



XRADIO MBEDTLS应用指南

Revision 1.0

Otc 23, 2019

Declaration

THIS DOCUMENTATION IS THE ORIGINAL WORK AND COPYRIGHTED PROPERTY OF XRADIO TECHNOLOGY ("XRADIO"). REPRODUCTION IN WHOLE OR IN PART MUST OBTAIN THE WRITTEN APPROVAL OF XRADIO AND GIVE CLEAR ACKNOWLEDGEMENT TO THE COPYRIGHT OWNER.

THE PURCHASED PRODUCTS, SERVICES AND FEATURES ARE STIPULATED BY THE CONTRACT MADE BETWEEN XRADIO AND THE CUSTOMER. PLEASE READ THE TERMS AND CONDITIONS OF THE CONTRACT AND RELEVANT INSTRUCTIONS CAREFULLY BEFORE USING, AND FOLLOW THE INSTRUCTIONS IN THIS DOCUMENTATION STRICTLY. XRADIO ASSUMES NO RESPONSIBILITY FOR THE CONSEQUENCES OF IMPROPER USE (INCLUDING BUT NOT LIMITED TO OVERVOLTAGE, OVERCLOCK, OR EXCESSIVE TEMPERATURE).

THE INFORMATION FURNISHED BY XRADIO IS PROVIDED JUST AS A REFERENCE OR TYPICAL APPLICATIONS, ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENT DO NOT CONSTITUTE A WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. XRADIO RESERVES THE RIGHT TO MAKE CHANGES IN CIRCUIT DESIGN AND/OR SPECIFICATIONS AT ANY TIME WITHOUT NOTICE.

NOR FOR ANY INFRINGEMENTS OF PATENTS OR OTHER RIGHTS OF THE THIRD PARTIES WHICH MAY RESULT FROM ITS USE. NO LICENSE IS GRANTED BY IMPLICATION OR OTHERWISE UNDER ANY PATENT OR PATENT RIGHTS OF XRADIO. THIRD PARTY LICENCES MAY BE REQUIRED TO IMPLEMENT THE SOLUTION/PRODUCT. CUSTOMERS SHALL BE SOLELY RESPONSIBLE TO OBTAIN ALL APPROPRIATELY REQUIRED THIRD PARTY LICENCES. XRADIO SHALL NOT BE LIABLE FOR ANY LICENCE FEE OR ROYALTY DUE IN RESPECT OF ANY REQUIRED THIRD PARTY LICENCE. XRADIO SHALL HAVE NO WARRANTY, INDEMNITY OR OTHER OBLIGATIONS WITH RESPECT TO MATTERS COVERED UNDER ANY REQUIRED THIRD PARTY LICENCE.

Revision History

Version	Date	Summary of Changes
1.0	2019-10-23	Initial Version

Contents

Declaration.....	2
Revision History.....	3
Contents.....	4
1 mbedtls 的配置文件.....	5
2 如何给 mbedtls 开打印.....	5
3 mbedtls 中的常见问题.....	6
3.1 连接时，死机重启.....	6
3.2 握手失败，返回-0x7200 错误码.....	6
3.3 握手失败，返回-0x7780 错误码.....	6
3.4 mbedtls 握手速度慢.....	7
3.5 如何配置合适的加解密套件.....	7

1 mbedtls 的配置文件

可通过 mbedtls 的配置文件，来配置、裁剪 mbedtls 的功能，减少 mbedtls 的资源占用。

1. 文件位置

配置文件在 include/net/mbedtls/configs 目录中。SDK 选择的配置文件为 config-xr-mini-cliserv.h。

2. 修改 SDK 配置文件

mbedtls 的配置文件需要在 Makefile 文件中修改。mbedtls 的 Makefile 文件在 /src/net/mbedtls-2.2.0 目录中，修改项为 CC_FLAGS += -DMBEDTLS_CONFIG_FILE='<config-xr-mini-cliserv.h>'。一般不建议修改为其他配置文件。

3. 配置文件的修改

配置文件主要是通过添加或减去一些宏来配置 mbedtls 的功能，比如支持的加解密算法、密钥协商算法、输入输出缓冲区大小等。配置文件所有的宏都来源于 include/net/mbedtls/config.h，该文件对每个宏都有详细的解释。也可参考链接：

<https://www.rt-thread.org/document/site/submodules/mbedtls/docs/footprint-optimization-guide/>

