深圳市华曦达科技股份有限公司		文 档 编 号	版本号	密级
		文档编号	V2.11	机密
文档名称	设备管理系统通信协议		日期	2016 -11-21

设备管理系统通信协议

文档作者：	**金志建**	**日期：**	**2016-11-21**
项目经理：		**日期：**	
审　核：		**日期：**	
批　准：		**日期：**	

深圳市华曦达科技股份有限公司

文档历史发放及记录

序号	变更（+/-）说明	作者	版本号	日期	批准
1	初始版本	金志建	V1.0	2015-03-26	
2	增加 APP 更新模块、数据传输改为加密版本 V2	金志建	V2.0	2015-04-07	
3	增加了 App 更新状态和结果信息的返回	金志建	V2.1	2015-04-27	
4	增加协议 20，web 获取盒子端内存信息	金志建	V2.2	2015-05-20	
5	增加 DM2016 检测，局域网 OTA 升级，AIS 锁屏功能	金志建	V2.3	2015-05-28	
6	修改了 ota 升级与 app 升级时，获取到的升级信息的格式	金志建	V2.4	2015-06-03	
7	1. OTA 升级增加 Release_note 与 wipe_data 字段 2. 客户端与服务端增加协议 21，服务端发 APP 升级命令	金志建	V2.5	2015-07-02	
8	支持 https 协议	金志建	V2.6	2015-07-10	
9	修改协议 20 逻辑，将收集到的信息组成文件，上传	金志建	V2.7	2015-07-15	
10	修改 getclientinfo 为 boxcheck，名为故障采集	金志建	V2.8	2015-11-03	
11	1. 客户端与服务端增加协议 22，烧录 Mac 与 SN 命令 2. 修改登录命令 1，登陆时，将设备管理应用版本上报 3. Apiinfo 增加 burnmacsn 接口	金志建	V2.9	2016-06-25	
12	1. 采集网络数据时，增加字段，获取 wifi 信号等数据 2. 增加自定义命令，恢复出厂设置 3. 获取日志时，增加设备管理应用运行日志	金志建	V2.10	2016-09-12	
13	增加 4G 模块升级功能	金志建	V2.11	2016-11-21	

目录

文档简要功能及适用范围

1. 文档的简要功能

   设备管理服务系统的交互协议的描述

2. 文档的适用范围

   sdmc 研发人员

1、 使用协议简介

1.1 系统使用 HTTP、HTTPS 及 TCP/IP 协议进行数据通信

1.2 HTTP，HTTPS 协议描述

通过 Http，Https 的 Get 和 Post 方式获取服务器的数据和上传数据至服务器。

1.3 TCP 协议描述

基本格式为： 校验头 + 数据长度 + 数据

校验头为长度 4 的字符串，暂定为"-sdr",

数据长度为 int 型占 4 个 byte，指加密的数据的长度，真实加密后数据的长度可根据该长度算出。

数据为使用 AES 解密后的数据，读取后要先进行解密，然后才能按照协议进行解析。文件传输使用 2 进制，其他命令采用 json 传输。

1.4 获取盒子端应用所对应的服务器地址，访问服务器的地址(ems.xml 文件格式)

http 协议：<Server url="http://api.iottplus.com:8002/v2/apiinfo/"></Server>

https 协议: <Server url="https://api.iottplus.com/v2/apiinfo/"></Server>

apiinfo 获取接口:

http://api.iottplus.com:8002/v2/apiinfo/?model=XXX&mac=XXX&pid=XXX&vid=XXX
&version=x.x.x.x

可通过配置 ems.xml 控制 OTA 升级，APP 升级，RemoteDebug，Collector 服务，AIS 锁屏功能的开启和关闭，默认开启。并且配置 app 更新轮询时间，默认为 24 小时。

具体格式如下：

```
<Server url="http://api.iottplus.com:8002/v2/apiinfo/"
        otaupdate="1"
        appupdate="1"
        remotedebug="1"
        collector="1"
        lockscreen="1"
        appupdate_checktime="60000" (1 分钟)>
</Server> ("1", 代表开启， "0",代表关闭， 不进行配置，默认为"1")
```

