没钥匙也要拧开BOOTLOADER的锁

闻观行

没钥匙也要拧开BOOTLOADER的锁

闻观行

启动流程

- 不可控过程
  - PBL - SBL - TZ - **aboot** (XBL - ABL)
  - BootROM - **FASTBOOT**

启动流程

- 不可控过程
  - PBL - SBL - TZ - **aboot** (XBL - ABL)
  - BootROM - **FASTBOOT**
- Bootloader是第一个交互窗口
  - load kernel (sig verify)
    - load system (dm-verity)

ABOOT

- fastboot oem unlock

ABOOT

- fastboot oem unlock
  - fastboot **flash** recovery twrp.img
  - fastboot **boot** boot.img
- No Verification

smartisan os

smartian os

- 官方不提供解锁
- rom.zip/firmware-update/emmc_appsboot.mbn
  - ELF32
  - Qualcomm lk
    - git clone git://codeaurora.org/kernel/lk.git

分析代码

- IDA Pro
  - 对比lk，迁移符号信息（sigmake，diaphora）
  - 修正关键结构（字符串，结构指针）的解析结果

处理逻辑

- aboot_init
  - Real Android Bootloader
- aboot_fastboot_register_commands

```
aFlash                  ; "flash:"
cmd_flash_mmc
aErase                  ; "erase:"
cmd_erase_mmc
aReboot                 ; "reboot"
cmd_reboot
aRebootBootload         ; "reboot-bootloader"
cmd_reboot_bootloader
aFastboot+4             ; "boot"
cmd_boot
aContinue               ; "continue"
cmd_continue
aOemUnlock              ; "oem unlock"
cmd_oem_unlock
aOemLock                ; "oem lock"
cmd_oem_lock
aOemVerified            ; "oem verified"
cmd_oem_verified
aOemDeviceInfo          ; "oem device-info"
cmd_oem_devinfo
aPreflash               ; "preflash"
cmd_preflash
aOemEnableCharg         ; "oem enable-charger-screen"
cmd_oem_enable_charger_screen
aOemDisableChar         ; "oem disable-charger-screen"
cmd_oem_disable_charger_screen
aOemSelectDispl         ; "oem select-display-panel"
cmd_oem_select_display_panel
```

```
v0 = result;
a3 = stack_cookie;
v1 = 1;
memcpy(result, &cmd_table, 120);
do
{
  while ( v1 > 4 && !control_flag )
  {
    ++v1;
    v0 += 8;
    if ( v1 == 15 )
      goto LABEL_6;
  }
  v2 = (_BYTE *)*((_DWORD *)v0 + 2);
  ++v1;
  v3 = (void (__fastcall *)(_BYTE *, int, int))*((_DWORD *)v0 + 3);
  v0 += 8;
  fastboot_register(v2, v3);
}
while ( v1 != 0xF );
```

```
aFlash                  ; "flash:"
cmd_flash_mmc
aErase                  ; "erase:"
cmd_erase_mmc
aReboot                 ; "reboot"
cmd_reboot
aRebootBootload         ; "reboot-bootloader"
cmd_reboot_bootloader
aFastboot+4             ; "boot"
cmd_boot
aContinue               ; "continue"
cmd_continue
aOemUnlock              ; "oem unlock"
cmd_oem_unlock
aOemLock                ; "oem lock"
cmd_oem_lock
aOemVerified            ; "oem verified"
cmd_oem_verified
aOemDeviceInfo          ; "oem device-info"
cmd_oem_devinfo
aPreflash               ; "preflash"
cmd_preflash
aOemEnableCharg         ; "oem enable-charger-screen"
cmd_oem_enable_charger_screen
aOemDisableChar         ; "oem disable-charger-screen"
cmd_oem_disable_charger_screen
aOemSelectDispl         ; "oem select-display-panel"
cmd_oem_select_display_panel
```

处理逻辑

- aboot_init
  - aboot_fastboot_register_commands
- fastboot_init
- fastboot_command_loop
- cmd->handle(cmdline, download_base, download_size)
  - cmd_flash_mmc

CMD_FLASH_MMC

```
data = data_0;
sz = a3;
arg = arg_0;
a3a = stack_cookie;
if ( strcmp(arg_0, "security") && !control_flag || get_value(control_flag, 0) && !strcmp(arg, "security") )
{
  fastboot_fail("flash write failure");
LABEL_11:
  result = fastboot_okay("", v12);
  goto LABEL_12;
}
```

```
  data = data_0;
  sz = a3;
  arg = arg_0;
  a3a = stack_cookie;
  if ( strcmp(arg_0, "security") && !control_flag || get_value(control_flag, 0) && !strcmp(arg, "security") )
  {
    fastboot_fail("flash write failure");
LABEL_11:
    result = fastboot_okay("", v12);
    goto LABEL_12;
  }
```

CMD_FLASH_MMC

```
if ( !strcmp(arg, "security") )
{
  is_allow_unlock = verify_security();
  if ( is_allow_unlock )
  {
    sub_F92B258();
    v18 = (_BYTE *)strlen("Unlock Success");
    v19 = screen_display("Unlock Success", v18, 200, 600, 6, 0);
  }
  else
  {
    sub_F92B258();
    v21 = (_BYTE *)strlen("Unlock Fail");
    v19 = screen_display("Unlock Fail", v21, 200, 600, 6, 0);
  }
```

is_allow_unlock = control_flag

VERIFY_SECURITY

- 对factory分区的序列号的签名校验
  - openssl
  - MD5 + RSA1024
  - hash == RSA.decode(security, pubkey)

解锁流程

- 用户提交序列号
- 厂商计算序列号的哈希
- 厂商用私钥给哈希签名得到security.img
- 厂商下发security.img到用户
- 用户使用fastboot刷入到security分区
- 签名校验通过，手机解锁

解锁流程

- 用户提交序列号
- 厂商计算序列号的哈希
- 厂商用私钥给哈希签名得到security.img
- 厂商下发security.img到用户
- 用户使用fastboot刷入到security分区
- 签名校验通过，手机解锁

VERIFY_SECURITY

```
read_method_from_security((int)&v8, 1);       // read from security partition+0x80 (1byte)
if ( v8 == 66 )
{
  rsa_init = (void (__cdecl *)(RSA *))sub_F924994;
}
else if ( v8 == 67 )
{
  rsa_init = (void (__cdecl *)(RSA *))sub_F924898;
}
else
{
  rsa_init = (void (__cdecl *)(RSA *))sub_F924A90;
}
read_serialnolen_from_security((int)(&v8 + 1), (_BYTE *)1);// read security partition+0x81 (1byte)
if ( *(&v8 + 1) <= 20u )
{
  v2 = 0;
  rsa = RSA_new();
  ((void (*)(void))rsa_init)();
```

