MIIO/OT Linux 客户端发布文档

Date	Version	Changes
2015/05/20	1.0	First Release
2015/06/09	1.1	1.1 Release
2015/06/19	1.2	2.1.2 Release
2015/07/30	1.3	2.1.3 Release
2015/08/24	1.4	2.1.4 Release
2015/10/28	1.5	3.0.0 Release
2015/12/01	1.6	3.1.0 Release
2016/02/25	1.7	3.2.2 Release
2016/05/31	1.8	3.3.1 Release
2016/06/23	1.9	3.3.2 Release
2016/07/05	2.0	3.3.5 Release
2016/08/05	2.1	3.3.6 Release
2017/02/28	2.2	3.3.7 Release
2017/03/29	2.2	3.3.8 Release
2017/04/12	2.3	3.3.9 Release
2017/09/15	2.4	3.3.10 Release
2017/12/20	2.5	3.4.1 Release
2018/03/07	2.6	3.4.2 Release
2018/05/09	2.7	3.4.3 Release
2018/05/15	2.8	3.4.4 Release
2018/08/02	2.9	3.4.5 Release
2018/10/16	3.0	3.4.6Release
2019/06/13	3.1	3.5.3Release

一，版本历史

Version 3.5.3

Version 3.4.6

Version 3.4.5
Version 3.4.4
Version 3.4.3
Version 3.4.2
Version 3.4.1
Version 3.3.10
Version 3.3.9
Version 3.3.8
Version 3.3.7
Version 3.3.6
Version 3.3.5
Version 3.3.3
Version 3.3.1
Version 3.2.2
Version 3.1.0
Version 3.0.0
Version 2.1.4
Version 2.1.3
Version 2.1.2
V1.1
V1.0

二，文件列表

1、mosquito 版本文件列表：

2、raw socket 版本文件列表：

三，已知问题

1，Glibc 库版本不一致问题

2，mosquitto 的配置

四，Copyright & License

一，版本历史

Version 3.5.3

- 新增消息交互统计信息
- 新增 http-dns 功能
- 新增云端消息自动回复功能
- 优化离线重连速度
- 其他 bug 修复

Version 3.4.6

- Helper 脚本修改为 jshon 解析标准 json 字符串
- 修改设备 token 生成方式为 /dev/urandom
- 其他 bug 修复

Version 3.4.5

- 支持特殊字符配置 Wi-Fi 网络

Version 3.4.4

- 去除打点信息

Version 3.4.3

- 修复异常退出 bug
- 增强局域网安全防护
- 其他 bug 修复

Version 3.4.2

- 修复解析消息 ID 错误的 bug
- 修复打点数据统计错误的 bug
- 修复当多个 bt 客户端连接时，miio_client 工作异常的 bug
- 其他安全 bug 修复

Version 3.4.1

- 优化连接国外服务器时间过长的问题；
- 为 mosquitto 版本添加获取 local 功能接口

- 添加绑定结果确认命令；用户进行快连流程后，后台会下发绑定结果确认消息，miio_client 根据该确认消息，广播绑定成功或者失败， 设备可以根据 miio_client 的广播消息来确认是否绑定成功(以 ssid 或者 wifi.conf 作为绑定状态标志可能会造成绑定状态不一致)。miio_client 广播的消息格式为：

  绑定成功：

  ```
  {method":"local.bind","result":"ok"}
  ```

  绑定失败：

  ```
  {method":"local.bind","result":"fail"}
  ```

  目前后台默认不会下发该命令，如果需要，请联系后台同事添加。

Version 3.3.10

- 修复当用户 socket 异常断开时，频繁打印 send error 的 bug；
- 添加国际化支持；
- 修复弱网络唤醒时，在极端情况下，miio 异常退出的 bug；
- 修复 mosquitto 版本，命令下发两次的 bug
- 引入 miio_agent 模块，主要作用是在 miio_client 和用户进程之间添加一个消息注册和分发模块，防止每次收到命令都唤醒所有连接的用户进程。miio_agent 是开源模块，用户可以通过 github 获得：https://github.com/MiEcosystem/miio_linux_open

  具体使用方式，请参考《MIIO/OT Linux 客户端软件架构与接口 3.3.10》

Version 3.3.9

- 修复下发时区信息为空时，造成时区链接错误的 bug；
- 添加配置打印级别的接口，各个用户程序可以根据需要进行解析；
- 精简打印 log；
- 当用户 socketfd 异常时，可能会同时检测到多个 revents，比如 POLLIN|POLLERR 或者 POLLOUT|POLLERR 等，这时如果只解析 POLLIN 或者 POLLOUT，就会早造成逻辑上的 bug。修复了这个 bug。