1.5 服务器返回的文件格式

发送获取 apiinfo 请求后，服务器返回经过 RSA 加密的字符串，解密后，内容如下：

<apis>
<api name="remotedebug" host="www.iottplus.com" url="www.iottplus.com"
port="6006"/>
<api name="update" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/update/"
port="8002"/>
<api name="appupdate" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/appupdate/" port="8002"/>
<api name="appupdateresult" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/appupdateresult/" port="8002"/>
<api name="updateresult" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/updateresult/" port="8002"/>
<api name="collector" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/collector/" port="8002"/>
<api name="getclientinfo" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/getclientinfo/" port="8002"/>
<api name="lockscreen" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/lockscreen/" port="8002"/>
<api name="burnmacsn" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/burnmacsn/" port="8002"/>
<api name="devicelistener" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/devicelistener/" port="8002"/>
<api name="upgrade4g" host="api.iottplus.com" url="http://api.iottplus.com:8002/v2/upgrade4g /" port="8002"/>
</apis>

1.6 密钥生成与使用描述

初始密钥由前端提供，使用前，使用如下命令在 linux(需安装 openssl)下对私钥进行 pkcs#8 编码，供终端解密使用。

openssl pkcs8 -topk8 -in private.pem -out pkcs8_rsa_private_key.pem -nocrypt

APP 升级模块，使用前端提供的公钥加密，经过 PKCS#8 编码的私钥进行加密解密。

RSA 加密规范： 字符串加密，每次加密 100 字节，解密时，每次解密 128 字节，其中 100 字节为需要的数据。超过 10000 字节的文件，取头 5000 和尾 5000 加密，不然整个文件加密。

1.7 命令类型及协议编号描述

命令名称	命令 id
终端登录	1
服务端回应登录	2
服务端发送 SH 命令	3
终端回应 SH 命令执行结果	4
服务端发起文件获取	5
终端回应文件获取	6
服务端发起文件推送	7
终端回应文件推送	8
服务端命令 OTA 升级	9
终端回应 OTA 升级	10
终端心跳包发起	11
服务端心跳包回应	12
查看当前在线的盒子	13
查看当前在线的盒子的响应	14
连接盒子命令	15
连接盒子响应	16
服务端键值发送	17
服务端发起 MSG 推送	18
终端回应 MSG 推送	19
服务器端发送获取盒子端信息的命令	20
服务端发送 APP 升级的命令	21
服务端发送烧录 Mac 与 SN 命令	22
服务端发送 4G 模块升级的命令	23

2、 OTA 升级与设备管理服务器交互协议描述

2.1 查询升级包信息格式：

http://api.iottplus.com:8002/v2/update/?model=XXXXXX&mac=XXXXXX&pid=XXXXXX&vid=XXX&version=X.X.X.X

返回的 XML 文件的格式(经过 RSA 解密后)：

```
<updateinfo>
  <info>
      <version>4.0.1.41</version>
      <url>![CDATA[stb6507_fuso__V4.0.1.40.bin]]</url>
      <release_note>BASE64 转码的数据</release_note>
      <wipe_data>0</wipe_data><!-- 1 为擦除 data，0 为不擦除 data -->
  </info>
  <info>
      <version>4.0.1.40</version>
      <inc_version>4.0.1.39</inc_version>
      <url>![CDATA[update_4.0.1.40.bin]]</url>
      <release_note>BASE64 转码的数据</release_note>
      <wipe_data>1</wipe_data><!-- 1 为擦除 data，0 为不擦除 data -->
  </info>
</updateinfo>
```

2.2 查询节目表信息

http://www.sdmc.cn/programlist/?model=xxxxx&pid=xxxx&vid=xxxx&sn=xxxx

返回的 XML 文件的格式

```
<programs_update  date="20140107"  url="xxxxxxxxxxxxxxx"></programs_update>
```