RSA_INIT

```
v1 = a1;
v2 = BN_bin2bn(m3_n, (_BYTE *)128, a1->n);
v3 = v1->e;
v1->n = v2;
v4 = BN_bin2bn(&m3_e, (_BYTE *)1, v3);
v5 = v1->d;
v1->e = v4;
v6 = BN_bin2bn(&m3_d, (_BYTE *)1, v5);
v7 = v1->p;
v1->d = v6;
v8 = BN_bin2bn(m3_p, (_BYTE *)64, v7);
v9 = v1->q;
v1->p = v8;
v10 = BN_bin2bn(m3_q, (_BYTE *)64, v9);
```

```
v4 = BN_bin2bn(&m3_e, (_BYTE *)1, v3);
v5 = v1->d;
v1->e = v4;
v6 = BN_bin2bn(&m3_d, (_BYTE *)1, v5);
v7 = v1->p;
v1->d = v6;
v8 = BN_bin2bn(m3_p, (_BYTE *)64, v7);
v9 = v1->q;
v1->p = v8;
v10 = BN_bin2bn(m3_q, (_BYTE *)64, v9);
```

$n = p \times q$

$d = e^{-1} \bmod (p-1) \times (q-1)$

e, n : public

d, n : private

$sign(msg) = msg^d \bmod n$

$verify(sig) = sig^e \bmod n$

RSA_INIT

```
v1 = a1;
v2 = BN_bin2bn(m3_n, (_BYTE *)128, a1->n);
v3 = v1->e;
v1->n = v2;
v4 = BN_bin2bn(&m3_e, (_BYTE *)1, v3);
v5 = v1->d;
v1->e = v4;
v6 = BN_bin2bn(&m3_d, (_BYTE *)1, v5);
v7 = v1->p;
v1->d = v6;
v8 = BN_bin2bn(m3_p, (_BYTE *)64, v7);
v9 = v1->q;
v1->p = v8;
v10 = BN_bin2bn(m3_q, (_BYTE *)64, v9);
```

RSA_INIT

```
v1 = a1;
v2 = BN_bin2bn(m3_n, (_BYTE *)128, a1->n);
v3 = v1->e;
v1->n = v2;
v4 = BN_bin2bn(&m3_e, (_BYTE *)1, v3);
v5 = v1->d;
v1->e = v4;
v6 = BN_bin2bn(&m3_d, (_BYTE *)1, v5);
v7 = v1->p;
v1->d = v6;
v8 = BN_bin2bn(m3_p, (_BYTE *)64, v7);
v9 = v1->q;
v1->p = v8;
v10 = BN_bin2bn(m3_q, (_BYTE *)64, v9);
```

$$\mathbf{d} = e^{-1} \bmod (p-1) \times (q-1)$$

解锁流程

- 用户提交序列号
- 厂商计算序列号的哈希
- 厂商用私钥给哈希签名得到security.img
- 厂商下发security.img到用户
- 用户使用fastboot刷入到security分区
- 签名校验通过，手机解锁

VERIFY_SECURITY

```
*(_DWORD *)(rsa + 0x3C) |= 0x100u;
read_sig_from_security(sig, 128);               // read security [0-128]
do
{
  if ( !(v2 & 0xF) )
    printf("\n", v4);
  v5 = sig[v2++];
  printf("%02x ", v5);
}
while ( v2 != 128 );
printf("\n", v4);
read_serialno_from_factory((int)serialno, *(&v8 + 1));// read from factory [5:5+len]
MD5((int)serialno, *(&v8 + 1), (signed __int8 *)digest);
v6 = RSA_verify(4, (char *)digest, 16, (int)sig, 128, rsa);
sub_F94C304(rsa, v7);
result = (int)v6;
if ( v6 )
  result = 1;
```

修复

```
DCD aFlash                ; "flash:"
DCD sub_F9238E0
DCD aErase                ; "erase:"
DCD sub_F9238E0
DCD aReboot               ; "reboot"
DCD sub_F9238E0
DCD aFastboot+4           ; "boot"
DCD sub_F9238E0
```

修复

```
DCD aFlash                ; "flash:"
DCD sub_F9238E0
DCD aErase                ; "erase:"
DCD s
DCD a
DCD s
DCD a
DCD s
```

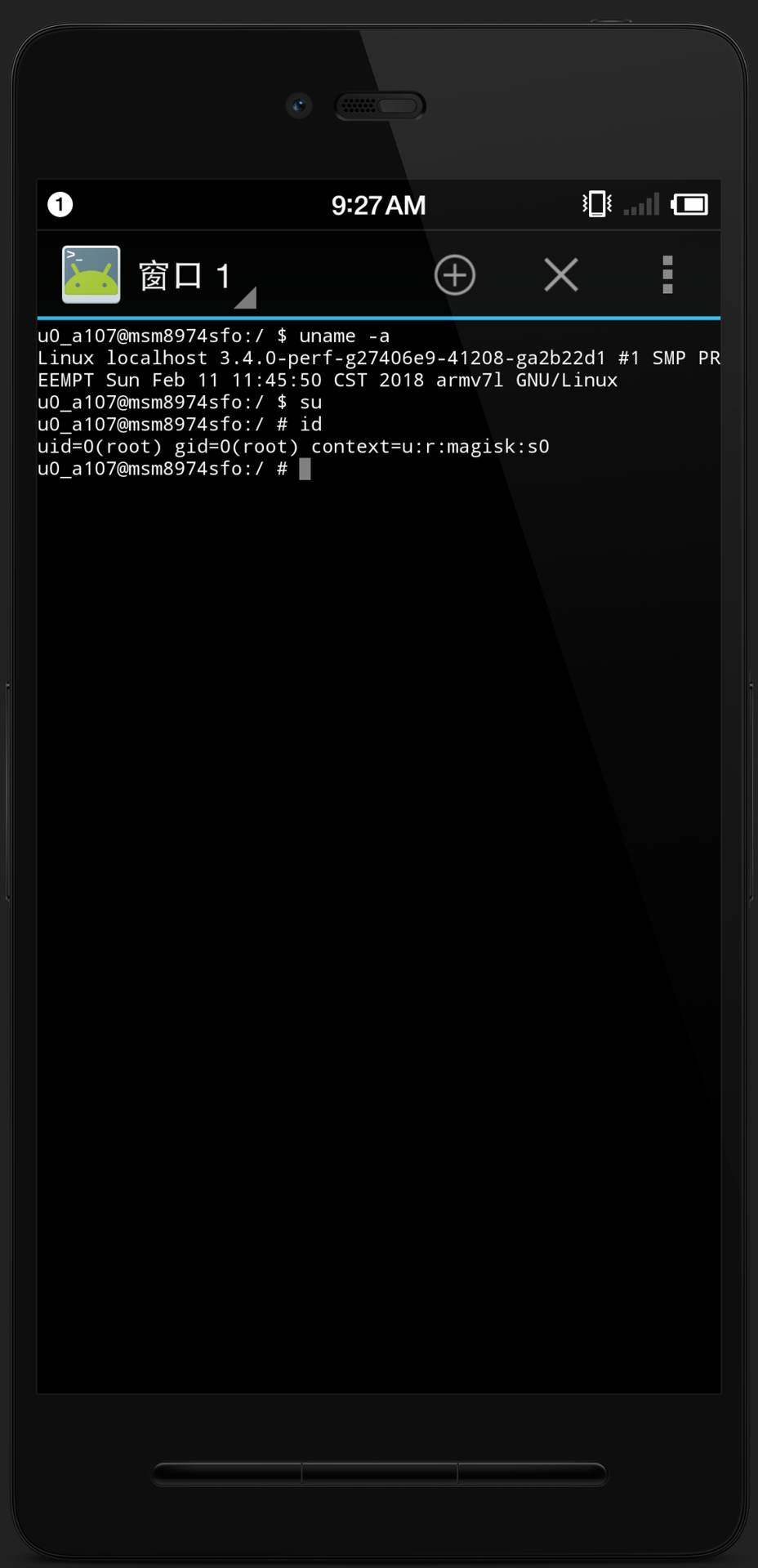
```
int __fastcall sub_F9238E0(int a1)
{
  int v1; // r0@1
  int v2; // r0@1

  sub_F928470(a1);
  v1 = strlen("No Fastboot cmd");
  v2 = screen_display("No Fastboot cmd", v1, 200, 600);
  sub_F917E18(v2);
  sub_F9276E0("no fastboot cmd registered");
  return sub_F927724("");
}
```