Version 3.3.8

- 修复重放攻击等安全漏洞；
- 添加时区配置的接口；
- 修复 mdns 异常输入造成程序崩溃的 bug；

Version 3.3.7

- 修复 getaddrinfo 域名解析的一个 bug；

- 增加命令行选项可以指定连接的服务器；
- Keep alive 从 15 秒改为 30 秒；
- 修复扫描不到隐藏 WiFi 的 bug；
- 增加查询本地时间接口 `local.query_time`；
- 增加查询本地状态接口 `local.query_status`；
- 增加查询本地域名接口 `local.query_country`；
- 安全修复，快连完成之后不再响应本地 `token` 请求；
- 解决当同时产生多个 `requst` 消息时，`json` 串解析错误的 `bug`；
- 增大可同时连接的 `client` 个数，目前最大连接数为 `20`；
- 增加蓝牙快连功能；
- 优化 `helper` 脚本；
- 少量 `Bug` 修复；

Version 3.3.6

- 有些有 bug 的路由器在上电之后很短的时间内（比如 5s）DNS 解析不正确，把小米云服务器的地址解析成自己的 ip（192.168.1.1）。解决办法是 miio 做了一个服务器 ip 地址的基础校验，如果解析出来的小米云服务器 ip 跟本地路由器在一个局域网里面，认为这次解析是失败的，不停重试；

Version 3.3.5

- 某些极端情况应用会在 100ms 之内连续的发几个 json RPC 给 miio_client, 我们会切分每个 json，分别发给云端；
- OTA 上报的一些文档更新，现在上报 ota_state 和 ota_progress 的格式大致如下：

```
{"id":123,"method":"props","params":{"ota_state":"idle"}}
{"id":123,"method":"props","params":{"ota_progress":80}}
```

Version 3.3.3

- 某生态链产商支持；
- 增加写 log 到某个指定文件功能：

  [-L --logfile=file] output log into file instead of stdout

Version 3.3.1

- 重写 did/key/token 读写代码，现在的实现更灵活，不限制 did/key/token 一定是写在文件里面，也可以存储在 DB 甚至任何设备合适的地方；
- 添加选项支持加密存储的 key：

  [-e --enckey] key(s) are encrypted saved

  使用这个选项启动 miio_client 的话，did/key 中的 key 必须已经是密文存储，miio_client 会先解密，然后再使用解密出来的 key 跟云端通讯。如需细节，可咨询小米相关负责人；

- 添加选项支持 app 和 miio_client 之间通过 socket 通讯的内容进行加密：

  [-E --encdata] data communication are encrypted

  使用这个选项启动 miio_client 的话，第三方应用跟 miio_client 之间的通讯内容都是加密的。如需细节，可咨询小米相关负责人；
- 加上”--enable-timerfd-wrapper”功能，Android NDK API level < 19 支持；
- config_router 命令里面 uid 选项支持；
- 处理 SIGPIPE，否则的话在某些平台上会导致程序异常崩溃退出；
- 从云端下来的“restore”和“reboot”命令转给上层应用处理；用户在手机 app 端解绑设备的时候云端会给设备发送“restore”这个命令；
- 报告 otc.info 的时候，倾向于使用 app 传过来的 ssid，而不是自己动态扫描，这能解决部分中文 ssid 的问题。

Version 3.2.2

- miio data 目录（device.conf 和别的一些数据文件放置的目录）可配置，”[-d --datadir=<path>]”；
- 加上设备休眠/唤醒方面的接口，具体参看软件与接口文档；
- 内部定时器使用 clock_gettime(CLOCK_MONOTONIC)而不是 gettimeofday()。这样做的好处是定时器不会因为外部时间改变而受到影响；
- 相应的，我们现在维护一个本地时间与服务器时间的差值，不再需要写系统时间（settimeofday）的权限；
- 去掉不必要的 getaddrinfo()调用；
- otc.info 包重发优化：当断线的时候，还没发出去的 otc.info 直接丢弃；发送 otc.info 失败会重发，但只有在线的时候重发；
- 断线的时候，所有没有发送成功在队列里面的包都直接丢弃；
- otc.info 包的内容和格式移到 helper.sh，这样便于移植；
- 包头的协议版本数据表示，修正一个大小端的问题；
- 替换内部使用的 MD5 库，原先的库有大小端的问题；
- 服务器申请 did，key 方案的一些加强和优化；
- 新产品（小米电视平台）的支持；
- 修复 POLLHUP 和 POLLERR 的处理；

