MatrixSSL 3.2.X Open Source Release Notes

PeerSec Networks, Inc. 410 Broadway Ave E. #205 Seattle, WA 98102 **T** 425.646.7850 **F** 206.501.4366

support@peersec.com www.peersec.com

Overview

This document is an aggregation of the release notes from MatrixSSL versions 3.2.0 and 3.2.1.

Who Is This Document For?

- Software developers that are securing applications with MatrixSSL
- Anyone wanting to learn more about MatrixSSL

MatrixSSL 3.2.0

This section highlights the differences between version 3.1.4 and 3.2.0

Enhancements to Features and Functionality

Added TLS 1.1 protocol support

TLS 1.1 protocol support is now available in the open source version of MatrixSSL. The protocol is enabled/disabled through the compile time define `USE_TLS_1_1` in *matrixsslConfig.h*. If enabled, the protocol negotiation will default to TLS 1.1 for any communicating SSL peer that also supports it. It is also now possible to disable SSL 3.0 using the `DISABLE_SSLV3` define if only TLS version protocols are desired

Added PKCS#8 private key parsing

The PKCS#8 standard is becoming more widespread for newly issued private keys. PKCS#8 parsing is now included by default in the open source version of MatrixSSL. This support is built into the existing `matrixSslLoadRsaKeys` API.

IN and OUT default buffer sizes

Previous versions of MatrixSSL used a single compile time setting for the default internal input and output buffers. This define has now been split into `SSL_DEFAULT_OUT_BUF_SIZE` and `SSL_DEFAULT_IN_BUF_SIZE` defines to give the user more memory control for the specific use case. For example, if the integrator knows that incoming data will be short requests and the outgoing reply data will be large files, the `SSL_DEFAULT_IN_BUF_SIZE` may be set smaller that `SSL_DEFAULT_OUT_BUF_SIZE` to help streamline this implementation.

Zero length SSL records now returned to user

Callers of `matrixSslReceivedData` will now be informed of zero length SSL records with the standard return code of MATRIXSSL_APP_DATA and length values of 0. Previous versions of MatrixSSL would silently discard empty records.

Public API Changes

Added matrixSslEncodeToOutdata

An SSL record encoding alternative to the existing `matrixSslGetWritebuf`/ `matrixSslEncodeWritebuf` combination has been introduced. The new `matrixSslEncodeToOutdata` enables integrators to encode plaintext from where it exists in an external memory location.

This differs from the `matrixSslEncodeWritebuf` API that requires the plaintext has been written or copied into the internal library buffer. The new `matrixSslEncodeToOutdata` function will leave the plaintext buffer untouched while encoding to the internal library buffer. The encoded data is still retrieved for sending using `matrixSslGetOutdata`. Please see the API documentation for more information on this new function.

MatrixSSL 3.2.1

This section highlights the differences between version 3.2.0 and 3.2.1

Feature Addition

PKCS#8 Parsing Added for Parsing Memory Buffer Keys

MatrixSSL 3.2.0 included default PKCS#8 parsing when reading PEM keys from files. This version adds default support for PKCS#8 parsing formats when reading binary keys from memory locations.

Bug Fix

Multiple App Data Records Parsing in TLS 1.1

MatrixSSL 3.2.0 did not correctly parse multiple application data records if they were in a single flight of traffic. **This is an important bug fix if using MatrixSSL 3.2.0 and TLS 1.1**.

Minor Tweaks

Client will not create CLIENT_HELLO if no valid cipher suites are found

If the user has not loaded the correct key material to match the set of enabled cipher suites the CLIENT_HELLO message will not be created and an error will be returned from `matrixSslNewClientSession`.

Disabled REHANDSHAKE_TEST in Client Application

The default *client.c* compile performed a re-handshake test after the initial server connection. This could cause a bit of confusion if the user was using the default client to test a connection to a server that does not support re-handshaking.

Removed the use of likely( ) and unlikey( )

These GCC branch hint macros have been removed from the code due to underuse and namespace collision.

Support and Bug Reporting

Contacting Support or Reporting Bugs

Email support@peersec.com