

HTTP/HTTPS 开发指导手册

版本：V1.2.0

发布日期：2022/10/14

服务与支持

如果您有任何关于模组产品及产品手册的评论、疑问、想法，或者任何无法从本手册中找到答案的疑问，请通过以下方式联系我们。



中移物联网有限公司

OneMO官网: onemo10086.com

邮箱: SmartModule@cmiot.chinamobile.com

客户服务热线: 400-110-0866

微信公众号: CMOneMO



中国移动
China Mobile

文档声明

注意

本手册描述的产品及其附件特性和功能，取决于当地网络设计或网络性能，同时也取决于用户预先安装的各种软件。由于当地网络运营商、ISP，或当地网络设置等原因，可能也会造成本手册中描述的全部或部分产品及其附件特性和功能未包含在您的购买或使用范围之内。

责任限制

除非合同另有约定，中移物联网有限公司对本文档内容不做任何明示或暗示的声明或保证，并且不对特定目的适销性及适用性或者任何间接的、特殊的或连带的损失承担任何责任。

在适用法律允许的范围内，在任何情况下，中移物联网有限公司均不对用户因使用本手册内容和本手册中描述的产品而引起的任何特殊的、间接的、附带的或后果性的损坏、利润损失、数据丢失、声誉和预期的节省而负责。

因使用本手册中所述的产品而引起的中移物联网有限公司对用户的最大赔偿（除在涉及人身伤害的情况中根据适用法律规定的损害赔偿外），不应超过用户为购买此产品而支付的金额。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。公司保留随时修改本手册中任何信息的权利，无需进行提前通知且不承担任何责任。

商标声明



为中国移动注册商标。

本手册和本手册描述的产品中出现的其他商标、产品名称、服务名称和公司名称，均为其各自所有者的财产。

进出口法规

出口、转口或进口本手册中描述的产品（包括但不限于产品软件和技术数据），用户应遵守相关进出口法律和法规。

隐私保护

关于我们如何保护用户的个人信息等隐私情况，请查看相关隐私政策。

操作系统更新声明

操作系统仅支持官方升级；如用户自己刷非官方系统，导致安全风险和损失由用户负责。

固件包完整性风险声明

固件仅支持官方升级；如用户自己刷非官方固件，导致安全风险和损失由用户负责。

版权所有©中移物联网有限公司。保留一切权利。

本手册中描述的产品，可能包含中移物联网有限公司及其存在的许可人享有版权的软件，除非获得相关权利人的许可，否则，非经本公司书面同意，任何单位和个人不得擅自摘抄、复制本手册内容的部分或全部，并以任何形式传播。



中国移动
China Mobile

关于文档

修订记录

版本	描述
V1.0.0	初版
V1.1.0	更新手册适用范围； 新增API说明示例信息； 新增编程注意事项。
V1.2.0	更新手册适用范围。



中国移动
China Mobile

目录

服务与支持..... ii

文档声明..... iii

关于文档..... v

1. 概述..... 7

 1.1. 适用范围..... 7

2. 功能说明..... 8

3. API说明..... 9

4. 应用指导..... 12

 4.1. HTTP普通连接示例..... 13

 4.2. HTTPS无认证连接示例..... 14

 4.3. HTTPS单向认证连接示例..... 15

 4.4. HTTPS双向认证连接示例..... 16

5. 编程设计注意..... 17

6. 附录..... 18



中国移动
China Mobile

1. 概述

本文档介绍了OpenCPU SDK的HTTP功能，以便于用户参照文档和SDK中的示例程序完成应用程序编写。

1.1. 适用范围

Table 1. 适用模组

模组系列	模组子型号
MN316	MN316-DBRS/MN316-DLVS
MN316-S	MN316-S-DLVS
M5310-E	M5310-E-BR/M5310-E-LR
ML302	ML302-ANLM/ML302-CNLM/ML302-DNLM/ML302-ENLM/ML302-FNLM/ML302-GNLM/ML302-HNLM/ML302-INLM/ML302-JNLM/ML302-KNLM/ML302-LNLM/ML302-MNLM/ML302-PNLM/ML302-QNLM
ML305	ML305-DNLM/ML305-RNLM/ML305-SNLM/ML305-TNLM
ML307A	ML307A-DCLN/ML307A-DSLN

2. 功能说明

超文本传输协议（Hyper Text Transfer Protocol，HTTP）是一个请求-响应协议，通常采用TCP/IP通信协议传递数据。OpenCPU SDK提供的HTTP API接口最大支持创建4路HTTP实例，且支持GET、POST和PUT等常用请求方法。



