

WS63V100 TLS&DTLS

开发指南

文档版本 02

发布日期 2024-10-14

前言

概述

本文档主要介绍 TLS/DTLS 组件的开发实现示例。

TLS/DTLS 以及其他加密套基于开源组件 mbedtls 3.1.0 实现，详细说明请参见官方说明：<https://tls.mbed.org/api/index.html>

如果官方说明版本与 SDK 版本不一致，请参考官方 release 说明：
<https://github.com/ARMmbed/mbedtls/releases>

产品版本

与本文档相对应的产品版本如下。

| 产品名称 | 产品版本 |
|------|------|
| WS63 | V100 |






读者对象

本文档主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

| 符号 | 说明 |
|--|--|
|  危险 | 表示如不可避免则将会导致死亡或严重伤害的具有高等级风险的危害。 |
|  警告 | 表示如不可避免则可能导致死亡或严重伤害的具有中等级风险的危害。 |
|  注意 | 表示如不可避免则可能导致轻微或中度伤害的具有低等级风险的危害。 |
|  须知 | 用于传递设备或环境安全警示信息。如不可避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。 |
|  说明 | 对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。 |

修改记录

| 文档版本 | 发布日期 | 修改说明 |
|-------|------------|--------------------------------|
| 02 | 2024-10-14 | 更新“4.4 关于 FCC 认证默认配置变更说明”小节内容。 |
| 01 | 2024-04-10 | 第一次正式版本发布。 |
| 00B01 | 2024-03-15 | 第一次临时版本发布。 |

目 录

前言i

1 API 接口说明.....1

1.1 结构体说明.....1

1.2 API 列表.....1

1.3 配置说明1

2 开发指南.....2

3 硬件适配.....3

3.1 配置说明3

3.2 适配说明3

3.2.1 AES 适配3

3.2.2 大数模幂适配.....3

3.2.3 随机数适配.....4

3.2.4 RSA 数字签名适配.....4

3.2.5 HASH 算法适配4

3.2.6 ECP 适配.....4

4 注意事项.....5

4.1 关于配置 SSL 接收缓存的注意事项.....5

4.2 关于数字证书有效期验证的注意事项6

4.3 关于部分默认配置变更的说明6

4.4 关于 FCC 认证默认配置变更说明.....6

1 API 接口说明

1.1 结构体说明

1.2 API 列表

1.3 配置说明

1.1 结构体说明

mbedtls 详细的结构体说明请参考官方说明文档：
<https://tls.mbed.org/api/annotated.html>

1.2 API 列表

mbedtls 详细的 API 说明请参考官方说明文档：
https://tls.mbed.org/api/globals_func.html

1.3 配置说明

mbedtls 详细的配置项说明请参考官方说明文档：
https://tls.mbed.org/api/config_8h.html#ab3bca0048342cf2789e7d170548ff3a5

2 开发指南

MBEDTLS 详细的开发 DEMO 请参考官方说明文档：
<https://tls.mbed.org/api/modules.html>

3 硬件适配

3.1 配置说明

3.2 适配说明

3.1 配置说明

在工程的 `build\config\target_config\ws63\config.py` 中的对应编译目标中添加 `MBEDTLS_HARDEN_OPEN` 宏，开启硬件加速回调接口注册功能。

在 `mbbedtls_v3.1.0\harden\platform\connect\mbbedtls_platform_hardware_config.h` 中开启对应的算法宏，会直接调用硬件驱动接口。

目前支持的硬件算法有 AES, RSA, HASH, 大数模幂，随机数，ECP 算法。

3.2 适配说明

3.2.1 AES 适配

- 使能 `MBEDTLS_AES_ALT` 后，AES 算法在使用硬件加速器时，会锁定硬件加速器资源，即 AES 操作是阻塞的，直至驱动获取资源或超时返回失败。

3.2.2 大数模幂适配

- 使能 `MBEDTLS_BIGNUM_EXP_MOD_USE_HARDWARE` 后，会调用硬件驱动接口完成大数模幂运算。

3.2.3 随机数适配

使能 MBEDTLS_ENTROPY_HARDWARE_ALT 后，系统会选用默认增加硬件随机数作为一个强随机数源。如果此宏被关闭，而用户也没有注册其他强随机数源，会导致 mbedTLS 无法提供安全随机数，影响系统的安全性。

3.2.4 RSA 数字签名适配

使能 MBEDTLS_RSA_ALT 编译宏之后，mbedTLS 会对 RSA 数字签名操作和验签操作进行硬件加速。

3.2.5 HASH 算法适配

目前 WS63 规格支持的硬件加速 HASH 算法有 SHA1,SHA224,SHA256,SHA384,SHA512,分别开启 MBEDTLS_SHA1_USE_HARDWARE, MBEDTLS_SHA224_USE_HARDWARE, MBEDTLS_SHA256_USE_HARDWARE, MBEDTLS_SHA384_USE_HARDWARE, MBEDTLS_SHA512_USE_HARDWARE 来使能。

