TinySSL Server

TinySSL is a Secure Sockets Layer (SSL v2/v3) and Transport Layer Security Web Server Daemon based on TinyWeb.

TinySSL versions up to 1.7 were using SSLeay library which is a free implementation of Netscape's Secure Socket Layer written by Eric Young.

TinySSL versions 1.8 and later are using OpenSSL which is a successor of SSLeay.

You have to get the OpenSSL Windows binaries and put the `libssl32.dll` and `libeay32.dll` files to the same directory with tinyssl.exe.

Installing a Secure Socket Layer Server using TinySSL

In order to initiatine a SSL-connection, the secure server must have a certificate or, in other words, Digital ID. The certificates are issued by Certificate authorities. Although the client can also have a certificate, TinySSL does not currently provide client-certificate verifications or authentication of clients based on certificates.

First of all, you must generate your secure server's private key. We will give examples on how to generate a private key that uses the RSA algorithm. For that, feed `.rnd` file with lots of interesting and varied data, that would be used for key generation. `.rnd` is unformatted file, size don't care. You can copy a wav-file with digitized noize to it, or just a text-file with randomly-typed words and phrases.

To generate a key, type:

```
openssl genrsa -rand .rnd -out key.pem
1024
```

This command sequence will generate a 1024-bit RSA private key and store it in the file `key.pem`. The key should look like:

```
-----BEGIN RSA PRIVATE KEY-----
MIIBOwIBAAJBALtv55QyzG6i2PlwZ1pah7++Gv8L5j6Hnyr/uTZE1NLG0ABDDexm
q/R4KedLjFEIYjocDui+IXs62NNtXrT8odkCAwEAAQJAbwXq0vJ/+uyEvsNgxLko
nWmM1KvqnAo5uQIhALqEADu5U1Wvt8UN8UDGBRPQulHWNycuNV45d3nnskWPAiAw
ueTyr6WsZ5+SD8g/Hy3xuvF3nPmJRH+rwvVihlcFOg==
-----END RSA PRIVATE KEY-----
```

Remember, that your secure server certificate (Digital ID) will be useless without the key.

Obtaining a certificate (Digital ID) for secure server

Then you should generate your Certificate Signing Request (CSR). The CSR is what contains the name information for the certificate (Country, State/Province, City, Organization, Division, Web Server Domain Name, etc). It also contains your public key. The formats of certificate and CSR used by TinySSL are the same as for Apache or for OpenSSL. CSR should be sent for verification to Certificate Authority (CA) e.g. to Thawte (www.thawte.com). Thawte is issuing certificates for TinySSL without any problem . Choose TinySSL when you will be prompted for Web Server Software. If there will be no OpenSSL in the list, just select Apache or OpenSSL. After verification you will probably receive the certificate.

To generate your CSR, run:

```
openssl req -new -key key.pem -out req.pem -config
openssl.cnf
```

This command sequence will prompt you for the attributes of your certificate. Remember to give the secure server domain name when you would be prompted for "Common Name".

The request (saved to `req.pem` file) should look like:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBGzCBxgIBADBjMQswCQYDVQQGEwJBVTETMBEGA1UECBMKUXVlZW5zbGFuZDEa
MBgGA1UEChMRQ3J5cHRTb2Z0IFB0eSBMdGQxIzAhBgNVBAMTGkNsaWVudCB0ZXN0
2NNtXrT8odkCAwEAATANBgkqhkiG9w0BAQQFAANBAC5JBTeji7RosqMaUIDzIW13
oO6+kPhx9fXSpMFHIsY3aH92Milkov/2A4SuZTcnv/P6+8klmS0EaiUKcRzak4E=
-----END CERTIFICATE REQUEST-----
```

You will now have a private key in `key.pem` and a CSR in `req.pem`

Make sure to store `key.pem` in a safe place. You will need the key to operate your secure server when CA issue your certificate. Note, that it is very important to backup the private key that corresponds to the certificate you purchased. Without the private key the certificate is quite useless. For good security reasons the most of CA's are unable to reissue certificates arbitrarily if you cannot access your private key!

Then send `req.pem` to CA.

Upon reception of a signed certificate from CA, put it to cert.pem.

The certificate should look like:

```
-----BEGIN CERTIFICATE-----
MIICLjCCAZcCAQEwDQYJKoZIhvcNAQEEBQAwWzELMAkGA1UEBhMCQVUxEzARBgNV
BAgTClF1ZWVuc2xhbmQxGjAYBgNVBAoTEUNyeXB0U29mdCBQdHkgTHRkMRswGQYD
dp7jnmWZwKZ9cXsNUS2o4OL07qOk2HOywC0YsNZQsOBu1CBTYYkIefDiKFL1zQHh
8lwwNd4NP+OE3NzUNkCfh4DnFfg9WHkXUlD5UpxNRJ4gJA==
-----END CERTIFICATE-----
```

You can also generate a temporary but ready to use untrusted test certificate by running:

```
openssl req -new -key key.pem -out cert.pem -x509 -config
openssl.cnf
```

Access Authentication

TinySSL supports Basic Access Authentication (rfc-2068), which is configured in `realms.cfg` file. There are MD5/DES-hybrid hashes (also may be called digests) that allow avoiding cleartext reusable passwords to be stored in `realms.cfg` file. Each line of the file describes a single realm and has the following format:

```
ListOfURLs RealmName User1 User2 User3 User4
....
```

`ListOfURLs` is a list of URLs (pipe-seperated) belonging to specified realm, `RealmName` is name of the realm

as per rfc-2068 and `UserN` is user name and hash of a password. To produce a hash, run `str2key.exe` utility, passing password as a command line parameter (no space characters are allowed). As you see, you may assign several users and URLs to an realm. If you do not need access authentication, simply leave `realms.cfg` empty (but do not even think to delete it). The sample file with two realms looks like this:

```
/cgi-bin/*|/view.html|/edit.html Operations mickey|7a4064683b98bf5e
/photos.html Photos ronnie|4f1fab620816ea8a
coolman|f1578aa107bc4aef
```

Here user `mickey` will have access to `Operations` realm and will be able to retrieve `/cgi-bin/*`, `/view.html` and `/edit.html`; users `ronnie` and `coolman` will have access to `Photos` realm with `/photos.html`.

`str2key.exe` utility produces a hash in the following steps: applies MD5 algorinthm to a password string; resulting 128 bits are split on two 64-bit blocks, 56 bits from one block is used as a DES key to ECB-encrypt 64 bits of another block; 64 bits produced by DES ecnryption are taken as hash.

Changing of `realms.cfg` without restarting server is allowed. TinySSL will reload the file if it was modified since last load.

Starting secure server

Before starting TinySSL, make sure `key.pem`, `cert.pem`, `.rnd` and `realms.cfg` files are in the same directory with `TinySSL.exe`.

Run `TinySSL.exe` with the same parameters as needed for `tiny.exe`.