2.3 升级结果上传至服务器格式(Get 方式)：

http://api.iottplus.com:8002/v2/updateresult/?mac=XXX&version=X.X.X.X&status=fail

status：success/fail/downloading/downloadfail

Web 端返回：ok/fail

3、 远程调试与设备管理服务器交互协议描述

3.1 MSG 推送(服务端发起)(经过 AES 加密/解密)

服务器端发送：MsgSend{int id; int msgId; int type;(0 表示文字，其他再做扩展) String content, String mac;}

终端回应：MsgSendRes{ int id; int msgId; int state; (0 表示已接收，1 表示已阅读) String mac; }

3.2 登录(终端发起)

客户端发送：loginInfo {int id; String model; String mac; String sn; int pid; int vid; String version;String EMSVersionName}

服务端回应：loginRes {int id; boolean isLoginSuccess; String mac;}

3.3 SH 命令(服务端发起)

服务器端发送：

SHCmd{int id; String cmd; String mac;}

终端回应：

SHCmdRes{int id; int type;（1 表示正常返回，2 表示错误信息）String cmd;}

注：sh 命令执行为异步，故回应不会立刻返回，一些命令也可能没有返回结果。

以下为自定义 SH 命令：

stopcmd：终止正在执行的命令。

factoryreset：恢复出厂设置(重启之后，通过 devicelistener 接口，携带 mac，model，pid，vid，version，result(success/fail)，String cmd(指定命令)参数 post 至 web 端)

3.4 文件获取(服务端发起)

服务器端发送：FileGet{int id; String boxPath; String webPath; String mac;} 注：path 请使用绝对路径

终端回应：FileGetRes{int id; String boxPath; String webPath; long fileSize; (未经 AES 加密的流的长度)}接收到的流为经 AES 加密过的文件流

3.5 文件推送(服务端发起)

服务器端发送：FileSend{int id; String boxPath; String webPath; long fileSize; String mac;}

后面为经 AES 加密过的文件流。

终端回应：FileSendRes{int id; int state; (0 表示成功，1 表示失败) String boxPath; String webPath;}

3.6 OTA 升级(服务端发起)

服务器端发送：OTAUpdate {int id; String mac;}

终端回应：OTARes {int id; int state;(0: 升级失败, 1: 升级中, 2: 升级成功)}

3.7 心跳包(终端发起)

客户端发送：Ping{int id; int state;//保留参数 String mac;}

服务端回应：PingRes {int id; int state;//保留参数 String mac;}

心跳包每隔 10 分钟发送一次，30 秒内未收到回应则认为连接已断开。

3.8 查看当前在线盒子(终端发起)

客户端发送：showbox{int id; string mac; int page;}

服务器响应：Showboxres{int id; String mac; int total; int page;}

3.9 终端连接盒子(终端发起)

客户端发送：connect{int id; String mac;}

服务器响应：onnectres{int id; string result; string desc;}

3.10 键值发送(服务端发起)

服务器端发送：Keycode{int id; int action; (按下为 0，弹起为 1) int keycode;}

3.11 获取客户端信息(服务端发起)

服务器端发送：BoxCheck {int id; String cmd; String mac; String action; String param;}

Web 端发送 BoxCheck 命令，经服务器至盒子端，获取盒子端信息与操作盒子端。

服务器端：BoxCheck = '{"id":20,"cmd":"boxcheck","mac":"' + mac + '",
"guid":"' +guid+ '","action":"' +action+ '","param":"' +param+ '","user":"sdmcadmin",
"passwd":"'+password+'","user-agent":"sdmcems/php"}'

action: {"getboxinfo", "getapplist", "checknetwork", "getboxmemoy", "ping", "getlog"}

param:当 action 为 ping 时， param 为 ip 地址，多个 ip 地址以逗号隔开！

1. getboxinfo: boxcheck_boxinfo.xml

```
<boxinfo>
    <mac></mac>
    <sn></sn>
    <pid></pid>
    <vid></vid>
    <model></model>
    <version></version>
    <kernel_version></kernel_version>
    <android_version></android_version>
    <sdk_version></sdk_version>
    <cpu_usage></cpu_usage>
   <memory_total></memory_total>   <memory_free></memeory_free>
   <data_size></data_size>   data_free></data_free>
   <cache_size></cache_size>
   <cache_free></cache_free>
   <system_size></system_size>
   <system_free></system_free>
   <inner_size></inner_size>
   <inner_free></inner_free>
   <inner_appsize></inner_appsize>
   <inner_cachesize></inner_cachesize>
</boxinfo>
```

