Deploy Server Certificate

Once created, the server certificate (.crt) and the key used to create it (.key) need to be deployed to the server which contains the service(s) you want to secure.

Each operating system has a default, and even different distributions of the same operating systems may use a different default. However, since we must manually configure each services, we can choose a common location for our certificates if we want. On Unix, I create a directory named /etc/certificates and put all of my local (LAN) certificates in there.

Which files to copy

The files that need to be copied are the certificate (.crt) and privte key (.key) files. In some cases, it is useful to combine them into one file, which the .pem format is excellent for (PEM is designed to hold multiple types of information). To create the .pem equivilent, simply cat the key and crt files and send that output to a .pem. You can do it with a text editor, or with the Unix command:

```
cat servername.crt servername.key > servername.pem
```

Copying the files

These are simple text files, so I use scp.

```
cat servername.crt servername.key > servername.pem
ssh target_server 'mkdir -p /etc/certificates'
scp servername.crt servername.key servername.pem
target_server:/etc/certificates
ssh target_server 'chmod 755 /etc/certificates && chmod 644
/etc/certificates/* && chown -fR root:root /etc/certificates'
```

This will ensure /etc/certificates exists (mkdir -p), copy the files to it, and set ownership and permissions.

Setting Service to use it

For now, I'm only going to do the Apache server on a Devuan (Debian) Linux server. Everyone and every program does it differently.

Apache

We need to edit the default SSL configuration file and enable it. Note that these instructions assume a

base Devuan server with no modifications made to the existing configuration files. You will need to adapt if you have already changed things.

On Debian based servers, the default Apache2 ssl site configuration is located in /etc/apache2/sites-available/default-ssl.conf. Use your favorite editor to change that.

```
cd /etc/apache2/sites-available
cp default-ssl.conf default-ssl.conf.bak # make a backup of the conf file
edit default-ssl.conf
```

Look for two directives, **SSLCertificateFile** and **SSLCertificateKeyFile**, and edit them to point to /etc/certificates/server.crt and /etc/certificates/server.key. They are on line 32 in my current setup. So, you will see:

```
                SSLCertificateFile      /etc/certificates/servername.crt
                SSLCertificateKeyFile /etc/certificates/servername.key
```

Obviously, change servername to the actual name used for your keys. Save the file

Now, assuming you have not been using SSL before, we need to tell Apache to begin using SSL, and to use default-ssl.conf.

```
a2enmod ssl
a2ensite default-ssl.conf
service apache2 reload
```

If the final command (the service reload) shows no error, you are good. You can now go to one of the workstations that has the CA installed and open the web browser using https. If it produced an error, you will need to fix it, obviously.

Note: With Apache, it is very simple to force SSL (https) on a connection using mod_rewrite. After you know https is working, you can go back and do the following:

```
a2enmod mod_rewrite
edit /etc/apache2/sites-available/000-default.conf
```

Now, add the following lines between **<VirtualHost *:80>** and **</VirtualHost>**. Actual location does not matter, but not within other blocks. I tend to put it just before the </VirtualHost> line.

```
    RewriteEngine On
    RewriteCond %{HTTPS} off
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
```

Reload the apache server

```
service apache2 reload
```

and traffic going to the server on http will be redirected to https.

Future Deployments

Once you have the services set up on a server, you can run the same commands from *Copying the files*, then afterwards, reload the service(s). Following is a modified set of commands to deploy a new certificate which also reloads the Apache web service, telling it to reread the SSL certs you have just put out there.

```
cat servername.crt servername.key > servername.pem
ssh target_server 'mkdir -p /etc/certificates'
scp servername.crt servername.key servername.pem
target_server:/etc/certificates
ssh target_server 'chmod 755 /etc/certificates && chmod 644
/etc/certificates/* && chown -fR root:root /etc/certificates'
ssh target_server 'service apache2 reload'
```