Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'SSL_key_update.3ossl' command

$ man SSL_key_update.3ossl

SSL_KEY_UPDATE(3ossl) OpenSSL SSL_KEY_UPDATE(3ossl)

NAME

SSL_key_update, SSL_get_key_update_type, SSL_renegotiate, SSL_renegotiate_abbreviated, SSL_renegotiate_pending - initiate and obtain information about updating connection keys

SYNOPSIS

```
#include <openssl/ssl.h>

int SSL_key_update(SSL *s, int updatetype);
int SSL_get_key_update_type(const SSL *s);

int SSL_renegotiate(SSL *s);
int SSL_renegotiate_abbreviated(SSL *s);
int SSL_renegotiate_pending(const SSL *s);
```

DESCRIPTION

SSL_key_update() schedules an update of the keys for the current TLS connection. If the updatetype parameter is set to SSL_KEY_UPDATE_NOT_REQUESTED then the sending keys for this connection will be updated and the peer will be informed of the change. If the updatetype parameter is set to SSL_KEY_UPDATE_REQUESTED then the

sending keys for this connection will be updated and the peer will be informed of the change along with a request for the peer to additionally update its sending keys. It is an error if updatetype is set to SSL_KEY_UPDATE_NONE.

SSL_key_update() must only be called after the initial handshake has been completed and TLSv1.3 has been negotiated, at the same time, the application needs to ensure that the writing of data has been completed. The key update will not take place until the next time an IO operation such as SSL_read_ex() or SSL_write_ex() takes place on the connection. Alternatively SSL_do_handshake() can be called to force the update to take place immediately.

SSL_get_key_update_type() can be used to determine whether a key update operation has been scheduled but not yet performed. The type of the pending key update operation will be returned if there is one, or SSL_KEY_UPDATE_NONE otherwise.

SSL_renegotiate() and SSL_renegotiate_abbreviated() should only be called for connections that have negotiated TLSv1.2 or less. Calling them on any other connection will result in an error.

When called from the client side, SSL_renegotiate() schedules a completely new handshake over an existing SSL/TLS connection. The next time an IO operation such as SSL_read_ex() or SSL_write_ex() takes place on the connection a check will be performed to confirm that it is a suitable time to start a renegotiation. If so, then it will be initiated immediately. OpenSSL will not attempt to resume any session associated with the connection in the new handshake.

When called from the client side, SSL_renegotiate_abbreviated() works in the same was as SSL_renegotiate() except that OpenSSL will attempt to resume the session associated with the current connection in the new

handshake.

When called from the server side, SSL_renegotiate() and SSL_renegotiate_abbreviated() behave identically. They both schedule a request for a new handshake to be sent to the client. The next time an IO operation is performed then the same checks as on the client side are performed and then, if appropriate, the request is sent. The client may or may not respond with a new handshake and it may or may not attempt to resume an existing session. If a new handshake is started then this will be handled transparently by calling any OpenSSL IO function.

If an OpenSSL client receives a renegotiation request from a server then again this will be handled transparently through calling any OpenSSL IO function. For a TLS connection the client will attempt to resume the current session in the new handshake. For historical reasons, DTLS clients will not attempt to resume the session in the new handshake.

The SSL_renegotiate_pending() function returns 1 if a renegotiation or renegotiation request has been scheduled but not yet acted on, or 0 otherwise.

RETURN VALUES

SSL_key_update(), SSL_renegotiate() and SSL_renegotiate_abbreviated() return 1 on success or 0 on error.

SSL_get_key_update_type() returns the update type of the pending key update operation or SSL_KEY_UPDATE_NONE if there is none.

SSL_renegotiate_pending() returns 1 if a renegotiation or renegotiation request has been scheduled but not yet acted on, or 0 otherwise.

SEE ALSO

ssl(7), SSL_read_ex(3), SSL_write_ex(3), SSL_do_handshake(3)

HISTORY

The SSL_key_update() and SSL_get_key_update_type() functions were added in OpenSSL 1.1.1.

COPYRIGHT