Version 3.1.0

- 发布文档中增加 Copyright & License 章节；
- 修正短时间内重发 req 的 bug；

- Keep alive 从 10 秒改为 15 秒；
- _otc.info 频率从 10 分钟改为 30 分钟；
- 修正一个 helper script 在某些出错情况下吃 CPU 的问题；
- 加上 base64 encode/decode；
- 加上 SHA1 和 SHA256 的代码；
- 修正一个长时间运行内存泄漏问题；
- 修正一个 PKCS 补齐的问题；
- 加上 AES256 代码；
- 新产品 wifi 音箱的支持；
- 修正 base64 encode/decode 库在 ARM 平台下的一个 bug；
- 添加 mqtt_recv_line 以修正 helper 脚本里面的一个 race 问题，在某些情况下会导致设备即使有 WiFi 也连不上小米云端；

Version 3.0.0

- 增加 Raw socket 接口，用户可以选择以 MQTT 或者 Raw socket 的方式跟 miio_client 进行通信；Raw socket 的方式不用依赖 MQTT，使得移植到别的环境比如 Android 更加容易。
- 增加对小米路由器的支持；
- 增加动态申请 did 和 key 的机制；
- wifi.conf 的检查放入 helper 脚本，更加方便移植；
- 更改启动顺序：先启动 miio_client，然后再启动对应的 helper 脚本，脚本起来之后会通知 miio_client；
- 增加两个小程序 miio_send_line 和 miio_recv_line，用于 helper 脚本和主程序之间的通信；
- 每个包最大发送数据长度从 548 改为 1024；
- 增加通用的 helper 脚本，便于在 PC 上调试；
- 调整_otc.info 上报机制，避免一上电的时候短时间内上报多个_otc.info；
- 确保_otc.info 上报的时候，路由器 MAC 地址是大写；
- 每次起机完成，上报完_otc.info 之后，上报一个 ota_state 属性标记 OTA 状态；
- “miio_client -v” 显示版本号；

Version 2.1.4

- 如果设备连接的路由器是小米路由器，当用户更改 ssid 或者密码，小米路由器会把新的 ssid 和密码发送给设备，设备则不需要重置和重新快连的过程。增强用户体验。详细指南请参看《MIIO/OT Linux 客户端软件架构与接口（外部文档）》第六章；

- 手机本地控制设备 Bug 修复；
- 修复 miio_client 跑在 Ubuntu 下 dash 脚本的一个 Bug；

Version 2.1.3

- 超时机制：3s 超时，3s 没有收到服务器回包，认为超时；失败计数++；失败 3 次，认为服务器连接断，切下一个服务器；
- 重传机制：UDP 链接的时候，每个用户数据都会尝试重传，重传 3 次失败会返回发送方失败；重传的时候使用同一个 id；TCP 链接没有重传。
- UDP 链接为主，TCP 为辅；
- Keep Alive 机制：间隔 10s，miio_client 会向服务器发送 keep alive 包，keepalive 没有收到回包，失败计数++，失败 3 次，认为服务器链接断，切下一个服务器；
- getaddrinfo 返回的列表，打乱顺序，不要每次都从第一个开始；
- _otcinfo 发送成功之后，服务器会回给设备一个可用服务器列表。列表作为一个整体列表存放，并做一个 hash 值；每次返回的列表，比对 hash，不同的话，整体替换。
- mDNS 实现，用于本地局域网手机发现设备；
- 快连的优化：密码错误的时候，miio_client 会重试，重试次数到了之后返回 AP 模式；
- “`miio_client -l/--loglevel=<level>`“ 参数指定程序打印 Log 级别，可用参数：3 (LOG_DEBUG)，2 (LOG_INFO)，1 (LOG_WARNING)，0 (LOG_ERROR) 4 个级别；
- “`miio_client -i/--interval=<int>`“ 参数指定汇报_otc.info 时间间隔，时间单位毫秒；
- 每个数据包最大长度从 1472 改为 548，这也是互联网上比较安全的，不会被分包的 UDP 包长度；
- 软件版本好规则改动：由 2.1.2_**15**061700 改为 2.1.2_**2015**061700；
- 大量稳定性相关 Bug Fix。

Version 2.1.2

- 得到并且维护服务器列表，重连服务器的时候依次切换服务器；
- 合并 miio_misc 和 miio_client，现在只有一个 miio_client 程序；
- miio_misc_helper.sh 重命名 miio_client_helper.sh；
- 设备连接不上 wifi 的时候会不停重试；
- 设备状态广播。（请参考《MIIO/OT Linux 客户端软件架构与接口（外部文档）》，”六， 设备联网状态广播“）；
- 确定软件版本号规则，例子：2.1.2_15061700，（请参考《MIIO/OT Linux 客户端软件架构与接口（外部文档）》，”七，设备固件/软件升级“）；
- 确定 OTA 升级命令格式，并转发相关命令给具体应用；

