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_pending.3ossl' command

$ man SSL_pending.3ossl

SSL_PENDING(3ossl) OpenSSL SSL_PENDING(3ossl)

NAME

SSL_pending, SSL_has_pending - check for readable bytes buffered in an SSL object

SYNOPSIS

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

int SSL_pending(const SSL *ssl);
int SSL_has_pending(const SSL *s);
```

DESCRIPTION

Data is received in whole blocks known as records from the peer. A whole record is processed (e.g. decrypted) in one go and is buffered by OpenSSL until it is read by the application via a call to SSL_read_ex(3) or SSL_read(3).

SSL_pending() returns the number of bytes which have been processed, buffered and are available inside ssl for immediate read.

If the SSL object's read_ahead flag is set (see SSL_CTX_set_read_ahead(3)), additional protocol bytes (beyond the

current record) may have been read containing more TLS/SSL records. This also applies to DTLS and pipelining (see SSL_CTX_set_split_send_fragment(3)). These additional bytes will be buffered by OpenSSL but will remain unprocessed until they are needed. As these bytes are still in an unprocessed state SSL_pending() will ignore them. Therefore, it is possible for no more bytes to be readable from the underlying BIO (because OpenSSL has already read them) and for SSL_pending() to return 0, even though readable application data bytes are available (because the data is in unprocessed buffered records).

SSL_has_pending() returns 1 if s has buffered data (whether processed or unprocessed) and 0 otherwise. Note that it is possible for SSL_has_pending() to return 1, and then a subsequent call to SSL_read_ex() or SSL_read() to return no data because the unprocessed buffered data when processed yielded no application data (for example this can happen during renegotiation). It is also possible in this scenario for SSL_has_pending() to continue to return 1 even after an SSL_read_ex() or SSL_read() call because the buffered and unprocessed data is not yet processable (e.g. because OpenSSL has only received a partial record so far).

RETURN VALUES

SSL_pending() returns the number of buffered and processed application data bytes that are pending and are available for immediate read. SSL_has_pending() returns 1 if there is buffered record data in the SSL object and 0 otherwise.

SEE ALSO

SSL_read_ex(3), SSL_read(3), SSL_CTX_set_read_ahead(3), SSL_CTX_set_split_send_fragment(3), ssl(7)

HISTORY

The SSL_has_pending() function was added in OpenSSL 1.1.0.

COPYRIGHT