中国移动
China Mobile

3. API说明

OpenCPU SDK提供一套完整的HTTP API接口，并在`cm_http.h`头文件中提供详细接口定义及说明。

常用HTTP参数结构体如下。

```
/**HTTP可配参数*/
typedef struct {
    uint8_t ssl_enable; //是否使用HTTPS
    int32_t ssl_id; //ssl索引号
    uint8_t cid; //PDP索引
    uint8_t conn_timeout; //连接超时时间
    uint8_t rsp_timeout; //响应超时时间
    uint8_t dns_priority; //dns解析优先级 0: 使用全局优先级。1: v4优先。2: v6优先
}cm_httpclient_cfg_t;

/**HTTP 同步接口输入参数*/
typedef struct {
    cm_httpclient_request_type_e method; //请求类型
    const uint8_t *path; //请求路径
    uint32_t content_length; //数据长度
    uint8_t *content; //数据
}cm_httpclient_sync_param_t;

/**HTTP 同步接口响应数据*/
typedef struct {
    uint32_t response_code; //请求成功时的响应结果码
    uint32_t response_header_len; //响应报头长度
    uint32_t response_content_len; //响应消息体长度
    uint8_t *response_header; //响应报头（内部分配空间，使用完后可调用cm_httpclient_sync_free_data释放）
    uint8_t *response_content; //响应消息体（内部分配空间，使用完后可调用cm_httpclient_sync_free_data释放）
}cm_httpclient_sync_response_t;
```

常用HTTP接口定义如下。

```
/**
 * @brief 创建客户端实例
 *
 * @param[in] url 服务器地址（服务器地址url需要填写完整，例如（服务器url仅为格式示例）"https://39.106.55.200:80"）
 * @param[in] callback 客户端相关回调函数
 * @param[out] handle 实例句柄
 *
 * @return 0 成功/其他失败（见cm_httpclient_ret_code_e）
 *
 * @details 创建客户端实例
 */
cm_httpclient_ret_code_e cm_httpclient_create(const uint8_t *url, cm_httpclient_event_callback_func callback,
cm_httpclient_handle_t *handle);

/**
 * @brief 删除客户端实例
 *
 * @param[in] handle 客户端实例句柄
 */
```

```

*
* @return 0 成功/其他失败 ( 见cm_httpclient_ret_code_e )
*
* @details 本接口会将close socket ( HTTP ) 操作发送至eloop模块中让其执行close
socket操作。本接口返回成功代表操作已发送至eloop中，不代表已完成实例删除操作。用户连续两次调用本接口时建议中间保证
100ms的延时。
*/
cm_httpclient_ret_code_e cm_httpclient_delete(cm_httpclient_handle_t handle);

/**
* @brief 检测是否在执行请求
*
* @param [in] client 客户端实例句柄
*
* @return false 未执行请求/ true 正在执行请求
*
* @details 检测是否在执行请求
*/
bool cm_httpclient_is_busy(cm_httpclient_handle_t handle);

/**
* @brief 客户端参数设置
*
* @param [in] handle 客户端实例句柄
* @param [in] cfg 客户端配置
*
* @return 0 成功/其他失败 ( 见cm_httpclient_ret_code_e )
*
* @details
客户端参数设置，创建实例后设置,请求过程中不可设置；服务器应答重定向信息的情况下，采用cm_httpclient_set_cfg()配置的参数
连接重定向的服务器
*/
cm_httpclient_ret_code_e cm_httpclient_set_cfg(cm_httpclient_handle_t handle, cm_httpclient_cfg_t cfg);

/**
* @brief 终止http连接
*
* @param [in] handle 客户端实例句柄
*
* @return
*
* @details 终止http连接
*/
void cm_httpclient_terminate(cm_httpclient_handle_t handle);

/**
* @brief 发送请求(同步接口)
*
* @param [in] handle 客户端实例句柄
* @param [in] param 输入参数 ( 见cm_httpclient_sync_param_t )
* @param [out] response 响应结果 ( 见cm_httpclient_sync_response_t )
*
* @return 见cm_httpclient_ret_code_e
*
* @details 发送http请求同步接口，只可用于非chunk模式发送，需先创建实例并完成相关参数设置，
* 响应结果中的数据未内部动态分配空间，使用完后可通过cm_httpclient_sync_free_data接口释放，


```

```

*下次请求时也会自动释放。
*注意：HTTP模块底层依赖asocket、async_dns和eloop模块，使用HTTP功能时应先初始化前述模块
*/
cm_httpclient_ret_code_e cm_httpclient_sync_request(cm_httpclient_handle_t handle, cm_httpclient_sync_param_t param,
cm_httpclient_sync_response_t *response);

/**
* @brief 释放响应数据(同步接口)
*
* @param [in] handle 客户端实例句柄
*
* @return
*
* @details 释放响应数据。
*/
void cm_httpclient_sync_free_data(cm_httpclient_handle_t handle);