- 去掉 openssl 依赖；

V1.1

- 手机本地通信支持；
- WiFi AP 快连进度状态广播；
- 只有当设备被重置的时候才改变 Token；
- MIIO/OT 和服务器之间的通信不再基于 Websocket，而是普通的 TCP/UDP；
- 跟服务器之间的心跳包（保活）和时间同步实现；
- JSON 标准规定的 key 和 string 都用双引号扩起来，不是单引号；
- 发送消息提交机制，这样可以并行处理多个消息提交；
- 去掉 libwebsocket 依赖；
- Ping-Pong 机制检测断网；
- mosquitto 去掉 libcares 依赖；

V1.0

- 第一个对外发布版本

二，文件列表

目前 MIIO/OT Linux 客户端提供两个版本，一个基于 mosquito，一个基于 raw socket，请根据您的需要选择对应的版本。

1、mosquito 版本文件列表：

```
$ tree
.
├── etc
│   ├── miio
│   │   ├── device.conf.sample
│   │   └── wifi_start.sh
│   └── mosquitto
│       └── mosquitto.conf
├── usr.mqtt
│   ├── bin
│   │   ├── miio_client
│   │   ├── miio_client_helper.sh
│   │   ├── mosquitto
│   │   ├── mosquitto_pub
│   │   ├── mosquitto_sub
```

```
│   │   └── mqtt_recv_line
│   └── lib
│       ├── libjson-c.so -> libjson-c.so.2.0.1
│       ├── libjson-c.so.2 -> libjson-c.so.2.0.1
│       ├── libjson-c.so.2.0.1
│       ├── libmosquitto.so -> libmosquitto.so.1
│       └── libmosquitto.so.1
└── usr.nomqtt
    ├── bin
    │   ├── miio_client
    │   ├── miio_client_helper.sh
    │   ├── miio_recv_line
    │   └── miio_send_line
    └── lib
        ├── libjson-c.so -> libjson-c.so.2.0.1
        ├── libjson-c.so.2 -> libjson-c.so.2.0.1
        └── libjson-c.so.2.0.1

9 directories, 21 files
```

2、raw socket 版本文件列表：

```
$ tree
.
├── etc
│   └── miio
│       ├── device.conf.sample
│       └── wifi_start.sh
└── usr
    ├── bin
    │   ├── miio_client
    │   ├── miio_client_helper_nomqtt.sh
    │   ├── miio_recv_line
    │   └── miio_send_line
    └── lib
        ├── libjson-c.so -> libjson-c.so.2.0.1
        ├── libjson-c.so.2 -> libjson-c.so.2.0.1
        ├── libjson-c.so.2.0.1
        ├── libm-0.9.33.2.so
        ├── libm.so.0 -> libm-0.9.33.2.so
        ├── libpthread-0.9.33.2.so
        ├── libpthread.so.0 -> libpthread-0.9.33.2.so
        ├── librt-0.9.33.2.so
        └── librt.so.0 -> librt-0.9.33.2.so

5 directories, 15 files
```

（注：nomqtt 和 mqtt 版本是分开打包，不会在一个包里同时出现）

三，已知问题

1，Glibc 库版本不一致问题

miio_linux.le_glibc_armv7-a_cortex-a9_hardvfpv3.tar.gz 里面的可执行文件和库链接的 Glibc 库版本是 2.19，当你的目标板上的 Glibc 比这个版本老的时候，可能会出现不能运行的情况。

解决办法 1：使你的目标板 Glibc 变为 2.19；
解决办法 2：联系小米方面负责人，把你的整个交叉编译环境给他，并且告知详细的使用你们交叉编译环境的步骤和方法，让他在你们的交叉编译环境下为你们生成可执行文件和所需要的库。
解决办法 3：提出需求，让小米方面提供的 miio_client 和其他可执行文件/库基于你的 Glibc 版本。

2，mosquitto 的配置

现在 mosquitto 的配置文件（/etc/mosquitto.conf）很简单，这是为了方便开发与调试，产商可以根据自己需要增加配置与权限控制，参考文档：http://mosquitto.org/man/mosquitto-conf-5.html

四，Copyright & License

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

aes256.c aes256.h are taken from http://literatecode.com/aes256

```
/*
*   Byte-oriented AES-256 implementation.
*   All lookup tables replaced with 'on the fly' calculations.
*
*   Copyright (c) 2007-2011 Ilya O. Levin, http://www.literatecode.com
*   Other contributors: Hal Finney
*
*   Permission to use, copy, modify, and distribute this software for any
*   purpose with or without fee is hereby granted, provided that the above
*   copyright notice and this permission notice appear in all copies.
*
*   THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
*   WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
*   MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
*   ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
*   WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
*   ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
*   OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
*/
```