MatrixSSL 3.1.4 Open Source Release Notes

Overview

MatrixSSL 3.1.4 Release Notes

Enhancements to Features and Functionality

Public API Changes

Support and Bug Reporting

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

Thank you for choosing MatrixSSL. The 3.1 version is a major revision to the previous releases and enables users to implement strong SSL security into their applications faster than ever. With a design emphasis on further reducing memory usage and providing an easier integration API, MatrixSSL 3.1 is the security solution for virtually any networked application on any platform.

If you are migrating from a 2.x version of MatrixSSL you will want to read the Migrating To MatrixSSL 3 document to learn more about the specific changes.

Who Is This Document For?

- Software developers that are securing applications with MatrixSSL
- Software developers upgrading to MatrixSSL 3.1.4 from a previous version
- Anyone wanting to learn more about MatrixSSL 3.1

MatrixSSL 3.1.4 Release Notes

This section highlights the differences between version 3.1.3 and 3.1.4

Enhancements to Features and Functionality

Primary crypto algorithms now have configuration options for size vs. speed tradeoffs

Previous versions of MatrixSSL had an undocumented compile time define (`SMALL_CODE`) that influenced the binary code size of some symmetric cipher algorithms. Each algorithm that used this define has now been given its own define to control whether the user wants to build the library for faster algorithm support at the cost of an increased binary code size. The size vs. speed tradeoff is platform dependent but, in general, the speed improvements will be about 5%-10% at the cost of 10-20KB for each algorithm. The default, in each case, is that these defines are disabled in *cryptoConfig.h* to compile in favor of smallest binary footprint.

Define	Notes
`PS_AES_IMPROVE_PERF_INCREASE_CODESIZE`	Enable to improve AES performance at the cost of a larger binary footprint. Speed vs size tradeoffs are platform dependent.
`PS_3DES_IMPROVE_PERF_INCREASE_CODESIZE`	Enable to improve 3DES performance at the cost of a larger binary footprint. Speed vs size tradeoffs are platform dependent.
`PS_MD5_IMPROVE_PERF_INCREASE_CODESIZE`	Enable to improve MD5 performance at the cost of a larger binary footprint. Speed vs size tradeoffs are platform dependent.

Define	Notes
PS_SHA1_IMPROVE_PERF_INCREASE_CODESIZE	Enable to improve SHA1 performance at the cost of a larger binary footprint. Speed vs size tradeoffs are platform dependent.

RSA algorithm now has configuration option for memory usage vs. speed tradeoff

A pair of defines have been added to determine whether the RSA algorithm should be compiled for smaller RAM usage or faster performance. The default is to compile for smaller RAM usage.

Define	Notes
PS_PUBKEY_OPTIMIZE_FOR_SMALLER_RAM	The memory savings for optimizing for ram is around 50%
PS_PUBKEY_OPTIMIZE_FOR_FASTER_SPEED	The speed gain for optimizing for speed is around 5%

Servers can now disable specific cipher suites at runtime

Cipher suites that have been compiled into the library can now be programatically disabled (and re-enabled) on a per-session basis. This is useful for servers that wish to limit the supported ciphers suites for a specific connecting client. A new API, `matrixSslSetCipherSuiteEnabledStatus`, has been added to support this functionality. Please see the MatrixSSL API documentation for detailed information on this new feature.

An Xcode project for iPhone development is now included

In the *apps/iphone* directory the user can now find a Mac Xcode project for developing SSL/TLS client applications for the iPhone.

Server compatibility with Chrome browsers that use “false start”

The Google Chrome browser has introduced a new protocol mechanism called “false start” that is incompatible with strict TLS implementations that do not allow application data exchange before the handshake protocol is complete. Enabling `ENABLE_FALSE_START` in *matrixsslConfig.h* will allow newer versions of the Chrome browser to connect with MatrixSSL servers.

A new explicit int16 data type has been added

The *osdep.h* file now includes a typedef for a 16-bit integer type called `int16`. The initial internal use of this new data type can be found in the *pstm.c* math function to help improve performance on some platforms.

Public API Changes

Compile-time define for file system support has been renamed

The `USE_FILE_SYSTEM` define has been renamed to include a `PS_` prefix so that it is now `PS_USE_FILE_SYSTEM`. In addition, this define is no longer present in the *coreConfig.h* header file. It should be included in the platform build environment as a compile-time define if file system support is needed.

Return types changed for osdep.c Open and Close routines

The platform interface functions implemented in osdep.c have undergone prototype changes.

New 3.1.4 Prototype	Old 3.1.3 Prototype
`int osdepTimeOpen(void)`	`int32 osdepTimeOpen(void)`
`void osdepTimeClose(void)`	`int32 osdepTimeClose(void)`
`int osdepEntropyOpen(void)`	`int32 osdepEntropyOpen(void)`
`void osdepEntropyClose(void)`	`int32 osdepEntropyClose(void)`
`int osdepTraceOpen(void)`	`int32 osdepTraceOpen(void)`
`void osdepTraceClose(void)`	`int32 osdepTraceClose(void)`
`int osdepMutexOpen(void)`	`int32 osdepMutexOpen(void)`
`void osdepMutexClose(void)`	`int32 osdepMutexClose(void)`

Support and Bug Reporting

Contacting Support or Reporting Bugs

Email support@peersec.com