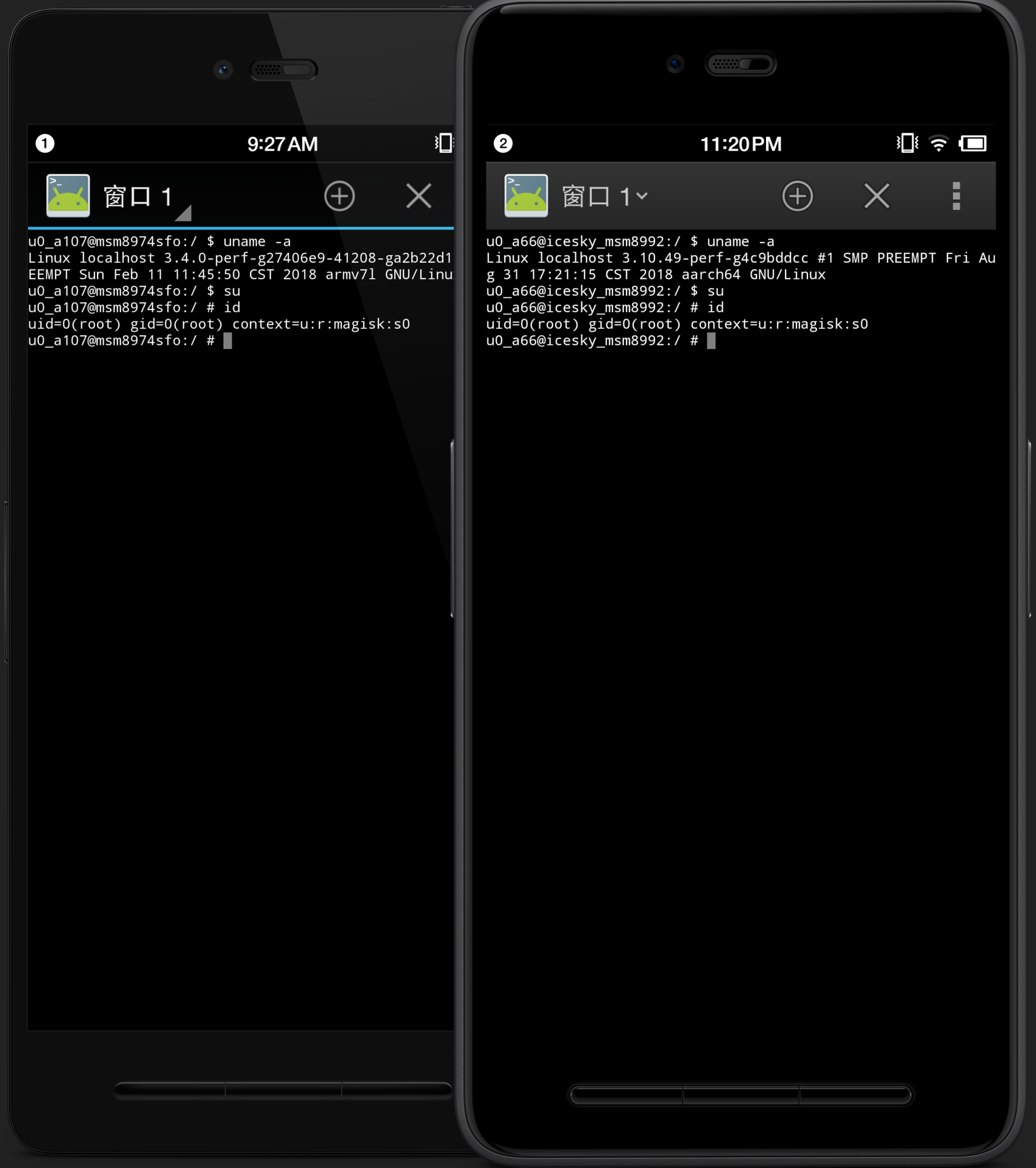
修复 & BYPASS

- recovery刷回老版本
- 同样的方法解锁
- 升级回新版本，但不更新bootloader

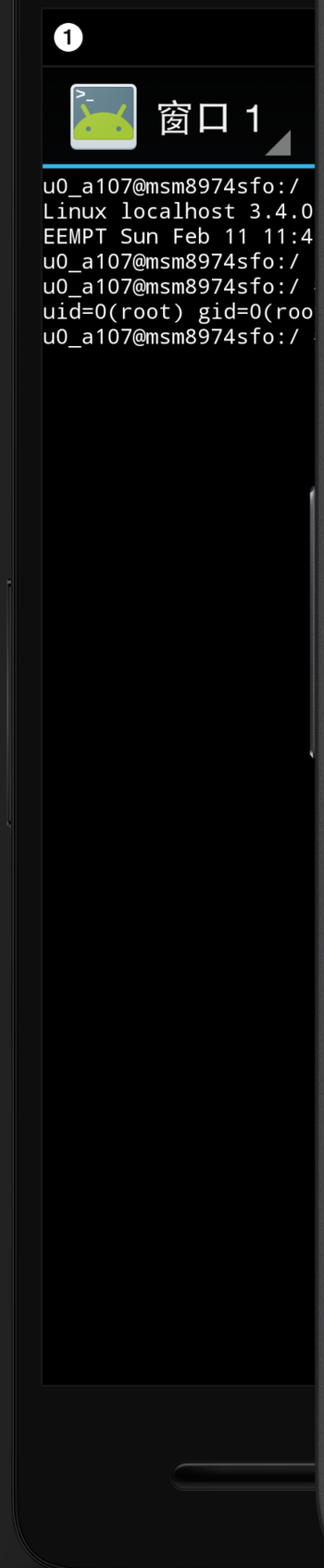
修复 & BYPASS

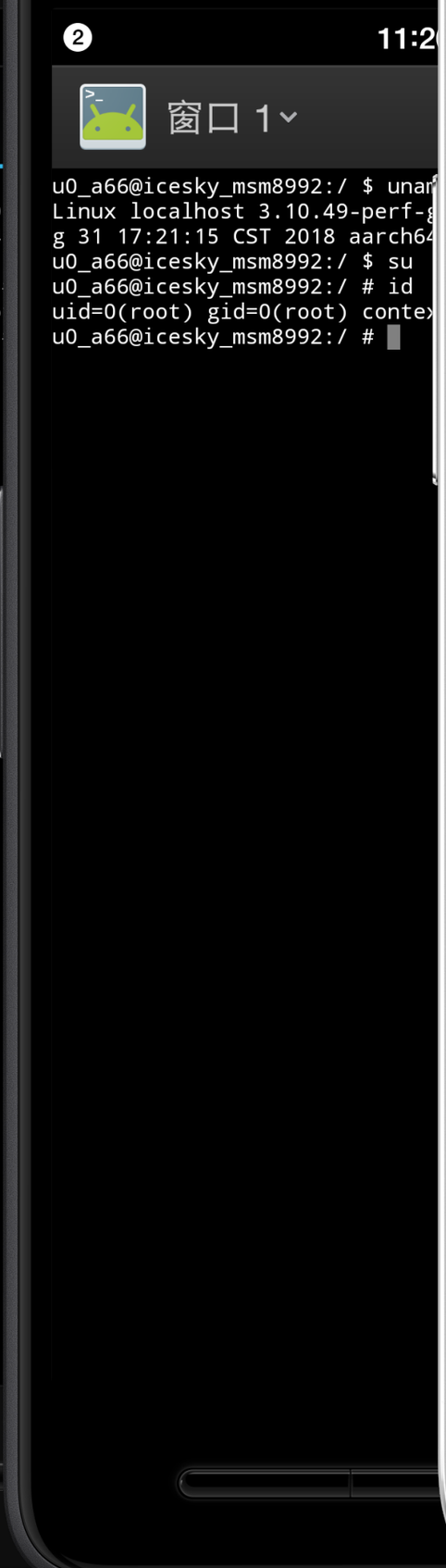
- recovery刷回老版本
- 同样的方法解锁
- 升级回新版本，但不更新bootloader
- M1, PRO, PRO2/2s, R1
  - no aboot_fastboot_register_commands
  - PBL/XBL


1
9:27 AM
窗口 1
u0_a107@msm8974sfo:/ $ uname -a
Linux localhost 3.4.0-perf-g27406e9-41208-ga2b22d1 #1 SMP PR
EEMPT Sun Feb 11 11:45:50 CST 2018 armv7l GNU/Linux
u0_a107@msm8974sfo:/ $ su
u0_a107@msm8974sfo:/ # id
uid=0(root) gid=0(root) context=u:r:magisk:s0
u0_a107@msm8974sfo:/ #

