I added support for completely disabling client renegotiation on SSL/TLS servers in r35797. Client renegotiation is still considered a problem, even with secure renegotiation support.

It's now possible to either completely disable client renegotiation at all or to specify a maximum number of handshakes. The feature is opt-in, the default is as it was before, to allow arbitrary client renegotiation attempts. The feature is meant to help in scenarios where the OpenSSL extension is used to run a server that should not support client renegotiation for security reasons.

Because we don't support renegotiation in the OpenSSL extension, it wasn't possible to write explicit test cases, but I created a simple server script [1] that can be tested with tools such as OpenSSL's s_client [2]. For example, when running the server script at [1], testing the feature could be accomplished by:

`openssl s_client -connect localhost:8443`

And then pressing 'R' and Enter, you should receive an error like this:

```
RENEGOTIATING
140639302223680:error:1409E0E5:SSL routines:SSL3_WRITE_BYTES:ssl handshake failure:s3_pkt.c:591:
```

If you don't, any feedback is much appreciated!

[1] https://gist.github.com/2791400

History

#1 - 06/10/2012 01:53 AM - MartinBosslet (Martin Bosslet)

Updated in r35994. There is a generic renegotiation_cb attribute for SSLContext now instead of explicit configuration parameters. This reduces the code in osssl_ssl.c and gives users maximum flexibility on how they'd like to act upon renegotiation attempts.

A simple "disable client renegotiation entirely" callback could be implemented as follows:

```ruby
num_handshakes = 0
ctx.renegotiation_cb = lambda do |ssl|
  num_handshakes += 1
  raise RuntimeError.new("Client renegotiation disabled") if num_handshakes > 1
end
```

This way, the initial handshake passes, but further attempts will be rejected. I also updated the test server script at https://gist.github.com/2791400.

#2 - 09/04/2012 08:06 AM - MartinBosslet (Martin Bosslet)

- Status changed from Feedback to Closed

Closing, as it did not seem to have negative impact of any kind so far.