```

 **Note:** 不同发布SDK版本中，API中部分信息可能存在差异，请以当前SDK版本中cm_http.h头文件说明为准。



中国移动
China Mobile

4. 应用指导

本章主要介绍常见应用示例。

 **Note:** M5310-E暂不支持HTTPS；MN316暂不支持HTTPS双向认证连接。



中国移动
China Mobile

4.1. HTTP普通连接示例

本节以GET请求为例，提供HTTP普通连接操作下的代码示例。

```
if (NULL == client[0])
{
    ret = cm_httpclient_create(TEST_HTTP_SERVER, NULL, &client[0]);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
    {
        cm_printf("cm_httpclient_create() error!\r\n");
        return;
    }

    cm_httpclient_cfg_t client_cfg;
    client_cfg.ssl_enable = false;
    client_cfg.ssl_id = 0;
    client_cfg.cid = 0;
    client_cfg.conn_timeout = HTTPCLIENT_CONNECT_TIMEOUT_DEFAULT;
    client_cfg.rsp_timeout = HTTPCLIENT_WAITRSP_TIMEOUT_DEFAULT;
    client_cfg.dns_priority = 0;
    ret = cm_httpclient_set_cfg(client[0], client_cfg);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
    {
        cm_printf("cm_httpclient_set_cfg() error!\r\n");
        return;
    }
}

cm_httpclient_sync_response_t response = {};
cm_httpclient_sync_param_t param = {HTTPCLIENT_REQUEST_GET, TEST_HTTP_PATH, 0, NULL};

ret = cm_httpclient_sync_request(client[0], param, &response);

if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
{
    cm_printf("cm_httpclient_sync_request() error! ret is %d\r\n", ret);
    return;
}
else
{
    cm_printf("response_code is %d\r\n", response.response_code);
    cm_printf("response_header_len is %d\r\n", response.response_header_len);
    cm_printf("response_content_len is %d\r\n", response.response_content_len);
}

cm_httpclient_sync_free_data(client[0]);
```

4.2. HTTPS无认证连接示例

本节以GET请求为例，提供HTTPS无认证连接操作下的代码示例。

```
if (NULL == client[0])
{
    ret = cm_httpclient_create(TEST_HTTPS_SERVER, NULL, &client[0]);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
    {
        cm_printf("cm_httpclient_create() error!\r\n");
        return;
    }

    cm_httpclient_cfg_t client_cfg;
    client_cfg.ssl_enable = true;
    client_cfg.ssl_id = 0;
    client_cfg.cid = 0;
    client_cfg.conn_timeout = HTTPCLIENT_CONNECT_TIMEOUT_DEFAULT;
    client_cfg.rsp_timeout = HTTPCLIENT_WAITRSP_TIMEOUT_DEFAULT;
    client_cfg.dns_priority = 0;
    ret = cm_httpclient_set_cfg(client[0], client_cfg);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
    {
        cm_printf("cm_httpclient_set_cfg() error!\r\n");
        return;
    }
}

int tmp = 0;
cm_ssl_setopt(0, CM_SSL_PARAM_VERIFY, &tmp);

cm_httpclient_sync_response_t response = {};
cm_httpclient_sync_param_t param = {HTTPCLIENT_REQUEST_GET, TEST_HTTPS_PATH, 0, NULL};

ret = cm_httpclient_sync_request(client[0], param, &response);

if (CM_HTTP_RET_CODE_OK != ret || NULL == client[0])
{
    cm_printf("cm_httpclient_sync_request() error! ret is %d\r\n", ret);
    return;
}
else
{
    cm_printf("response_code is %d\r\n", response.response_code);
    cm_printf("response_header_len is %d\r\n", response.response_header_len);
    cm_printf("response_content_len is %d\r\n", response.response_content_len);
}

cm_httpclient_sync_free_data(client[0]);
```

