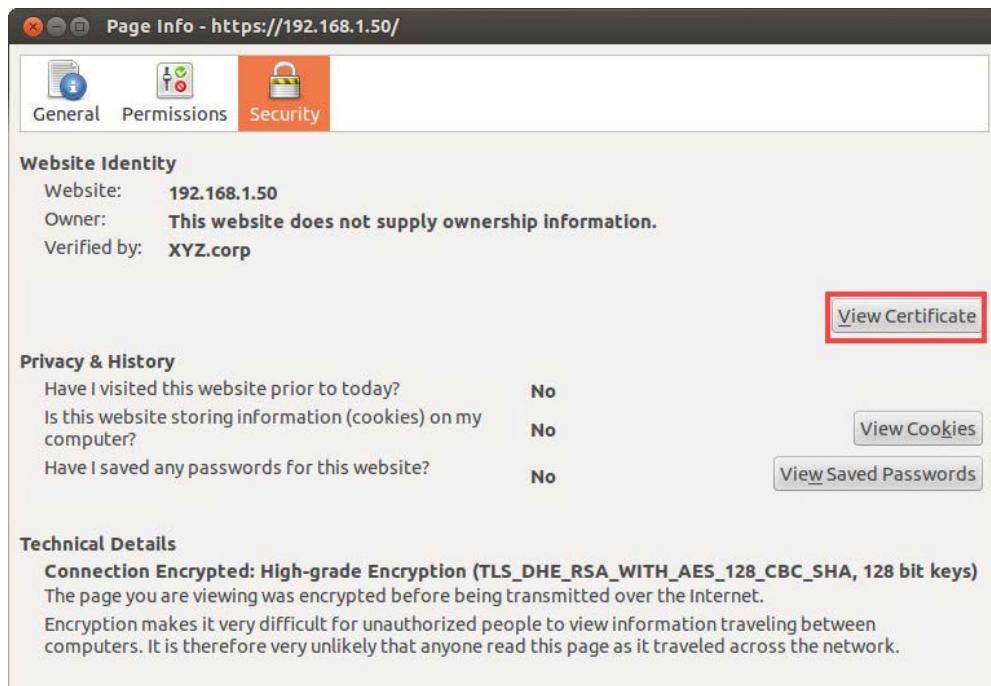
12. Notice the web page headlined with “*Testing HTTPS Service.*” To view the contents of the *server certificate*, click the **lock** icon located to the left of the URL.



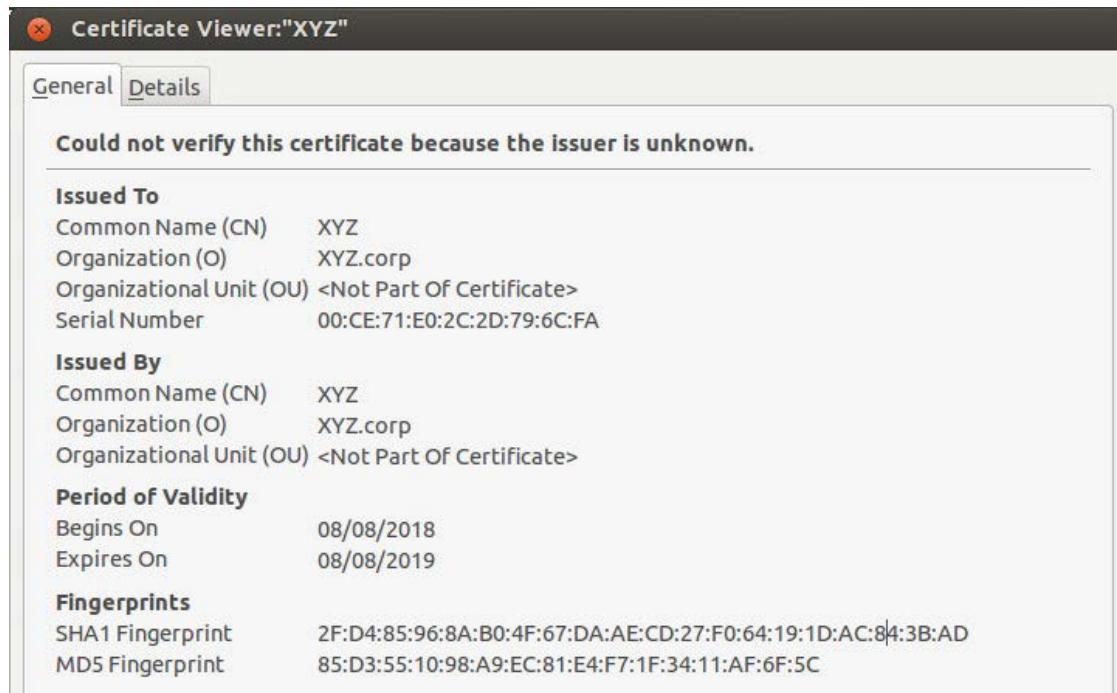
13. A small window will pop-up. Click on the *More Information* button.



14. On the *Page info* screen, notice the *Website Identity* information. Click the **View Certificate** button.



15. On the *Certificate Viewer* window, notice the entries for *Issued To* and *Issued By* as well as the *Period of Validity*. All values are reflective upon the contingencies set when self-sign the certificate took place at the beginning of *Task 2.1*.



16. The lab is now complete; you may end the reservation.