2. getapplist: boxcheck_appinfo.xml

```
<appinfos>
<appinfo>
    <name></name>
    <packagename></packagename >
    <location></location>
    <cache_size></cache_size>  <!-- 缓存大小 -->
    <data_size></data_size> <!-- 数据大小 -->
    <app_size></app_size> <!-- 程序大小 -->
</appinfo>
……..
</appinfos>
```

3. checknetwork: boxcheck_network.xml

```
<networks>
<network>
    <name></name>
    <status></status> <!—up or down -->
    <ip></ip>
    <mac></mac>  <!-- 以太网 Mac 默认为盒子 mac，wifi 连接时，会上报无线网 mac -->
    <gateway></gateway> <!-- 默认网关，以太网的网关，如果不为以太网，则为空 -->
```

<netmask ></netmask> <!-- 子网掩码 -->

<dns></dns> <!—网络 DNS -->

<wifi_ssid></wifi_ssid><!—无线网称 --><!-- 以下字段仅在 wifi 时有值，否则为空 -->

<wifi_bssid></wifi_bssid><!-- 无线网 Bssid -->

<wifi_linkspeed></wifi_linkspeed><!-- 无线网连接数度 -->

<wifi_rssi></wifi_rssi><!-- 无线网信号强度 -->

<wifi_security></wifi_security><!-- 无线网加密方式 -->

</network>……<networks>

4. ping: boxcheck_ping.txt
   10.10.121.62:success/fail
   ip:success/fail
5. getlog: logcat.txt, traces.txt, emsrun.log

客户端回应：将 model,pid,vid,version,mac,id,action 信息 post 至 web 端。

其中文档经过 AES 加密。

返回值：成功(ok)，失败(fail)

3.12 APP 升级命令(服务端发起)

服务器端发送：AppUpdate {int id, String mac}

如 {"id":21,"mac":"222222222222"}

3.13 烧录 Mac 与 SN(服务端发起)

服务器端发送：BurnMacOrSn{int id, String mac}

如{"id":22,"mac":"222222222222"}

当 Web 端下发了该指令，终端主动请求 burnmacsn 的 API 接口(Get 方式):

请求地址：http://api.iottplus.com:8002/v2/burnmacsn/?model=XXX&mac=XXX&pid=XXX&vid=XXX&version=x.x.x.x

服务端返回内容格式(经过 RSA 解密后)：

<burninfo>
<info>
<id>21</id>
<mac>222222222222</mac>
<sn>oldsn</sn>
<newmac>888888888888</newmac><!-- 可为空，当不为空且合法时，烧录完毕之后，会重新登录 -->
<newsn>newsn</newsn><!-- 可为空,当不合法时，不做处理 -->
<createtime>0</createtime>
</info></burninfo>

烧录结果上报(Http POST 方式)：

接口：http://api.iottplus.com:8002/v2/burnmacsn/；参数 model,pid,vid,mac(烧录前的 mac),version,id,result(0,成功；-1,失败),errormessage(当 result 为-1 时，该值为错误描述，如 result=0，该值为空)

3.14 升级 4G 模块命令

服务器端发送：update4G {int id, String mac, String modelid}

如 {"id":23,"mac":"222222222222"," modelid":""}

// modelid ：WiFi_AP， LTE， Remote, Bluetooth

当 Web 端下发了该指令，终端主动请求 upgrade4g 的 API 接口(Get 方式)

具体请求格式，请参见 8. 4G 模块升级功能；

4、 信息采集与设备管理服务器交互协议描述

4.1 APP 使用数据存储格式(AppInfo.xml)

```
<datacollection>
   <appinfo
         mac = "xxx";                    //设备 MAC 地址
         PID = "xxx";                    //产品 ID
         CID = "xxx";                    //客户 ID
         appName = "xxx";                //应用名
         packageName = "xxx";            //包名
         className = "xxx";              //类名
         startTime = "xxx";              //开始时间
         endTime = "xxx";                //结束时间
         duration = "xxx";               //持续时间(秒)
         IP = "xxx";                     //访问 IP/盒子端 IP
     ></appinfo></datacollection>
```