4.3. HTTPS单向认证连接示例

本节以GET请求为例，提供HTTPS单向认证连接操作下的代码示例。

```
if (NULL == client[2])
{
    ret = cm_httpclient_create("https://www.baidu.com", NULL, &client[2]);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[2])
    {
        cm_printf("cm_httpclient_create() error!\r\n");
        return;
    }

    cm_httpclient_cfg_t client_cfg;
    client_cfg.ssl_enable = true;
    client_cfg.ssl_id = 2;
    client_cfg.cid = 0;
    client_cfg.conn_timeout = HTTPCLIENT_CONNECT_TIMEOUT_DEFAULT;
    client_cfg.rsp_timeout = HTTPCLIENT_WAITRSP_TIMEOUT_DEFAULT;
    client_cfg.dns_priority = 1;
    ret = cm_httpclient_set_cfg(client[2], client_cfg);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[2])
    {
        cm_printf("cm_httpclient_set_cfg() error!\r\n");
        return;
    }

    int tmp = 1;
    cm_ssl_setopt(2, CM_SSL_PARAM_VERIFY, &tmp);
    cm_ssl_setopt(2, CM_SSL_PARAM_CA_CERT, (char*)http_ca);
}

cm_httpclient_sync_response_t response = {};
cm_httpclient_sync_param_t param = {HTTPCLIENT_REQUEST_GET, "/", 0, NULL};

ret = cm_httpclient_sync_request(client[2], param, &response);

if (CM_HTTP_RET_CODE_OK != ret || NULL == client[2])
{
    cm_printf("cm_httpclient_sync_request() error! ret is %d\r\n", ret);
    return;
}
else
{
    cm_printf("response_code is %d\r\n", response.response_code);
    cm_printf("response_header_len is %d\r\n", response.response_header_len);
    cm_printf("response_content_len is %d\r\n", response.response_content_len);
}

cm_httpclient_sync_free_data(client[2]);
```

4.4. HTTPS双向认证连接示例

本节以GET请求为例，提供HTTPS双向认证连接操作下的代码示例。

```
if (NULL == client[3])
{
    ret = cm_httpclient_create("https://120.27.12.119:10443", NULL, &client[3]);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[3])
    {
        cm_printf("cm_httpclient_create() error!\r\n");
        return;
    }

    cm_httpclient_cfg_t client_cfg;
    client_cfg.ssl_enable = true;
    client_cfg.ssl_id = 3;
    client_cfg.cid = 0;
    client_cfg.conn_timeout = HTTPCLIENT_CONNECT_TIMEOUT_DEFAULT;
    client_cfg.rsp_timeout = HTTPCLIENT_WAITRSP_TIMEOUT_DEFAULT;
    client_cfg.dns_priority = 0;
    ret = cm_httpclient_set_cfg(client[3], client_cfg);

    if (CM_HTTP_RET_CODE_OK != ret || NULL == client[3])
    {
        cm_printf("cm_httpclient_set_cfg() error!\r\n");
        return;
    }

    int tmp = 2;
    cm_ssl_setopt(3, CM_SSL_PARAM_VERIFY, &tmp);
    cm_ssl_setopt(3, CM_SSL_PARAM_CA_CERT, (char*)test_server_ca);
    cm_ssl_setopt(3, CM_SSL_PARAM_CLI_CERT, (char*)test_client_ca);
    cm_ssl_setopt(3, CM_SSL_PARAM_CLI_KEY, (char*)test_client_key);
}

cm_httpclient_sync_response_t response = {};
cm_httpclient_sync_param_t param = {HTTPCLIENT_REQUEST_GET, "/", 0, NULL};
ret = cm_httpclient_sync_request(client[3], param, &response);

if (CM_HTTP_RET_CODE_OK != ret || NULL == client[3])
{
    cm_printf("cm_httpclient_sync_request() error! ret is %d\r\n", ret);
    return;
}
else
{
    cm_printf("response_code is %d\r\n", response.response_code);
    cm_printf("response_header_len is %d\r\n", response.response_header_len);
    cm_printf("response_content_len is %d\r\n", response.response_content_len);
}
cm_httpclient_sync_free_data(client[3]);
```


5. 编程设计注意

HTTP功能在使用时应注意以下几点：

- 先初始化asocket、async_dns和eloop模块；
- 创建实例后再调用`cm_httpclient_set_cfg()`；该接口为客户端参数设置接口，在HTTP请求过程中不可调用；
- 创建实例并完成参数设置后再调用`cm_httpclient_sync_request()`；该接口为发送请求同步接口，只可用于非chunk模式发送；
- 调用`cm_httpclient_create()`时，其入参url中不能省略协议信息（http/https）。



中国移动
China Mobile

6. 附录

Table 2. 缩略语

缩写	英文全称	中文解释
API	Application Programming Interface	应用程序编程接口
HTTP	Hyper Text Transfer Protocol	超文本传输协议
HTTPS	Hyper Text Transfer Protocol over Secure Socket Layer	超文本传输安全协议
SSL	Secure Sockets Layer	安全套接字协议

