SecuritySpy

Setting Up SecuritySpy Over SSL

Secure Sockets Layer (SSL) is a cryptographic protocol that provides secure communications on the internet. It uses two keys to encrypt data: a public key and a private key. URLs that require an SSL connection start with *https://* insead of *http://* and operate on port 443 by default. SSL increases security as it makes it impossible for someone intercepting the stream of data to decode any information from it.

SecuritySpy does not have built-in support for SSL, however Mac OS X comes with Apache, a fully-featured and powerful web server, that can be used to set up the secure communication between the internet and SecuritySpy. In this way, Apache will be acting as a secure "reverse proxy" web server for SecuritySpy. This document describes how to set this up.

These instructions describe how to set up SSL using "self-signed" certificates. This allows you to get everything up and running, however for a proper installation you should ideally obtain a certificate from a Certificate Authority (such as Verisign or Thawte). A Certificate Authority is a trusted entity that confirms to whomever is connecting to your web server that you are who you say you are. This is only really applicable for web servers available to the general public, so a self-signed certificate is appropriate for when the server will be accessed by you or your employees or agents, as in this case there is no doubt about the server's authenticity.

Although we have made every effort to make this guide easy to follow, this is a complex setup that requires use of the Terminal and editing of Apache configuration files, so is not for the novice user. This guide assumes that you have already set up the web server feature of SecuritySpy and are familiar with concepts such as dynamic DNS, port forwarding, and IP addressing on local networks.

Step 1: Create a Certificate Authority (CA) certificate

Open Terminal (from your Applications folder). Type the following commands, shown in red:

mkdir ~/Documents/myssl

This creates a *myssl* folder where you will be creating the required key and certificate files.

cd ~/Documents/myssl

This sets the *myssl* folder as the current working directory

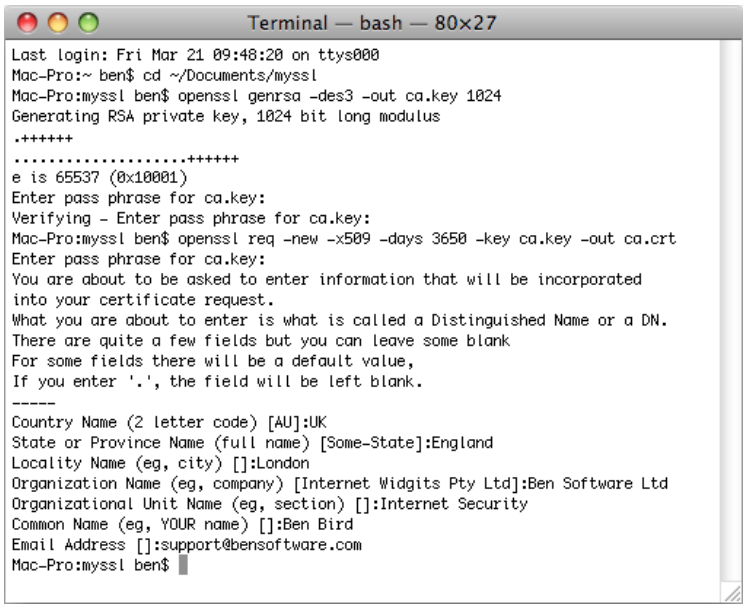
openssl genrsa -des3 -out ca.key 1024

This creates a triple-DES encrypted 1024-bit RSA key file called *ca.key*. You will be asked for a passphrase for the key - make a note of it.

openssl req -new -x509 -days 3650 -key ca.key -out ca.crt

This creates a Certificate Authority certificate. You will be asked for the passphrase for the key you just created, as well as some information about yourself (i.e. you as the Certificate Authority). Remember that this is for the certificate that will, for the moment at least, be a substitue for a proper one obtained from real Certificate Authority.