2 如何给 mbedtls 开打印

mbedtls 是以连接为单位，每条连接的资源都相互独立，比如打印函数、发送/接收函数、输入输出缓冲区等。故 mbedtls 的每条连接都可以单独设置打印函数，但打印等级不能单独设置。

1. 打开打印宏

在 include/net/mbedtls/configs/config-xr-mini-cliserv.h 文件中，将宏 MBEDTLS_DEBUG_C 打开。

2. 注册打印函数

在初始化一个 mbedtls 连接时，调用 mbedtls_ssl_conf_dbg 为这条连接注册打印回调函数。debug_function 为自己编写的打印函数。例如：

```
static void mbedtls_debug(void *ctx, int level, const char *file, int line, const char *str)
{
    printf("%s:%04d: %s", file, line, str);
}
```

```
mbedtls_ssl_conf_dbg(&conf, mbedtls_debug, stdout );
```

具体可参考 /src/net/mbedtls-2.2.0/mbedtls.c 或 /project/common/cmd/tls/client.c 中例子。

3. 设置打印等级

注册完打印回调函数后，调用 mbedtls_debug_set_threshold 函数设置打印等级。mbedtls 有 5 个打印等级 0:No debug 1>Error 2:State change 3:Informational 4:Verbose。一般设置为 3。

备注：如果使用 SDK 中 HTTPC 模块中的 HTTPC_get 函数下载 https 链接文件，则只需要打开宏 MBEDTLS_DEBUG_C，修改 /src/net/mbedtls-2.2.0/mbedtls.c 文件的宏 TLS_DEBUG_LEVEL，一般设置为 3。

3 mbedtls 中的常见问题

3.1 连接时，死机重启

查看调用 mbedtls 连接的线程栈大小，建议栈空间大于等于 3K，具体多少需要实际调试。

3.2 握手失败，返回-0x7200 错误码

1. 排查方式

搜索-0x7200 得到如下结果：

```
#define MBEDTLS_ERR_SSL_INVALID_RECORD -0x7200  /**< An invalid SSL record was received. */
```

这个错误表示 SSL 记录层收到一个无效的帧。产生这个问题的原因有两个：

1. 访问的服务器不支持数据分片。
2. mbedtls 设置输入输出缓冲区太小。

2. 解决办法：

增加 mbedtls 的输入输出缓冲区大小，修改 include/net/mbedtls/configs/config-xr-mini-cliserv.h 的宏 MBEDTLS_SSL_MAX_CONTENT_LEN 为(16*1024)。目前很多服务器都默认不支持数据分片，所以为了增加 mbedtls 的兼容性，设置缓冲区大小为 16K（最大为 16K），增加缓存区大小，会增加内存占用。

3.3 握手失败，返回-0x7780 错误码

1. 排查方式

搜索-0x7780 得到如下结果：

```
#define MBEDTLS_ERR_SSL_FATAL_ALERT_MESSAGE -0x7780  /**< A fatal alert message was received from our peer. */
```

该宏意思为，记录层接收到对端的一个致命警告。记录层的警告较多，可通过 mbedtls 的打印查看是什么类型的警告，常见的警告是客户端和服务端解密套件不同引起的。

2. 解决办法

确定警告类型：

- (1) 打开 mbedtls 打印，在通过打印确定错误位置

```
mbedtls: ssl_tls.c:3905 got an alert message, type: [2:40]
mbedtls: ssl_tls.c:3913 is a fatal alert message (msg 40)
mbedtls: ssl_cli.c:1402 mbedtls_ssl_read_record() returned -30592 (-0x7780)
mbedtls: ssl_tls.c:6286 <= handshake
[inf] _TLSConnectNetwork(430): failed ! mbedtls_ssl_handshake returned -0x7780
mbedtls: ssl_tls.c:6838 => write close notify
mbedtls: ssl_tls.c:6854 <= write close notify
```

- (2) 通过打印确定警告类型：警告类型是 2，致命警告，编码为 40。

(3) 查询 RFC4346 警报协议的相关资料，编码为 40 的警告为双方无法协商安全参数，即加密套件不匹配导致的。

3.4 mbedtls 握手速度慢

握手速度慢一般有两个原因：

1. 网络不稳定，导致数据传输慢。
2. 握手阶段密钥协商算法计算速度慢，常见于 ECDHE 和 ECDSA 密钥协商算法，解决办法是在 config-xr-mini-cliserv.h 添加宏 `#define MBEDTLS_ECP_NIST_OPTIM`，为每个 NIST 启用特定的实例，使相应曲线上的操作快 4 到 8 倍，缺点是 ROM 占用大。