4.2 DTV 使用数据存储格式(DTVInfo.xml)

```
<datacollection>
   <dtvinfo
         mac = "xxx";                    //设备 MAC 地址
         PID = "xxx";                    //产品 ID
         CID = "xxx";                    //客户 ID
         channelInfo = "xxx";            //频道信息
         programeInfo = "xxx";           //节目信息
         startTime = "xxx";              //开始时间
         endTime = "xxx";                //结束时间
         duration = "xxx";               //持续时间(秒)
         IP = "xxx";                     //访问 IP/盒子端 IP
></dtvinfo></datacollection>
```

4.3 网络视频数据存储格式(OTTInfo.xml)

```
<datacollection>
   <ottinfo
         mac = "xxx";                    //设备 MAC 地址
         PID = "xxx";                    //产品 ID
         CID =   "xxx";                  //客户 ID
         url= "xxx";                     //播放地址
         startTime = "xxx";              //开始时间
         endTime = "xxx";                //结束时间
         duration = "xxx";               //持续时间(秒)
         IP = "xxx";                     //访问 IP/盒子端 IP
></ottinfo></datacollection>
```

4.4 数据采集数据提交接口

4.4.1 接口

APPInfo：

http://api.iottplus.com:8002/v2/collector/?type=app

DTVInfo:

http://api.iottplus.com:8002/v2/collector/?type=dtv

OTTInfo:

http://api.iottplus.com:8002/v2/collector/?type=video

参数及返回值

HTTP 请求方式： post

支持格式： 文本(经过 AES 加密)

请求参数： mac(设备以太网 mac 地址)， uploadfile(上传文件)

返回值： success(成功)， fail(失败)

5、 App 更新与设备管理服务器交互协议描述

5.1 请求与返回数据的协议

盒子端请求：
http://api.iottplus.com:8002/v2/appupdate/?model=XXXXXX&mac=XXXXXX&pid=XXXXXX&vid=XXX&version=X.X.X.X

服务器端回应(经过 RSA 解密后)：

```
<appupadte>
        <app type="apk" name="a.apk" packagename="com.sdmc.a" version="100" >
                <url><![CDATA[http://10.10.1.1/a.apk&]]></url>
        </app>
        <app type="zip" name="b.bin" packagename="com.sdmc.b" version="101" >
                <url><![CDATA[http://10.10.1.1/b.bin]]></url>
        </app>
</appupdate>
```

服务器反馈的信息格式如上所示。每个 APP 更新信息由一个 app 标签记录，每个 app 标签包含以下属性：

type：apk 或 zip，apk 表示其只是一个 apk，下载下来后可直接安装；zip 表示其为一系列关联文件，里面必定包含一个脚本文件，解压 zip 包后执行脚本文件即可。

name：更新 APP 的名称，仅作为一个文件名使用，无特殊用途。

packagename：更新 APP 的包名，作为一个 APP 的唯一标示符。

version:APP 的版本号，普通 APK 即其版本号，对于特殊的 zip 文件，version 需由上传者慎重决定。客户端在 system/sdmc/zipupdate.log 文件中记录着升级的版本号和文件名。

url：APP 或 Zip 下载地址。

5.2 Zip 文件格式解析

对于一些特殊更新信息，如需替换 so 文件等，需使用特定的文件格式进行存储。目前该文件格式定义如下：

1）将这些文件压缩成一个 zip 文件；

2）对该文件使用 RSA 加密，通过加密工具进行压缩和加密操作；

3）在该文件前添加长度为 1024 字节的文件头，如下图

Packagename(256byte)	Version(32byte)	Length(8byte)	Reserved

4）在文件尾添加长度为 1024 字节的文件尾，记录 zip 文件的 MD5 码。

5）最后生成一个.bin 文件。

客户端下载该 Zip 包后，进行逆向的解析过程，获取最终的安装脚本和安装包。

脚本文件中支持的 shell 命令： mv、rm、cp、chmod、chown

install：安装 apk，如 install a.apk;

mv、rm、cp、chmod：均同 shell 命令；

#：注释；

start、stop：分别处于文件头和文件尾，用于表示脚本开始和结束;