The Terminal output for this step should look something like this:



Step 2: Generate a private key for the web server

In Terminal, type:

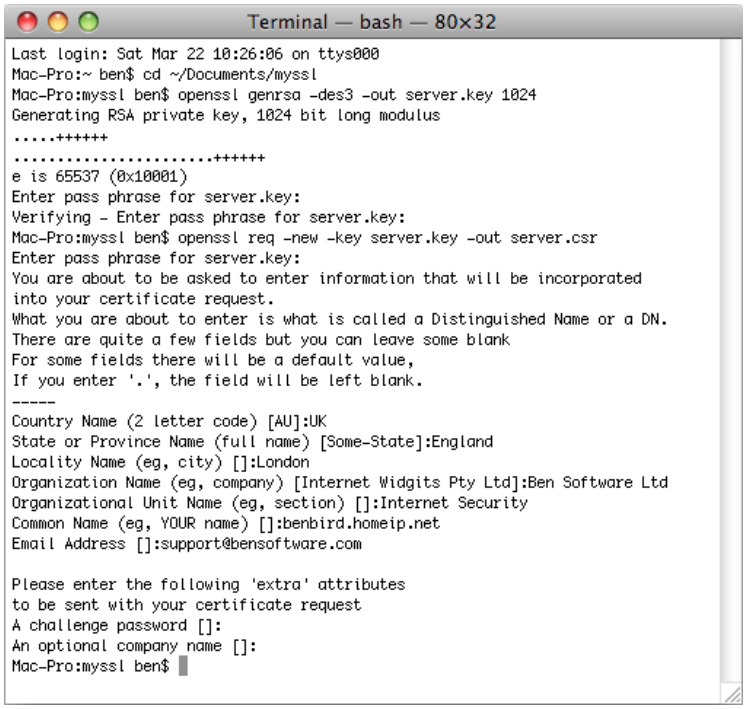
openssl genrsa -des3 -out server.key 1024

This creates a triple-DES encrypted 1024-bit RSA key file called "server.key". You will be asked for a passphrase for the key - make a note of it.

openssl req -new -key server.key -out server.csr

You will be asked for the passphrase for the key you just created, as well as some information about yourself (i.e. you as the web server administrator). The vital thing here is, when it askes you to enter the *Common Name*, you must enter the host name (or IP address) of your server as a client would connect to it over the internet. For example, if you have a dynamic DNS name set up for accessing you server over the internet called *myserver.dyndns.org*, this is what you enter here. It may also ask you for a *challenge password* and *optional company name* - just leave these blank.

The Terminal output for this step should look something like this:



Step 3: Sign the server key with your CA certificate

You need to download the latest mod_ssl, find the *sign.sh* file from the *pkg.contrib* folder, and place it in your ~/Documents/myssl folder.

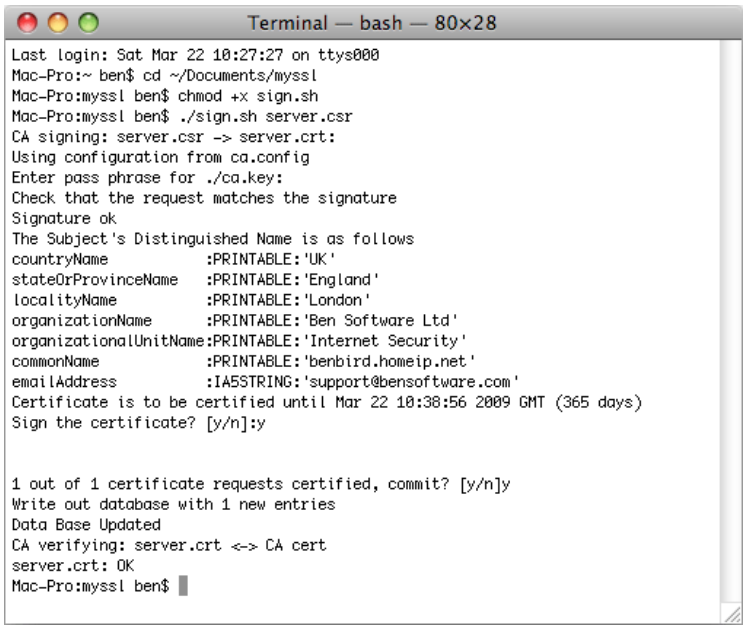
Back in the Terminal, type:

chmod +x sign.sh

./sign.sh server.csr

You will be asked for the passphrase for the ca.key file that you created above. For the questions *Sign the certificate?* and *commit?*, type y and enter. This creates the server certificate *server.crt* file.