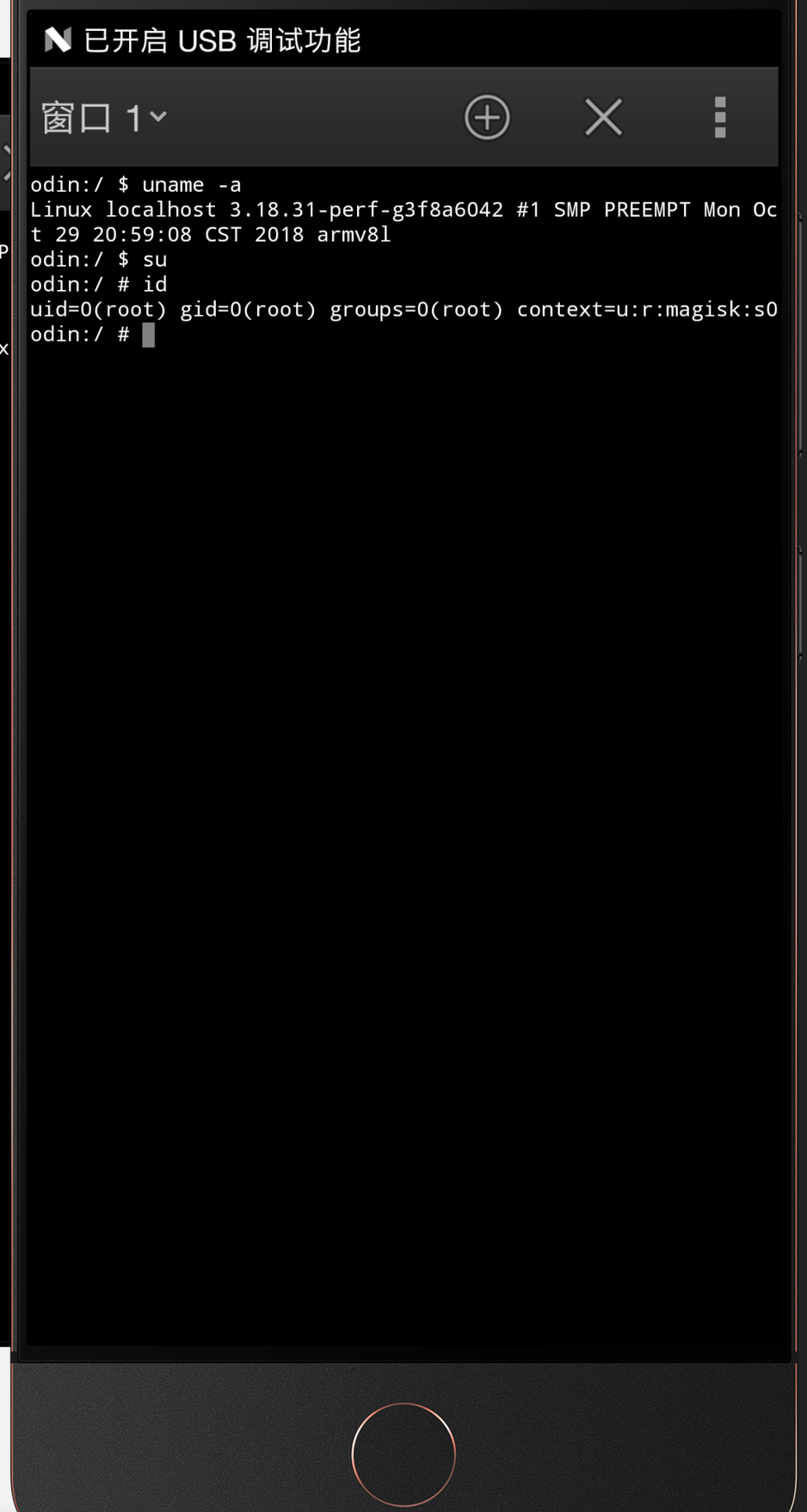

1
9:27 AM
窗口 1
u0_a107@msm8974sfo:/ $ uname -a
Linux localhost 3.4.0-perf-g27406e9-41208-ga2b22d1
EEMPT Sun Feb 11 11:45:50 CST 2018 armv7l GNU/Linu
u0_a107@msm8974sfo:/ $ su
u0_a107@msm8974sfo:/ # id
uid=0(root) gid=0(root) context=u:r:magisk:s0
u0_a107@msm8974sfo:/ #
2
11:20 PM
窗口 1
u0_a66@icesky_msm8992:/ $ uname -a
Linux localhost 3.10.49-perf-g4c9bddcc #1 SMP PREEMPT Fri Au
g 31 17:21:15 CST 2018 aarch64 GNU/Linux
u0_a66@icesky_msm8992:/ $ su
u0_a66@icesky_msm8992:/ # id
uid=0(root) gid=0(root) context=u:r:magisk:s0
u0_a66@icesky_msm8992:/ #


1
窗口 1
u0_a107@msm8974sfo:/
Linux localhost 3.4.0
EEMPT Sun Feb 11 11:4
u0_a107@msm8974sfo:/
u0_a107@msm8974sfo:/
uid=0(root) gid=0(roo
u0_a107@msm8974sfo:/
2
11:2
窗口 1
u0_a66@icesky_msm8992:/ $ unar
Linux localhost 3.10.49-perf-g
g 31 17:21:15 CST 2018 aarch64
u0_a66@icesky_msm8992:/ $ su
u0_a66@icesky_msm8992:/ # id
uid=0(root) gid=0(root) contex
u0_a66@icesky_msm8992:/ #
1
11:38PM
窗口 1
u0_a105@surabaya:/ $ uname -a
Linux localhost 3.18.20-perf-g1bf908fe #1 SMP PREEMPT Tue Oc
t 16 15:07:32 CST 2018 armv8l
u0_a105@surabaya:/ $ su
u0_a105@surabaya:/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:magisk:s0
u0_a105@surabaya:/ #


1
窗口 1
u0_a107@msm8974sfo:/
Linux localhost 3.4.0
EEMPT Sun Feb 11 11:4
u0_a107@msm8974sfo:/
u0_a107@msm8974sfo:/
uid=0(root) gid=0(roo
u0_a107@msm8974sfo:/


2
11:2
窗口 1
u0_a66@icesky_msm8992:/ $ unar
Linux localhost 3.10.49-perf-g
g 31 17:21:15 CST 2018 aarch64
u0_a66@icesky_msm8992:/ $ su
u0_a66@icesky_msm8992:/ # id
uid=0(root) gid=0(root) contex
u0_a66@icesky_msm8992:/ #


1
11:38PM
窗口 1
u0_a105@surabaya:/ $ uname -a
Linux localhost 3.18.20-perf-g1bf908fe #1 SMP
t 16 15:07:32 CST 2018 armv8l
u0_a105@surabaya:/ $ su
u0_a105@surabaya:/ # id
uid=0(root) gid=0(root) groups=0(root) contex
u0_a105@surabaya:/ #


已开启 USB 调试功能
窗口 1
odin:/ $ uname -a
Linux localhost 3.18.31-perf-g3f8a6042 #1 SMP PREEMPT Mon Oc
t 29 20:59:08 CST 2018 armv8l
odin:/ $ su
odin:/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:magisk:s0
odin:/ #

EMUI

EMUI解锁流程

- 手机登录华为账号15天
- 用户提交手机序列号、设备识别码等一系列指纹信息
- 厂商计算这些信息，下发解锁key
- 用户执行fastboot oem unlock key
- 手机解锁

