Subject: Re: SSL server crash
Posted by Zbych on Thu, 20 Oct 2011 09:34:44 GMT
View Forum Message <> Reply to Message

Thank you Mirek for your review, but since I do not plan to remove clients from other threads I will stick with my version.

I also have a few suggestions regarding Web/SSL package, documentation:

1. WebSSL uses blocking IO while making a new connection, so if a server does not respond to a handshake, ssl client hangs.

2. SSL Server also uses blocking IO and waits for the handshake just after a new client has connected. That means that no new connection is accepted until handshake is done. One can easily make DOS attack on Web/SSL server using plain old telnet client.
I think that SSL handshake should be separated from connection acceptance.

3. SSL_shutdown sends close_notify to the other side, so when it returns 0 it should be called again.

4. When application sends close notify and there is something wrong with the connection, it can receive SIGPIPE. Application should install SIGPIPE handler.

So far to workaround blocking IO, I uncommented timeout functions in socket.cpp and socket.h:

```
  void          WriteTimeout(int msecs);
  void          ReadTimeout(int msecs);
```

```
void Socket::Data::WriteTimeout(int msecs)
{
 ASSERT(IsOpen());
 if(IsNull(msecs)) msecs = 0;
#if defined(PLATFORM_WIN32)
 if(setsockopt(socket, SOL_SOCKET, SO_SNDTIMEO, (const char *)&msecs, sizeof(msecs))) {
  SetSockError("setsockopt(SO_SNDTIMEO)");
 }
#elif defined(PLATFORM_POSIX)
 struct timeval tv;
 tv.tv_sec = msecs / 1000;
 tv.tv_usec = (msecs % 1000) * 1000;
 if(setsockopt(socket, SOL_SOCKET, SO_SNDTIMEO, &tv, sizeof(tv)) < 0) {
  SetSockError("setsockopt(SO_SNDTIMEO)");
```

```
 }
#endif
}



void Socket::Data::ReadTimeout(int msecs)
{
 ASSERT(IsOpen());
 if(IsNull(msecs)) msecs = 0;
#if defined(PLATFORM_WIN32)
 if(setsockopt(socket, SOL_SOCKET, SO_RCVTIMEO, (const char *)&msecs, sizeof(msecs))) {
  SetSockError("setsockopt(SO_RCVTIMEO)");
 }
#elif defined(PLATFORM_POSIX)
 struct timeval tv;
 tv.tv_sec = msecs / 1000;
 tv.tv_usec = (msecs % 1000) * 1000;
 if(setsockopt(socket, SOL_SOCKET, SO_RCVTIMEO, &tv, sizeof(tv)) < 0) {
  SetSockError("setsockopt(SO_RCVTIMEO)");
 }
#endif
}
```

and I added them to SSLSocketData::OpenClient and between calling SSLServerSocket and SSLSocketData::Accept. I did not test timeouts on windows, but on linux they work fine.

```
bool SSLSocketData::OpenClient(const char *host, int port, bool nodelay, dword *my_addr, int
timeout, bool blocking)
{

 if(!Data::OpenClient(host, port, nodelay, my_addr, timeout, /*blocking*/true))
  return false;

 Data::ReadTimeout(timeout);
 Data::WriteTimeout(timeout);

 if(!(ssl = SSL_new(ssl_context)))
 {
  SetSSLError("OpenClient / SSL_new");
  return false;
 }
[...]

bool SSLSocketData::Close(int timeout_msec)
```

```
{
 if(ssl){
  if(!SSL_shutdown(ssl)){
   Data::StopWrite();
   SSL_shutdown(ssl);
  }

 }

 bool res = Data::Close(timeout_msec);
 if(ssl) {
  SSL_free(ssl);
  ssl = NULL;
 }
 return res;
}
```

Maybe there is better place to insert those timeouts (for example socket.cpp?). Any suggestions?