Here is the Terminal output for this step:



Step 4: Remove the passphrase requirement from server.key

This step is required so that Apache can read the private key without you having to manually start Apache from the command line with the passphrase every time you need to enable the web server.

In Terminal, type:

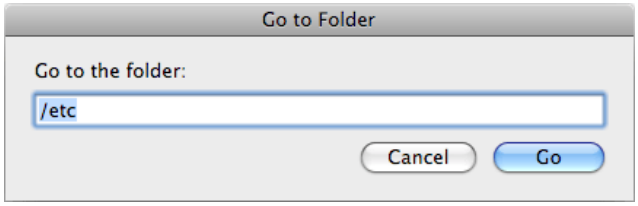
cp server.key server.key.original

openssl rsa -in server.key.original -out server.key

You will be asked for the passphrase that you initially specified for the server.key file.

Step 5: Configure the Apache web server for SSL in Mac OS X 10.4 "Tiger" (see next step for alternative Mac OS X 10.5 / 10.6 instructions)

Now we have all the files needed to enable SSL in Apache. In the Finder, choose *Go to folder* from the *Go* menu and type */etc*:



Within this folder, find the *httpd* folder - this is where the configuration files from the web server are stored. Click on the *httpd* folder and Get Info (from the File menu or Command-I on the keyboard). You need to temporarily change the permmissions of this folder so that you can make changes - click the padlock to unlock it (if it's not already open), and change the owner to the admin account of the computer (in the below example the admin account is called *Server*):

Open the *httpd* folder, locate the *users* folder, and also set the permissions for the users folder as above. It is a good idea to reset these permissions back to their previous values (owned by the system) when you have finished configuring the web server.

Within the *httpd* folder, create a new folder called *myssl*, and copy the *server.crt* and *server.key* files from ~/Documents/myssl into this folder.

Open the *httpd.conf* file in TextEdit. This is the main configuration file for Apache. Make a backup of this file before making any changes (call it *httpd.conf.backup* for example). These are the changes you need to make to this file:

- Uncomment (remove the # from the front of) the line *LoadModule ssl_module*
- Uncomment the line *AddModule mod_ssl.c*

- Uncomment the line *LoadModule proxy_module*
- Uncomment the line *AddModule mod_proxy.c*
- Save the file.

Locate the file *Server.conf* in the *users* folder. Duplicate this file and rename the duplicate *myssl.conf*. Open *myssl.conf* in TextEdit and follow these steps:

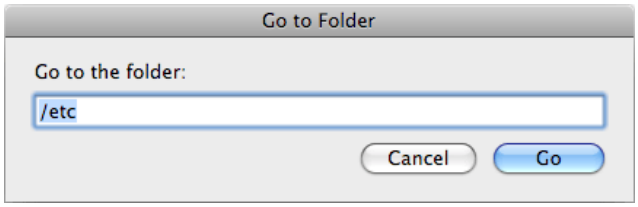
- Delete all existing text in this file.
- Copy and paste the following text into the file:

```
<IfModule mod_ssl.c>
Listen 443
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
< VirtualHost _default_:443>
SSLEngine on
ServerName SERVERNAME
ServerAdmin ADMINEMAIL
ErrorLog /var/log/httpd/error_log
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile "/private/etc/httpd/myssl/server.crt"
SSLCertificateKeyFile "/private/etc/httpd/myssl/server.key"
< /VirtualHost>
</IfModule>
```

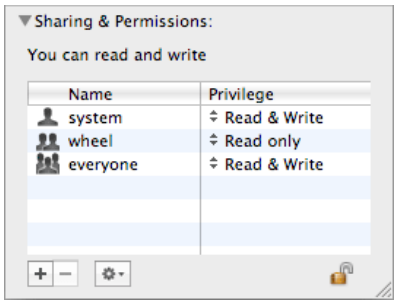
- Instead of SERVERNAME and ADMINEMAIL, enter the correct server external host name (or IP address) and email address of the server administrator.
- Save the file.