3.5 如何配置合适的加解密套件

不同的加解密套件占用的资源不一样，比如内存占用和加解密时间占用，所以在客户端配置合适的加解密套件可以尽量减少资源的消耗。

在配置客户端加解密套件前，必须要知道需要连接的服务器支持的加解密套件，可通过链接“<https://www.ssllabs.com/ssltest/>”测试并获取服务器的信息。

输入需要测试的网址：



等待分析完成：

SSL Report: dl.stream.qqmusic.qq.com

Assessed on: Fri, 21 Dec 2018 12:12:03 UTC | [Hide](#) | [Clear cache](#)





[Scan Another >>](#)

	Server	Test time	Grade
1	124.225.182.171 Ready	Fri, 21 Dec 2018 12:06:29 UTC Duration: 175.439 sec	B
2	124.225.182.172 Ready	Fri, 21 Dec 2018 12:09:25 UTC Duration: 158.366 sec	B

查看测试分析结果：（该图为服务器支持的加解密套件）



Cipher Suites

# TLS 1.2 (server has no preference)	
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f) WEAK	128
TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (0x41) WEAK	128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013) ECDH sect571r1 (eq. 15360 bits RSA) FS	128
TLS_RSA_WITH_AES_128_CBC_SHA256 (0x3c) WEAK	128
TLS_RSA_WITH_AES_128_GCM_SHA256 (0x9c) WEAK	128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027) ECDH sect571r1 (eq. 15360 bits RSA) FS	128
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) ECDH sect571r1 (eq. 15360 bits RSA) FS	128
TLS_RSA_WITH_AES_256_CBC_SHA (0x35) WEAK	256
TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (0x84) WEAK	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) ECDH sect571r1 (eq. 15360 bits RSA) FS	256
TLS_RSA_WITH_AES_256_CBC_SHA256 (0x3d) WEAK	256
TLS_RSA_WITH_AES_256_GCM_SHA384 (0x9d) WEAK	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) ECDH sect571r1 (eq. 15360 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) ECDH sect571r1 (eq. 15360 bits RSA) FS	256
# TLS 1.1 (server has no preference)	
# TLS 1.0 (server has no preference)	
# SSL 3 (server has no preference)	

如果设备只连接固定的服务器，为了减少内存的使用，可选择较简单的加解密算法，以降低内存和时间的消耗。上图中，可尽量选择 TLS_RSA_WITH_AES_128_CBC_SHA 加解密套件。密钥交换算法为 RSA，对称加密为 AES，密钥长度为 128，模式为 CBC 模式，哈希算法为 SHA 算法。通过修改 config-xr-mini-cliserv.h 中的宏可修改设备端支持的加解密套件。当设备端和服务器进行握手时，设备端会将支持的加解密套件列表发送给服务器，服务器会在列表中选择第一个匹配到的加解密套件来进行后续的加解密通讯。根据上图，可以选择宏 MBEDTLS_RSA_C，MBEDTLS_AES_C，MBEDTLS_CIPHER_MODE_CBC，MBEDTLS_SHA1_C，MBEDTLS_KEY_EXCHANGE_RSA_ENABLED。

下图为另外一台服务器支持的加解密套件列表。

Cipher Suites

# TLS 1.2 (suites in server-preferred order)	
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) ECDH secp521r1 (eq. 15360 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) ECDH secp521r1 (eq. 15360 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) ECDH secp521r1 (eq. 15360 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) ECDH secp521r1 (eq. 15360 bits RSA) FS	128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027) ECDH secp521r1 (eq. 15360 bits RSA) FS	128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013) ECDH secp521r1 (eq. 15360 bits RSA) FS	128

上图的服务器只支持 ECDHE_RSA 秘钥交换算法，可选择 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA 加解密套件。因为上面的服务器只支持 ECDHE_RSA，所以需要开宏 MBEDTLS_ECDH_C 和 MBEDTLS_KEY_EXCHANGE_ECDHE_RSA_ENABLED，由于需要支持 ECDH secp521r1，所以还需要开宏 MBEDTLS_ECP_C 和 MBEDTLS_ECP_DP_SECP521R1_ENABLED。