MatrixSSL 3.4.1 Open Source Release Notes

Electronic versions are uncontrolled unless directly accessed from the QA Document Control system.

Printed version are uncontrolled except when stamped with ‘VALID COPY’ in red.

External release of this document may require a NDA.

TABLE OF CONTENTS

1 FEATURE ADDITIONS

SSL/TLS features new in MatrixSSL 3.4

1.1 Client Authentication

Client authentication (mutual authentication) is now available in the open source version of MatrixSSL. Enable the `USE_CLIENT_AUTH` define in *matrixsslConfig.h* to add support for this feature to the library.

Clients and servers are both supported and the example applications implement client authentication for reference. The `sslTest` utility will exercise the client authentication handshake variations as well.

For further details, see the MatrixSSL API documentation included in this package.

1.2 RSA Assembly Code Optimizations

Assembly code optimizations that were previously only available in commercial versions of MatrixSSL are now included in the open source packages. Optimizations for common processors such as ARM, x86, x86_64, and MIPS32 can now be enabled with the use of compile-time defines. RSA operations gain a significant speed advantage using these optimizations.

Common POSIX-based operating systems such as Linux and MacOS that use the provided Makefile system will automatically take advantage of these assembly optimizations.

For more information, see the MatrixSSL Developers Guide and the MatrixSSL Porting Guide.

1.3 Certificate Revocation Support

Two new APIs have been added to support common Certificate Revocation mechanisms. If a Certificate Authority uses the CRL Distribution Points extension to identify the URI where a CRL can be found, use the new `matrixSslGetCRL` API to aid in the fetch. If a local CRL is available use the `matrixSslLoadCRL` API to register the revoked certificates with the CA for testing during the SSL handshake.

The client example application implements these two new APIs as a reference.

2 API CHANGES

Public function changes for those updating from MatrixSSL 3.3.1

2.1 Session ID Management

Client management of the session ID for resumption is now more explicit. The new `matrixSslNewSessionId` and `matrixSslDeleteSessionId` APIs enable library control of the `sslSessionId_t` parameter used in `matrixSslNewClientSession`. Refer to the API documentation for more details.

The `matrixSslInitSessionId` macro has been removed from the library as part of this improvement.

2.2 Additional Parameter for NewSession APIs

An additional parameter has been added to the `matrixSslNewServerSession` and `matrixSslNewClientSession` APIs for compatibility with MatrixDTLS packages. For SSL usage, the final parameter should be 0 to both of these functions.

2.3 Platform Layer osdepMutexClose Uses int Return Type

This function prototype previously used a `void` return value. This change to an `int` return type was made simply to keep the *core/<platform>* module APIs consistent.

3 BUG FIXES AND IMPROVEMENTS

3.1 Timing Improvements for CBC Padding and MAC Attacks

Researches recently published details on exploiting a known time variation in SSL protocol processing of encrypted records that use a CBC cipher mode. Details can be found at http://www.isg.rhul.ac.uk/tls/ but the main idea behind the countermeasure is that the processing time for a given record length should always be the same. MatrixSSL 3.4.1 includes countermeasure code in the handling of CBC decryption to mitigate the risk of this attack.

3.2 Individual notBefore and notAfter Time Types

X.509 certificate parsing now includes separate time format fields for the `notBefore` and `notAfter` identifiers. UTCTIME and GENERALIZEDTIME are still supported. However, it is not correct to assume both must be the same type. The `psX509Cert_t` structure accessible through the certificate callback will contain `notBeforeTimeType` and `notAfterTimeType` members instead of `timeType`.

3.3 TLS 1.1 Alert Codes Passed Correctly to User

The alert type and description were not correctly passed to the user via `matrixSslReceivedData` when the TLS 1.1 protocol was being used.

3.4 Parsing Improvements

3.4.1 Improved parsing for X.509 GeneralNames

The length parser in the internal `parseGeneralNames` function assumed values less than 255. All lengths are supported now.

3.4.2 Improved PKCS#8 parsing

Optional Attributes in a PKCS#8 format are now properly recognized.

3.4.3 PKCS#12 improvements

The key generation algorithm is now more flexible. Previous implementations assumed a salt length of 8 bytes. Salts may now be up to 20 bytes. Also, certificates will be re-ordered in a child-to-parent hierarchy after the parse is complete.