fastboot.img

- update.app/06.fastboot.img
- 跳过签名
  - dd if=06.fastboot.img of=fastboot.img bs=1 skip=4096
- IDA Pro能识别大多数代码
  - 缺少segments info, function symbols

```
002a45b0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a45c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a45d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a45e0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a45f0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4600h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4610h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4620h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4630h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4640h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4650h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4660h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4670h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4680h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4690h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46a0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46b0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46e0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a46f0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4700h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4710h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4720h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4730h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4740h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4750h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4760h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4770h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4780h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
```

```
002a4730h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4740h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4750h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4760h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4770h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4780h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4790h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47a0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47b0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47e0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a47f0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
002a4800h: 0A 2F 63 6C 6F 75 64 2F 6A 65 6E 6B 69 6E 73 2F ; ./cloud/jenkins/
002a4810h: 63 69 2F 77 6F 72 6B 73 70 61 63 65 2F 63 68 69 ; ci/workspace/chi
002a4820h: 70 73 65 74 5F 63 6F 6D 70 6F 6E 65 6E 74 5F 68 ; pset_component_h
002a4830h: 69 33 36 35 30 5F 62 32 30 35 5F 63 68 69 6E 61 ; i3650_b205_china
002a4840h: 5F 62 75 69 6C 64 2F 63 6F 6D 70 6F 6E 65 6E 74 ; _build/component
002a4850h: 5F 63 6F 64 65 2F 6F 75 74 2F 74 61 72 67 65 74 ; _code/out/target
002a4860h: 2F 70 72 6F 64 75 63 74 2F 68 69 33 36 35 30 2F ; /product/hi3650/
002a4870h: 6F 62 6A 2F 46 41 53 54 42 4F 4F 54 5F 4F 42 4A ; obj/FASTBOOT_OBJ
002a4880h: 2F 66 61 73 74 62 6F 6F 74 2F 62 6F 6F 74 61 62 ; /fastboot/bootab
002a4890h: 6C 65 2F 62 6F 6F 74 6C 6F 61 64 65 72 2F 68 69 ; le/bootloader/hi
002a48a0h: 6C 65 67 61 63 79 2F 62 69 6E 2F 66 61 73 74 62 ; legacy/bin/fastb
002a48b0h: 6F 6F 74 2E 65 6C 66 3A 20 20 20 20 20 66 69 6C ; oot.elf:     fil
002a48c0h: 65 20 66 6F 72 6D 61 74 20 65 6C 66 36 34 2D 6C ; e format elf64-l
002a48d0h: 69 74 74 6C 65 61 61 72 63 68 36 34 0A 0A 53 59 ; ittleaarch64..SY
002a48e0h: 4D 42 4F 4C 20 54 41 42 4C 45 3A 0A 30 30 30 30 ; MBOL TABLE:.0000
002a48f0h: 30 30 30 30 30 30 30 30 30 30 30 20 6C 20 20 20 ; 000000000000 l
002a4900h: 20 20 64 20 20 2E 74 65 78 74 09 30 30 30 30 30 ;   d  .text.00000
002a4910h: 30 30 30 30 30 30 30 30 30 30 30 20 2E 74 65 78 ; 00000000000 .tex
```

```
13383 SYMBOL TABLE:
13384 0000000000000000 l    d  .text  0000000000000000 .text
13385 00000000001f9a98 l    d  __ex_table 0000000000000000 __ex_table
13386 00000000001f9ab8 l    d  .text.unlikely 0000000000000000 .text.unlikely
13387 00000000001f9b30 l    d  .data  0000000000000000 .data
13388 00000000002a3180 l    d  .got.plt 0000000000000000 .got.plt
13389 00000000002a3198 l    d  .module  0000000000000000 .module
13390 00000000002a35a0 l    d  .bss 0000000000000000 .bss