在编写脚本时， 如果没有写绝对路径，可直接写文件名，默认的路径为 zip 解压后的路径，如： /update/zip/。 rm 命令可增加删除方式， 如 rm -f qq.apk 强制删除一个文件

或 rm -fr /system/ 强制删除整个目录。 请确保这些命令正确，错误的命令得不到执行。

5.3 App 更新状态与结果返回协议

上传参数： model,pid,vid,mac,appname,packagename,version,type,result，resultcode;

appname， packagename，version，均为服务器上对应的 app 或者 zip 的信息。

type 含义： apk 或者 zip

result 与 resultcode 的含义:

result = 0, resultcode = 0,代表正在下载

result = 0, resultcode = -1; 代表下载失败

result = 1,代表成功， resultcode = 1；

result = -1，解析 XML 失败，resultcode = 0；

result = -2, 解密 zip 文件失败， resultcode=0;

result = -3，解析脚本失败 resultcode = 0；

result = -4，执行脚本失败 resultcode = 0；

result = -5. 安装 APK 失败 resultcode 为对应的出错编号！

请求方式： post

如果解析 xml 失败的话，获取不到 appname 等信息，那么 appname="", packagename="", version=0, type="zip";

6、 局域网 OTA 升级工具与终端的交互协议

6.1 命令类型及编号与编号描述

命令名称	命令 id	动作 action
获取盒子端存储目录	1	1： 不升级 10：升级并清除 data 11：升级不清除 data
获取盒子端存储目录回应	2	
发送文件至盒子端命令	3	
发送文件至盒子端回应	4	

6.2 交互协议描述

6.2.1 获取盒子端存储目录

客户端(PC 端)发起：

Json 语句： {"id":1}

服务端(盒子端)回应：

Json 语句： {"id":2,"value":"/update,/storage/ex...."}

6.2.2 发送文件至盒子端

客户端(PC 端)发起：

Json 语句：

{"id":3,"length":1234,"action":1,"boxpath":"/update/qq.apk"} //仅推送文件

{"id":3,"length":1234,"action":10,"boxpath":"/update/qq.apk"} //OTA 升级，并擦除 data

{"id":3,"length":1234,"action":11,"boxpath":"/update/qq.apk"} //OTA 升级，不擦除 data

服务端(盒子端)回应：

Json 语句： {"id":4,"value":"success"}

7、 IP 锁屏功能

锁屏机制：

系统第一次启动或者网络发生改变，终端会检测网络 IP，并且将信息 post 至服务器端，如果网络地址合法或连不上服务器，则继续使用，不合法，弹出对话框，锁定屏幕。

Post 提交内容： model,pid,vid,mac,ip(aes 加密，并 base64 转码)。

返回值： ok(成功，不锁屏)，fail(锁屏)！

8、 4G 模块升级功能

8.1 获取升级包消息

请求方式：HTTP，Get

请求地址：http://api.iottplus.com:8002/v2/upgrade4g/?model=XXX&mac=XXX&pid=XXX&vid=XXX&version=x.x.x.x&modelid=xxx&modelversion=x.x.x

modelid 包括: WiFi_AP， LTE， Remote, Bluetooth

modelversion 为当前终端该模块的版本号;

服务端返回内容格式(经过 RSA 解密后)：

{“id”: “1”, “name”: “filename”, “modelid” “wifi”,“version”: “1.2.3”,“isenforce”: “true”, “url”: “downloadAdress”,“md5”: “md5”}

8.2 烧录结果上报

请求方式：Http POST

接口：http://api.iottplus.com:8002/v2/upgrade4g/；参数 model,pid,vid,mac,version,modelid,id,modeloldversion,modelcurversion,result

result:success/fail/downloading/downloadfail

id：服务端返回数据中的 id

oldversion：升级前的模块版本号

curversion：升级后的模块版本号