NAME

Net::SSLeay::Handle − Perl module that lets SSL (HTTPS) sockets be handled as standard file handles.

SYNOPSIS

```
use Net::SSLeay::Handle qw/shutdown/;
my ($host, $port) = ("localhost", 443);

tie(*SSL, "Net::SSLeay::Handle", $host, $port);

print SSL "GET / HTTP/1.0\r\n";
shutdown(\*SSL, 1);
print while (<SSL>);
close SSL;
```

DESCRIPTION

Net::SSLeay::Handle allows you to request and receive HTTPS web pages using ‘‘old-fashion’’ file handles as in:

```
print SSL "GET / HTTP/1.0\r\n";
```

and

```
print while (<SSL>);
```

If you export the shutdown routine, then the only extra code that you need to add to your program is the tie function as in:

```
my $socket;
if ($scheme eq "https") {
    tie(*S2, "Net::SSLeay::Handle", $host, $port);
    $socket = \*S2;
else {
    $socket = Net::SSLeay::Handle−>make_socket($host, $port);
}
print $socket $request_headers;
...
```

FUNCTIONS

shutdown

```
shutdown(\*SOCKET, $mode)
```

Calls to the main **shutdown()** don’t work with tied sockets created with this module. This shutdown should be able to distinquish between tied and untied sockets and do the right thing.

debug

```
my $debug = Net::SSLeay::Handle−>debug()
Net::SSLeay::Handle−>debug(1)
```

Get/set debugging mode. Always returns the debug value before the function call. if an additional argument is given the debug option will be set to this value.

make_socket

```
my $sock = Net::SSLeay::Handle−>make_socket($host, $port);
```

Creates a socket that is connected to `$post` using `$port`. It uses `$Net::SSLeay::proxyhost` and proxyport if set and authentificates itself against this proxy depending on `$Net::SSLeay::proxyauth`. It also turns autoflush on for the created socket.

USING EXISTING SOCKETS

One of the motivations for writing this module was to avoid duplicating socket creation code (which is mostly error handling). The calls to **tie()** above where it is passed a `$host` and `$port` is provided for convenience testing. If you already have a socket connected to the right host and port, S1, then you can do something like:

```
my $socket \*S1;
if ($scheme eq "https") {
    tie(*S2, "Net::SSLeay::Handle", $socket);
    $socket = \*S2;
}
my $last_sel = select($socket); $| = 1; select($last_sel);
print $socket $request_headers;
...
```

Note: As far as I know you must be careful with the globs in the **tie()** function. The first parameter must be a glob (*SOMETHING) and the last parameter must be a reference to a glob (*SOMETHING_ELSE) or a scaler that was assigned to a reference to a glob (as in the example above)

Also, the two globs must be different. When I tried to use the same glob, I got a core dump.

EXPORT

None by default.

You can export the **shutdown()** function.

It is suggested that you do export **shutdown()** or use the fully qualified **Net::SSLeay::Handle::shutdown()** function to shutdown SSL sockets. It should be smart enough to distinguish between SSL and non-SSL sockets and do the right thing.

EXAMPLES

```
use Net::SSLeay::Handle qw/shutdown/;
my ($host, $port) = ("localhost", 443);

tie(*SSL, "Net::SSLeay::Handle", $host, $port);

print SSL "GET / HTTP/1.0\r\n";
shutdown(\*SSL, 1);
print while (<SSL>);
close SSL;
```

TODO

Better error handling. Callback routine?

CAVEATS

Tying to a file handle is a little tricky (for me at least).

The first parameter to **tie()** must be a glob (*SOMETHING) and the last parameter must be a reference to a glob (*SOMETHING_ELSE) or a scaler that was assigned to a reference to a glob ($s = *SOMETHING_ELSE). Also, the two globs must be different. When I tried to use the same glob, I got a core dump.

I was able to associate attributes to globs created by this module (like *SSL above) by making a hash of hashes keyed by the file head1.

CHANGES

Please see Net−SSLeay−Handle−0.50/Changes file.

BUGS

If you encounter a problem with this module that you believe is a bug, please create a new issue <https://github.com/radiator-software/p5-net-ssleay/issues/new> in the Net-SSLeay GitHub repository. Please make sure your bug report includes the following information:

- the code you are trying to run;
- your operating system name and version;
- the output of `perl −V`;

- the version of OpenSSL or LibreSSL you are using.

AUTHOR

Originally written by Jim Bowlin.

Maintained by Sampo Kellomäki between July 2001 and August 2003.

Maintained by Florian Ragwitz between November 2005 and January 2010.

Maintained by Mike McCauley between November 2005 and June 2018.

Maintained by Chris Novakovic, Tuure Vartiainen and Heikki Vatiainen since June 2018.

COPYRIGHT

LICENSE

SEE ALSO

Net::SSLeay, **perl** (1), http://openssl.org/