13391 0000000000000000 l    d  .debug_info 0000000000000000 .debug_info
13392 0000000000000000 l    d  .debug_abbrev 0000000000000000 .debug_abbrev
13393 0000000000000000 l    d  .debug_loc 0000000000000000 .debug_loc
13394 0000000000000000 l    d  .debug_aranges 0000000000000000 .debug_aranges
13395 0000000000000000 l    d  .debug_line 0000000000000000 .debug_line
13396 0000000000000000 l    d  .debug_str 0000000000000000 .debug_str
13397 0000000000000000 l    d  .debug_frame 0000000000000000 .debug_frame
13398 0000000000000000 l    d  .debug_ranges 0000000000000000 .debug_ranges
13399 0000000000000000 l    d  .comment 0000000000000000 .comment
13400 0000000000000000 l    d  .debug_macro 0000000000000000 .debug_macro
13401 0000000000000000 l    d  .text.boot_start 0000000000000000 .text.boot_start
13402 0000000000000000 l    d  .text.gic 0000000000000000 .text.gic
13403 0000000000000000 l    d  .text.cpu_init 0000000000000000 .text.cpu_init
13404 0000000000000000 l    d  .text.cluster_init 0000000000000000 .text.cluster_init
13405 00000000000001e4 l       .text  0000000000000000 cpu_resume_entry
13406 00000000000003a8 l       .text  0000000000000000 forever_wfi
13407 00000000000001dc l       .text  0000000000000000 secondary_cpu
13408 0000000000000184 l       .text  0000000000000000 main_cpu_entry
13409 00000000000001a4 l       .text  0000000000000000 ttb_setup
13410 00000000000001f8 l       .text  0000000000000000 hisi_cupidle_lock
13411 000000000000025c l       .text  0000000000000000 hisi_clear_cpuidle_bit
13412 0000000000000230 l       .text  0000000000000000 hisi_cupidle_unlock
13413 000000000000028c l       .text  0000000000000000 enable_smp
```

```
BOOTLOADER_END.                 0x2c2158
BOOTLOADER_HEAP_CACHE_TOP       0x2fffffff
BOOTLOADER_STACK                0x3c0000
BOOTLOADER_HEAP                 0x30000000
BOOTLOADER_LHEAP_TOP            0x2e000000
BOOTLOADER_HEAP_CACHE.          0x2e000000
BOOTLOADER_START                0x0
BOOTLOADER_HEAP_TOP             0x33ffffff
BOOTLOADER_BSS                  0x2a35a0
BOOTLOADER_LHEAP.               0xe000000
```

分析代码

- 定义命令结构体，修正解析结果
  - p_gui_ops = get_operators("gui");

```
gui_ops         DCQ gui_main
                DCQ gui_settext
                DCQ lcd_error_print
                DCQ display_boot_status
                DCQ display_lockchange_warning
```

- 修正字符串相关函数，方便快速定位

```
DCQ aOemHwdogCertif     ; "oem hwdog certify begin"
DCQ cmd_hwdog_certify_get
DCQ aFlashSlock         ; "flash:slock"
DCQ cmd_hwdog_certify_put
DCQ aOemHwdogCert_0     ; "oem hwdog certify close"
DCQ cmd_hwdog_certify_relock
DCQ aOemGetBootinfo     ; "oem get-bootinfo"
DCQ cmd_lock_stat_info
DCQ aOemCheckRootin     ; "oem check-rootinfo"
DCQ oem_check_root_info
DCQ aOemCheckImage      ; "oem check-image"
DCQ cmd_image_sign_check
DCQ aOemUnlock          ; "oem unlock"
DCQ usr_fastboot_unlock
DCQ aOemRelock          ; "oem relock"
DCQ usr_fastboot_relock
DCQ aFlashingUnlock     ; "flashing unlock"
DCQ usr_fastboot_unlock
DCQ aFlashingLock       ; "flashing lock"
DCQ usr_fastboot_relock
DCQ aOemHwdogCert_1     ; "oem hwdog certify set"
DCQ cmd_hwdog_certify_set
DCQ aOemFrpErase        ; "oem frp-erase"
DCQ cmd_frp_erase
DCQ aOemFrpUnlock       ; "oem frp-unlock"
DCQ cmd_frp_unlock
```

```
if ( is_unlocked == 1 )
{
  strncpy(v8, "FAILalready fastboot unlocked", 65i64);
  *(_BYTE *)(v8 + 64) = 0;
  result = v17;
}
else
{
  v18 = sha256_nvwvlock_cmp(key, keylen);
  v19 = v18;
  if ( !v18 )
  {
    v20 = (*(__int64 (**)(void))(p_gui_ops + 0x20))();
    v21 = v20;
    if ( v20 )
    {
      if ( v20 == 1 )
      {
        v30 = 1;
        strncpy(v8, "FAILYou choose not to unlock the phone", 65i64);
        *(_BYTE *)(v8 + 64) = v19;
        (*(void (**)(void))p_gui_ops)();
        return v30;
      }
      v22 = *(void (**)(void))p_gui_ops;
      is_unlocked = 1;
      v22();
      if ( !(oeminfo_lock_stat_write(1) & 0x80000000) )
        goto LABEL_9;
    }
    else
    {
      v32 = *(void (**)(void))p_gui_ops;
      v30 = 1;
      is_unlocked = 1;
      v32();
      if ( !(oeminfo_lock_stat_write(1) & 0x80000000) )
      {
        if ( (unsigned int)reboot_factory() )
```

```
gui_ops         DCQ gui_main
                DCQ gui_settext
                DCQ lcd_error_print
                DCQ display_boot_status
                DCQ display_lockchange_warning
```