3.2.6 ECP 适配

目前 WS63 规格支持的硬件加速 ECP 算法有 SECP192R1, SECP224R1,SECP256R1,SECP384R1,SECP521R1,BP256R1,BP384R1,BP512R1,CURVE25519,CURVE448,分别开启 MBEDTLS_SECP192R1_USE_HARDWARE, MBEDTLS_SECP224R1_USE_HARDWARE, MBEDTLS_SECP256R1_USE_HARDWARE, MBEDTLS_SECP384R1_USE_HARDWARE, MBEDTLS_SECP521R1_USE_HARDWARE, MBEDTLS_BP256R1_USE_HARDWARE, MBEDTLS_BP384R1_USE_HARDWARE, MBEDTLS_BP512R1_USE_HARDWARE, MBEDTLS_CURVE25519_USE_HARDWARE,MBEDTLS_CURVE448_USE_HARDWARE 来开启。

4 注意事项

- 4.1 关于配置 SSL 接收缓存的注意事项
- 4.2 关于数字证书有效期验证的注意事项
- 4.3 关于部分默认配置变更的说明
- 4.4 关于 FCC 认证默认配置变更说明

4.1 关于配置 SSL 接收缓存的注意事项

- SSL 接收缓存由编译项 MBEDTLS_SSL_IN_CONTENT_LEN 控制，默认为 16KB。如果实际应用中，用户可以确保 SSL 上层数据包的最大长度不超过 2KB 或 4KB，则可以通过 mbedtls_ssl_conf_max_frag_len 接口设置 SSL 接收缓存的长度，达到节省内存的目的。
注意：mbedtls_ssl_conf_max_frag_len 接口的调用必须先于 mbedtls_ssl_setup 接口。
- 考虑到多级数字证书可能导致 TLS 握手包的长度大于 1KB，因此调用 mbedtls_ssl_conf_max_frag_len 接口时，只有 mfl_code 为 MBEDTLS_SSL_MAX_FRAG_LEN_2048 或 MBEDTLS_SSL_MAX_FRAG_LEN_4096 时，mbedtls 才会修改接收缓存；如果 mfl_code 为 MBEDTLS_SSL_MAX_FRAG_LEN_512 或 MBEDTLS_SSL_MAX_FRAG_LEN_1024，则接收缓存仍然为 16KB。
- 如果修改 SSL 接收缓存为 2KB 或 4KB 后，Client 端接收到 Server 端的一个大于 2KB 或 4KB 的数据包，此时 mbedtls_ssl_read 接口会返回失败，错误码为 MBEDTLS_ERR_SSL_MSG_TOO_LONG（此为新增的一个特定错误码），当用户获得此错误码时，必须关闭 SSL 连接，不允许继续从 SSL 链路接收数据。

- 通过 `mbedtls_ssl_conf_max_frag_len` 接口设置 SSL 接收缓存，目前只对 SSL Client 有效。

4.2 关于数字证书有效期验证的注意事项

由于 WS63 平台无 Real Time Controller，因此系统启动后，无法获取 UTC 时间，这种情况下数字证书的有效期验证会失败，导致 TLS 建链失败。针对此种情况，`mbedtls` 默认关闭 `MBEDTLS_HAVE_TIME_DATE`，此时 TLS 的证书校验会关闭。如果用户可以确保 TLS 证书校验之前，可以通过其他方式获取 UTC 时间（例如：SNTP），则可以打开 `MBEDTLS_HAVE_TIME_DATE` 编译宏。

4.3 关于部分默认配置变更的说明

MbedTLS 安全库的默认配置见 `include/mbedtls/mbedtls_config.h` 文件。开源版本默认开启大部分功能，LiteOS 对 MbedTLS 的默认配置进行了适度修改，主要目的是增强 MbedTLS 的安全性，降低代码体积。修改后的 MbedTLS 满足 IoT 绝大部分场景，改动的主要原则如下：

- 默认配置必须保证安全性要求。
- 关闭不安全算法或功能。
- 关闭不适合 IoT 场景的功能，例如 TLS Server 模式、X509 证书签名请求 CSR。
- 关闭不适合 IoT 场景的算法，例如 `SECP521R1` ECC 曲线，IoT 场景下推荐使用 128 比特安全强度的 ECC 曲线。
- 关闭不常用算法或功能，例如 IoT 场景不常用的 PKCS#12 证书。如果打开 PKCS#12，可能存在一定的应用风险。

4.4 关于 FCC 认证默认配置变更说明

FCC 认证为 EMC 强制性认证，主要针对 9K-3000GHZ 的电子电器产品，内容涉及无线电、通信等各方面。FCC 认证过程依赖 `KEY_EXCHANGE_ECDHE_RSA` 算法，此算法默认未开启。如果设备有通过 FCC 认证的需求，可以开启此算法。开启方法：修改

`open_source/mbedtls/mbedtls_v3.1.0/harden/platform/connect/mbedtls_platform_har`

dware_config.h 文件，将#undef

MBEDTLS_KEY_EXCHANGE_ECDHE_RSA_ENABLED 注释掉。