Step 5: Configure the Apache web server for SSL in Mac OS X 10.5 "Leopard" or Mac OS X 10.6 "Snow Leopard" (see previous step for alternative Mac OS X 10.4 instructions)

Now we have all the files needed to enable SSL in Apache. In the Finder, choose *Go to folder* from the *Go* menu and type */etc*:



Within this folder, find the *apache2* folder – this is where the configuration files from the web server are stored. Click on the *apache2* folder and Get Info (from the File menu or Command-I on the keyboard). You need to temporarily change the permmissions of this folder so that you can make changes – click the padlock at the bottom of the Info window to unlock it, enter the computer's admin password, and change the permission for *everyone* to *Read & Write*:



Open the *apache2* folder, locate the *extra* folder, and also set the permissions for the *extra* folder as above. It is a good idea to reset these permissions back to *Read only* when have finished editing the configuration files.

Within the *apache2* folder, create a new folder called *myssl*, and copy the *server.crt* and *server.key* files from ~/Documents/myssl/ into this folder.

Open the *httpd.conf* file in TextEdit. This is the main configuration file for Apache. Make a backup of this file before making any changes (call it *httpd.conf.backup* for example). These are the changes you need to make to this file:

- Locate the *ServerName* parameter and set it to the external host name (or IP address) of your server. Remove any # character from the start of the line.
- Locate the line *Include /private/etc/apache2/extra/httpd-ssl.conf* and remove any # character at the start of this line.
- Save the file.

Open the *httpd-ssl.conf* file within the *extra* folder. Make a backup of this file before making any changes (call it *httpd-ssl.conf.backup* for example). Make the following changes:

- Locate the *ServerName* parameter and set it to the external host name (or IP address) of your server. Remove any # character from the start of the line.
- Locate the *SSLCertificateFile* parameter and set it to *"/private/etc/apache2/myssl/server.crt"* (with quotes). Remove any # character from the start of this line.
- Locate the *SSLCertificateKeyFile* parameter and set it to *"/private/etc/apache2/myssl/server.key"* (with quotes). Remove any # character from the start of this line.
- Save the file.

Step 6: Test Apache over SSL

Now you are ready to test the Apache web server over SSL. Go to System Preferences, click on Sharing, and enable Web Sharing ("Personal Web Sharing" on Mac OS X 10.4). This starts the Apache web server. If web sharing was already enabled, you need to disable it and then enable it again to restart the web server. Open Safari and enter:

https://127.0.0.1/

Note that this is *https* and not *http*. The IP address 127.0.0.1 is what is called the "loopback" address that refers to "this computer". You should get a warning about the certificate being invalid because of a host name mismatch. This is because the address that you are using to access the server (127.0.0.1) is different from the address that you specified in the server certificate. This error will not occur when accessing the web server from the internet using the proper host name (although the client will get a warning that the certificate was signed by an unknown authority). Simply click the Continue button to ignore the warning and enter the secure site. You should see the default Apache web page with a padlock in the top right of the Safari window, indicating a secure connection.

Step 7: Configure the Apache web server as a reverse proxy for SecuritySpy

Follow these steps:

- Open the *myssl.conf* file (Mac OS X 10.4) or *httpd-ssl.conf* file (Mac OS X 10.5 / 10.6) in TextEdit
- Scroll to the bottom of the file, and just above the "</VirtualHost>" tag, add these two lines:

  RewriteEngine On
  RewriteRule ^/(.*) http://127.0.0.1:8000/$1 [P]

  SecuritySpy uses port 8000 by default, but if you are using a different port you can enter it insead of 8000 in the above line. This line tells Apache to act as a reverse proxy server for SecuritySpy for all incoming SSL connections.
- Save the file.
- Stop and then start the web server from the Sharing System Preference.
- Open SecuritySpy.

Now back in Safari, go to https://127.0.0.1/ again (you might have to click the reload button because Safari may have cached the previous Apache page). You should now see the SecuritySpy web server over a secure connection.

Note that Apache is still also running on port 80 (the default port for HTTP). If you want to disable this, locate the line *Listen 80* in the *httpd.conf* file and comment it out by adding a # character at the beginning of the line. Alternatively, if you want to also use Apache as a reverse proxy server for normal HTTP connections on port 80, you can add the above two Rewrite lines to the bottom of the *httpd.conf* file. Mac OS X 10.4 only: to enable listening on port 80, add a *Listen 80* line below the *Listen 443* line in the *myssl.conf* file.

Step 8: Set up port forwarding

The final step is to set up port forwarding in your router to the server computer on port 443 - this is the port used by HTTPS.