sha256_nvwvlock_cmp

```
v144.nv_number = 315;
v144.nv_operation = 1;
strncpy(v144.nv_name, "USRKEY", 8i64);
v144.nv_name[7] = 0;
v144.valid_size = 0x20;
v58 = (*p_nve_ops)(&v144);
v66 = v58;
if ( v58 )
{
  v74 = "usbloader";
  v75 = ": ";
  v76 = "read wvkey nv data error\n";
  cprintf((__int64)"usbloader", v59, v60, v61, v62, v63, v64, v65);
  cprintf((__int64)": ", v119, v120, v121, v122, v123, v124, v125);
  goto LABEL_10;
}
memcpy(wvkey, v144.nv_data, 0x20u);
v141 = v66;
sha256_handle_standard(userinput_key, 16u, (__int64)hash);
result = strncmp(hash, wvkey, 0x20i64);
```

```
nve_ops                 DCQ hisi_nve_direct_access
                                                  ;
                        DCQ search_cmd_in_nve
                        DCQ nvme_updata
```

```
                                DCQ search_cmd_in_nve
                                DCQ nvme_updata
if ( v58 )
{
  v74 = "usbloader";
  v75 = ": ";
  v76 = "read wvkey nv data error\n";
  cprintf((__int64)"usbloader", v59, v60, v61, v62, v63, v64, v65);
  cprintf((__int64)": ", v119, v120, v121, v122, v123, v124, v125);
  goto LABEL_10;
}
memcpy(wvkey, v144.nv_data, 0x20u);
v141 = v66;
sha256_handle_standard(userinput_key, 16u, (__int64)hash);
result = strncmp(hash, wvkey, 0x20i64);
```

nvme.img

```
00029d60h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00029d70h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00029d80h: 3B 01 00 00 55 53 52 4B 45 59 00 00 01 00 00 00 ; ;...USRKEY......
00029d90h: 20 00 00 00 00 00 00 00 4F C4 07 33 7D 8F 1F 07 ;  .......O?3}?.
00029da0h: 0C DA 57 4D F5 AB 95 FC 7A A6 78 CC 29 90 00 E0 ; .赪M醌磬z ???
00029db0h: 1C 53 2C 8D 09 CA 2C 92 00 00 00 00 00 00 00 00 ; .S,???........
00029dc0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00029dd0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00029de0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00029df0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
```

解锁逻辑

- sha256(key) == nvme.USRKEY

```
if ( is_unlocked == 1 )
{
  strncpy(v8, "FAILalready fastboot unlocked", 65i64);
  *(_BYTE *)(v8 + 64) = 0;
  result = v17;
}
else
{
  v18 = sha256_nvwvlock_cmp(key, keylen);
  v19 = v18;
  if ( !v18 )
  {
    v20 = (*(__int64 (**)(void))(p_gui_ops + 0x20))();
    v21 = v20;
    if ( v20 )
    {
      if ( v20 == 1 )
      {
        v30 = 1;
        strncpy(v8, "FAILYou choose not to unlock the phone", 65i64);
        *(_BYTE *)(v8 + 64) = v19;
        (*(void (**)(void))p_gui_ops)();
        return v30;
      }
      v22 = *(void (**)(void))p_gui_ops;
      is_unlocked = 1;
      v22();
      if ( !(oeminfo_lock_stat_write(1) & 0x80000000) )
        goto LABEL_9;
    }
    else
    {
      v32 = *(void (**)(void))p_gui_ops;
      v30 = 1;
      is_unlocked = 1;
      v32();
      if ( !(oeminfo_lock_stat_write(1) & 0x80000000) )
      {
        if ( (unsigned int)reboot_factory() )
```

oeminfo_lock_stat_write

```
v150.nv_operation = 1;
v150.nv_number = 315;
strncpy(v150.nv_name, "USRKEY", 8i64);
v150.nv_name[7] = 0;
v150.valid_size = 32;
v18 = (*p_nve_ops)(&v150);
v26 = v18;
if ( v18 )
{
  v34 = "usbloader";
  v35 = ": ";
  v36 = "%s:Read nv data failed\n";
  cprintf((__int64)"usbloader", v19, v20, v21, v22, v23, v24, v25);
  cprintf((__int64)": ", v106, v107, v108, v109, v110, v111, v112);
  goto LABEL_8;
}
memcpy((char *)&usrkey, v150.nv_data, 0x10u);
v143 = v26;
if ( v1 == 1 )
{
  aes_encrypt_operate((__int64)"ABABCDCD", (char *)&usrkey, (__int64)&output);
}
else
{
  if ( v1 != 2 && v1 )
  {
    v34 = "usbloader";
    v35 = ": ";
    v36 = "%s:oeminfo lock state not match!\n";
    cprintf((__int64)"usbloader", v27, v28, v29, v30, v31, v32, v33);
    cprintf((__int64)": ", v37, v38, v39, v40, v41, v42, v43);
```

```
  memcpy((char *)&usrkey, v150.nv_data, 0x10u);
  v143 = v26;
  if ( v1 == 1 )
  {
    aes_encrypt_operate((__int64)"ABABCDCD", (char *)&usrkey, (__int64)&output);
  }
  else
  {
    if ( v1 != 2 && v1 )
    {
      v34 = "usbloader";
      v35 = ": ";
      v36 = "%s:oeminfo lock state not match!\n";
      cprintf((__int64)"usbloader", v27, v28, v29, v30, v31, v32, v33);
      cprintf((__int64)": ", v37, v38, v39, v40, v41, v42, v43);
LABEL_8:
      cprintf((__int64)v36, (__int64)"oeminfo_lock_stat_write", v44, v45, v46, v47, v48, v49);
      v50 = get_boot_time();
      log_buffer("[%d ms]", v50, v51, v52, v53, v54, v55, v56);
      log_buffer(v34, v57, v58, v59, v60, v61, v62, v63);
      log_buffer(v35, v64, v65, v66, v67, v68, v69, v70);
      log_buffer(v36, (__int64)"oeminfo_lock_stat_write", v71, v72, v73, v74, v75, v76);
      return 0xFFFFFFFFi64;
    }
    aes_encrypt_operate((__int64)"EFEFABCD", (char *)&usrkey, (__int64)&output);
  }
  if ( p_oeminfo_ops[1](93i64, 16i64, &output)
  {
    v34 = "usbloader";
    v35 = ": ";
    v36 = "%s:write oeminfo lock stat info failed!\n";
    cprintf((__int64)"usbloader", v78, v79, v80, v81, v82, v83, v84);
    cprintf((__int64)": ", v113, v114, v115, v116, v117, v118, v119);
    goto LABEL_8;
  }
```

```
oeminfo_ops     DCQ get_oeminfo
                DCQ set_oeminfo
                DCQ erase_oeminfo
```

