Subject: Any good reason why SSL support is hardcoded to SSLv3 usage?
Posted by steffen on Fri, 24 Oct 2014 15:57:24 GMT
View Forum Message <> Reply to Message

Hi,

I just made little test with HttpRequest and found that I could not connect to any of our secure servers.
SSLv3 was deprecated 15 years ago, and is now proven vulnerable to the poodle attack.

TcpSocket::SSLImp::Start() is hardcoded to use SSLv3:

```
 if(!context.Create(socket.mode == CONNECT ? const_cast<SSL_METHOD
*>(SSLv3_client_method())
                                    : const_cast<SSL_METHOD *>(SSLv3_server_method()))) {
  SetSSLError("Start: SSL context.");
  return false;
 }
```

Changing the methods to TLSv1_2 allowed my little test to connect with our servers:

```
 if(!context.Create(socket.mode == CONNECT ? const_cast<SSL_METHOD
*>(TLSv1_2_client_method())
                                    : const_cast<SSL_METHOD *>(TLSv1_2_server_method()))) {
  SetSSLError("Start: SSL context.");
  return false;
 }
```

In my case it works with TLSv1, TLSv1_1 and TLSv1_2.
Are there any plans on making the protocol selectable or not?
I can try to make a solution, adding a member variable like "SSL_METHOD *sslMethod;" and use it if it is not null otherwise fallback to current code.

There was another report on a similar request earlier this year:
http:// www.ultimatepp.org/forums/index.php?t=msg&th=8408&go to=42367&#msg_42367

Subject: Re: Any good reason why SSL support is hardcoded to SSLv3 usage?
Posted by mirek on Wed, 29 Oct 2014 08:57:08 GMT
View Forum Message <> Reply to Message

steffen wrote on Fri, 24 October 2014 17:57Hi,

I just made little test with HttpRequest and found that I could not connect to any of our secure servers.
SSLv3 was deprecated 15 years ago, and is now proven vulnerable to the poodle attack.

TcpSocket::SSLImp::Start() is hardcoded to use SSLv3:

```
 if(!context.Create(socket.mode == CONNECT ? const_cast<SSL_METHOD
*>(SSLv3_client_method())
                                   : const_cast<SSL_METHOD *>(SSLv3_server_method()))) {
  SetSSLError("Start: SSL context.");
  return false;
 }
```

Changing the methods to TLSv1_2 allowed my little test to connect with our servers:

```
 if(!context.Create(socket.mode == CONNECT ? const_cast<SSL_METHOD
*>(TLSv1_2_client_method())
                                   : const_cast<SSL_METHOD *>(TLSv1_2_server_method()))) {
  SetSSLError("Start: SSL context.");
  return false;
 }
```

In my case it works with TLSv1, TLSv1_1 and TLSv1_2.
Are there any plans on making the protocol selectable or not?
I can try to make a solution, adding a member variable like "SSL_METHOD *sslMethod;" and use it if it is not null otherwise fallback to current code.

There was another report on a similar request earlier this year:
http:// www.ultimatepp.org/forums/index.php?t=msg&th=8408&go to=42367&#msg_42367

Interestingly, I have hit the same issue too 14 days ago, so we are now using SSLv23_client_method/SSLv23_server_method, which according to docs should cover TLS methods as well and downgrade if not available.

Mirek