Obtaining certificates

TLS (https://en.wikipedia.org/wiki/Transport_Layer_Security) is the preferred method of encrypting and protecting HTTP sessions between a client and an HTTP server. Flow Enterprise Server requires TLS/SSL be used for user sessions and connectivity to your source code repositories and ticketing systems. This is an essential part of our security commitment.

When setting up Flow Enterprise Server, there are three ways to obtain and manage your TLS/SSL certificates.

- **Purchased certificates**: The most common method to obtain a TLS/SSL certificate is to purchase it from a public certificate authority like Verisign or DigiCert. These certificates work in most browsers without problems.
- **Use a company-wide certificate authority**: For large companies that use many TLS/SSL certificates, it is more cost effective to create their own certificate authority and self-sign certificates.
- **Create your own**: This is the least expensive option, but requires your users to accept the certificate before their browser can recognize it.

In all of these cases, you can create certificates in three ways to meet your needs:

1. Specifically for the DNS of your server. For example, a certificate with a **Common Name** of **flow.mycompany.com** would only work for a server that receives requests for **flow.mycompany.com**.
2. Using **Subject Alternative Names** is similar to the first option with a specific **Common Name**, but with added **Subject Alternative Names** attached. For example, you could create the certificate with a **Common Name** of **flow.mycompany.com** and add **Subject Alternative Names** like **flow-analytics.mycompany.com** and **developer-info.mycompany.com** and it would apply to any of those names.
3. Wildcard certificates allow you to create a single certificate for an entire domain. For example, using ***.mycompany.com** as the **Common Name** of the wildcard certificate would allow the certificate to be used for any DNS name that ended with **mycompany.com**.

Below describes how to configure Flow Enterprise for each of the three scenarios.

Purchased Certificates

Purchasing certificates is the most common method of getting TLS/SSL certificates for a server and application. This option provides your server with a certificate that is guaranteed to be recognized as secure by most browsers and operating systems. Purchased certificates can be configured for both the specific server hostname, multiple hostnames, or in a wildcard fashion for an entire domain.

To purchase certificates, you should pick a well-respected certificate authority that sells certificates. There are many available and you can find many of them in this list (https://en.wikipedia.org/wiki/Certificate_authority#Providers). Once you have picked a provider, you should follow their instructions for requesting and purchasing a certificate.

Once that process is complete, you will have three or four files that are important:

- The private key. It is most likely that you created this when you created a certificate request to the authority.
- A certificate request file, sometimes called the CSR.
- The certificate itself.
- An intermediate certificate chain file. This file is only provided by some authorities if it is required. If it is

provided, you will need it.

Using a Company Wide Certificate Authority

Many companies that grow to a large IT infrastructure, find the cost of purchasing certificates to be cost prohibitive. When that happens, they often establish their own internal certificate authority and begin signing and distributing their own certificates for internal and sometimes external use. Once in place, these organizations insure that their root certificate is trusted on all of the employee and infrastructure systems so that, as far as employees and servers are concerned, the certificates are 100% valid and trusted.

If your company has gone this route, you will need to request certificates through your company's established processes. This process may be as simple as a web application where you can request a certificate and download it, or it may be part of a task-tracking software like JIRA.

After obtaining the certificate, you should have, at a minimum:

- The private key for the certificate.
- The certificate itself.

You may also receive:

- The certificate request file, or CSR
- An intermediate certificate chain file that lets users find an appropriate trusted root when they access the servers using the certificate.

Signing Your Own Certificates

Signing your own certificates is relatively easy, however it has several downsides:

- It won't be automatically trusted by your users' browsers
- It won't be automatically trusted by other systems that contact the server
- It cannot be verified for authenticity in any way

This method is obviously inexpensive, but it has serious security and maintenance costs. If you choose to use this method, you can sign your own certificates using the following command on a server or workstation that has a modern version of OpenSSL on it.

```
openssl req -x509 -nodes -days -newkey rsa:2048 -keyout -out
```

This command takes several options and you should replace values in angle brackets with the values that best suit your needs. Here is an explanation:

- `req`: This tells OpenSSL that you want to make a new certificate request
- `-x509`: This option tells OpenSSL to use the x509 certificate format. This is the primary certificate format for all web certificates.
- `-nodes`: This tells OpenSSL that you do not want to use DES encryption on the private key. If you omit this option, you will be required to enter a password for the private key that is created. You will also be required to enter that password every time you restart a server that uses this private key.
- `-days`: This option specifies how many days the certificate should be valid for. For example, "-days 365"

would create a certificate valid for one year.

- `-newkey`: This option tells OpenSSL to create a new RSA private key. The option value is in the format of `:`. In the above example, it would create an RSA key that is 2048 bits of encryption strength.
- `-keyout`: This option places the generated private key into the file specified. For example `/apache2/certs/myserverkey.pem`.
- `-out`: This tells OpenSSL where to place the generated certificate.

When you execute the command, it is going to ask you several questions. You should enter values for all of them, but the one that matters the most is the **Common Name** field. You must enter a value here that matches the DNS name of the website you want to create the certificate for. For example, if you are configuring https://flow.mycompany.com, you would enter **flow.mycompany.com** when prompted for the **Common Name**. Details about the questions are:

- Country Name: This is a 2-letter country code. For example, US for United States.
- State or Province Name: The full name of your state or province. For example, Colorado.
- Locality Name: This is the name of your City or some other location. For example, Denver.
- Organization Name: This can be your company name.
- Organizational Unit: This can be your department or group name inside the company. For example, Software Engineering.
- Common Name: This is the fully qualified domain name of your website. For example, for https://flow.mycompany.com, you'd enter flow.mycompany.com.
- E-mail Address: The address that users can reach for information.

When the command finishes, you should have a file at the path you specified in the `-out` parameter. Verify the validity of this certificate by entering the following command:

```
openssl x509 -noout -text -in
```

This command prints the information about your certificate. Verify that the data matches what you expect.

back to top

If you need help, please email support@pluralsight.com () for 24/7 assistance.