解锁逻辑

- sha256(key) == nvme.USRKEY
  - oeminfo.state = aes("ABABCDCD", nvme.USRKEY)
  - oeminfo.state -> oeminfo[(93-1)<<12 + 0x200]
  - "EFEFABCD" -> relock
  - "others" -> lock

解锁逻辑

- sha256(key) == nvme.USRKEY
  - oeminfo.state = aes("ABABCDCD", nvme.USRKEY)
  - oeminfo.state -> oeminfo[(93-1)<<12 + 0x200]
  - "EFEFABCD" -> relock
  - "others" -> lock

GTX960

```
Session..........: hashcat
Status...........: Running
Hash.Type........: SHA-256
Hash.Target......: 754907de3cfe9eb194111cbaca22831aa24c288c2ac9ee08f36...48e79a
Time.Started.....: Thu Apr 13 14:14:33 2017 (2 secs)
Time.Estimated...: Sun Aug 27 09:29:44 2017 (135 days, 19 hours)
Guess.Mask.......: ?d?d?d?d?d?d?d?d?d?d?d?d?d?d?d?d [16]
Guess.Queue......: 1/1 (100.00%)
Speed.Dev.#1.....:   852.3 MH/s (300.08ms)
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 2097152000/10000000000000000 (0.00%)
Rejected.........: 0/2097152000 (0.00%)
Restore.Point....: 2097152/10000000000000 (0.00%)
Candidates.#1....: 1232558340111111 -> 9557823050111111
HWMon.Dev.#1.....: Temp: 45c Fan: 49% Util: 93% Core:1366MHz Mem:3004MHz Bus:16
```

hashcat -m1400 -a3 -w3 hash ?d?d?d?d...

16 NVIDIA K80 GPUs

```
Session..........: hashcat
Status...........: Running
Hash.Type........: SHA-256
Hash.Target......: 754907de3cfe9eb194111cbaca22831aa24c288c2ac9ee08f36...48e79a
Time.Started.....: Thu Jul 13 03:59:20 2017 (10 secs)
Time.Estimated...: Sat Jul 22 08:35:02 2017 (9 days, 4 hours)
Guess.Mask.......: ?d?d?d?d?d?d?d?d?d?d?d?d?d?d?d?d [16]
Guess.Queue......: 1/1 (100.00%)
Speed.Dev.#1.....:   793.5 MH/s (61.35ms)
Speed.Dev.#2.....:   757.0 MH/s (63.79ms)
Speed.Dev.#3.....:   771.9 MH/s (59.82ms)
Speed.Dev.#4.....:   748.0 MH/s (61.73ms)
Speed.Dev.#5.....:   787.9 MH/s (61.79ms)
Speed.Dev.#6.....:   733.2 MH/s (65.82ms)
Speed.Dev.#7.....:   797.1 MH/s (57.98ms)
Speed.Dev.#8.....:   745.6 MH/s (61.93ms)
Speed.Dev.#9.....:   808.2 MH/s (59.78ms)
Speed.Dev.#10.....:   770.0 MH/s (62.70ms)
Speed.Dev.#11.....:   794.5 MH/s (58.19ms)
Speed.Dev.#12.....:   738.4 MH/s (65.41ms)
Speed.Dev.#13.....:   798.3 MH/s (60.52ms)
Speed.Dev.#14.....:   751.7 MH/s (64.28ms)
Speed.Dev.#15.....:   796.7 MH/s (58.04ms)
Speed.Dev.#16.....:   742.3 MH/s (65.08ms)
Speed.Dev.#17.....:   257.9 MH/s (60.44ms)
Speed.Dev.#*.....: 12592.3 MH/s
```

从ANDROID内部解锁

- sha256(key) == nvme.USRKEY
- oeminfo.state = aes("ABABCDCD", nvme.USRKEY)

```
brw-rw---- 1 system system 179,7 mmcblk0p7
brw------- 1 root   root   179,8 mmcblk0p8
```

SYSTEM解锁

- /sbin/oeminfo_nvm_server
  - running as root
  - listening an unix socket /dev/socket/oeminfo_nvm
  - accessible to system

```
srw-rw---- root system /dev/socket/oeminfo_nvm
```

```
switch ( cmd[0] )
{
  case 1:
    v45 = *(_DWORD *)&v137.sun_family;
    len = (unsigned int)cmd[2];
    v138 = "oeminfo_write";
    if ( !cmd[2] )
    {
      sub_4019BC(
        6u,
        (__int64)"OEMINFO_NVM_SERVER",
        "%s,the size for malloc is less than 0\n",
        (__int64)"oeminfo_write_data",
        (unsigned int)cmd[0],
        v41,
        v42,
        v43,
        v44,
        v135);
      goto LABEL_5
}
idx = cmd[1];
buff_malloc = malloc((unsigned int)cmd[2]);
buff_malloc_ = buff_malloc;
```

```
memset(buff_malloc, 0, len);
v52 = recv_buff_from_remote(v45, buff_malloc_, len);
if ( v52 == (_DWORD)len )
{
  sub_40EDC4((unsigned __int16 *)&unk_451000);
  v54 = write_oeminfo(idx, len, buff_malloc_);
  sub_40F164((__int64)&unk_451000);
```

SYSTEM解锁

- /sbin/oeminfo_nvm_server
  - cmd[0]=1, cmd[1]=offset, cmd[2]=length
  - filename[260]
  - write file content to oeminfo

```
void *handle = dlopen("/vendor/lib/liboeminfo.so", RTLD_LAZY);
void (*rmt_oeminfo_write)(int, int, char);
rmt_oeminfo_write= dlsym(handle, "rmt_oeminfo_write");
rmt_oeminfo_write(93, 16, lockvalue);
```

SYSTEM解锁

- persistent root
- 用户可任意修改解锁状态，甚至改回lock状态
  - lock -> unlock
  - unlock -> relock

修复

- EMUI 5
  - [0-9]{16} -> [0-9A-Z]{16}
- EMUI X
  - PBK_SHA256_**RSA**
  - HMAC (KEY from HUK)

谢谢

Q&A

@hhj4ck

Q & A

@hhj4ck