Realistic Trends in Vulnerability based on Hacking into Vehicle

Car Hacking Village, DEF CON 28

Ryosuke Uematsu, Shogo Nakao, Ryoichi Teramura, Tatsuya Katsuhara

NDIAS, Ltd.

Agenda

- $ whoami
- Background
- HOW testing
- Vulnerability trends in Vehicles
- Conclusion & Perspective

$ whoami

NDIAS is …

- The company for automotive cybersecurity services
  - Specialist group for automotive cybersecurity assessment and consultation
  - Jointly established by NRI Secure and DENSO in 2018

DENSO × NRI SECURE

Global automotive components manufacturer

IT & IoT Security Services

Ndias

Don’t worry!
We **NEVER** share the client info. to DENSO and NRI secure without the permission!

NDIAS activities

- Business
  - We are working with more than 6 OEMs and some suppliers
- Competition
  - **DEFCON Car hacking Village CTF 2019: 4th place** as NDIER
    - NDIER is the joint team with NDIAS and IERAE security
  - **DEFCON Car hacking Village CTF 2018: 5th place** as katagaitai

Car Hacking Village CTF

Top Ranked Players

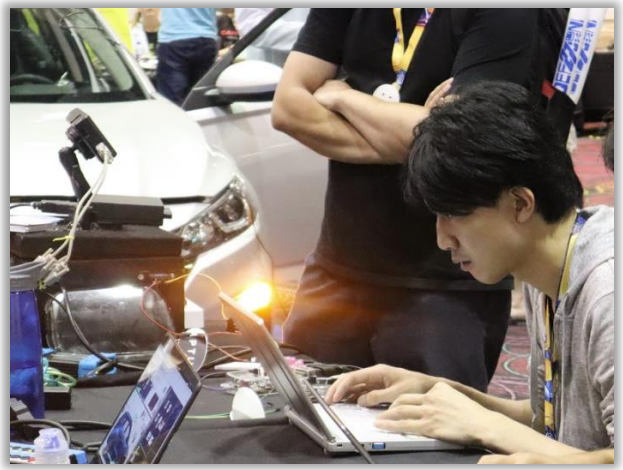
Rank	Player	Points	Num. Solved
1	CANucks (6)	5498	121
2	2x (5)	4741	113
3	Charity Case (5)	4449	106
4	ndier (6)	2826	92
5	Qwerty (4)	1674	66
6	I'm Not Your Dadmin (5)	1564	55
7	SucIC (4)	1272	56
8	Culvers (2)	1166	34
9	Welcome Thrillhouse Group (1)	1156	40
10	CMD7 (4)	1114	53

$ whoami

We are…

- **Ryosuke Uematsu**
  - Security engineer
  - @tansokun920

- **Shogo Nakao**
  - Security engineer



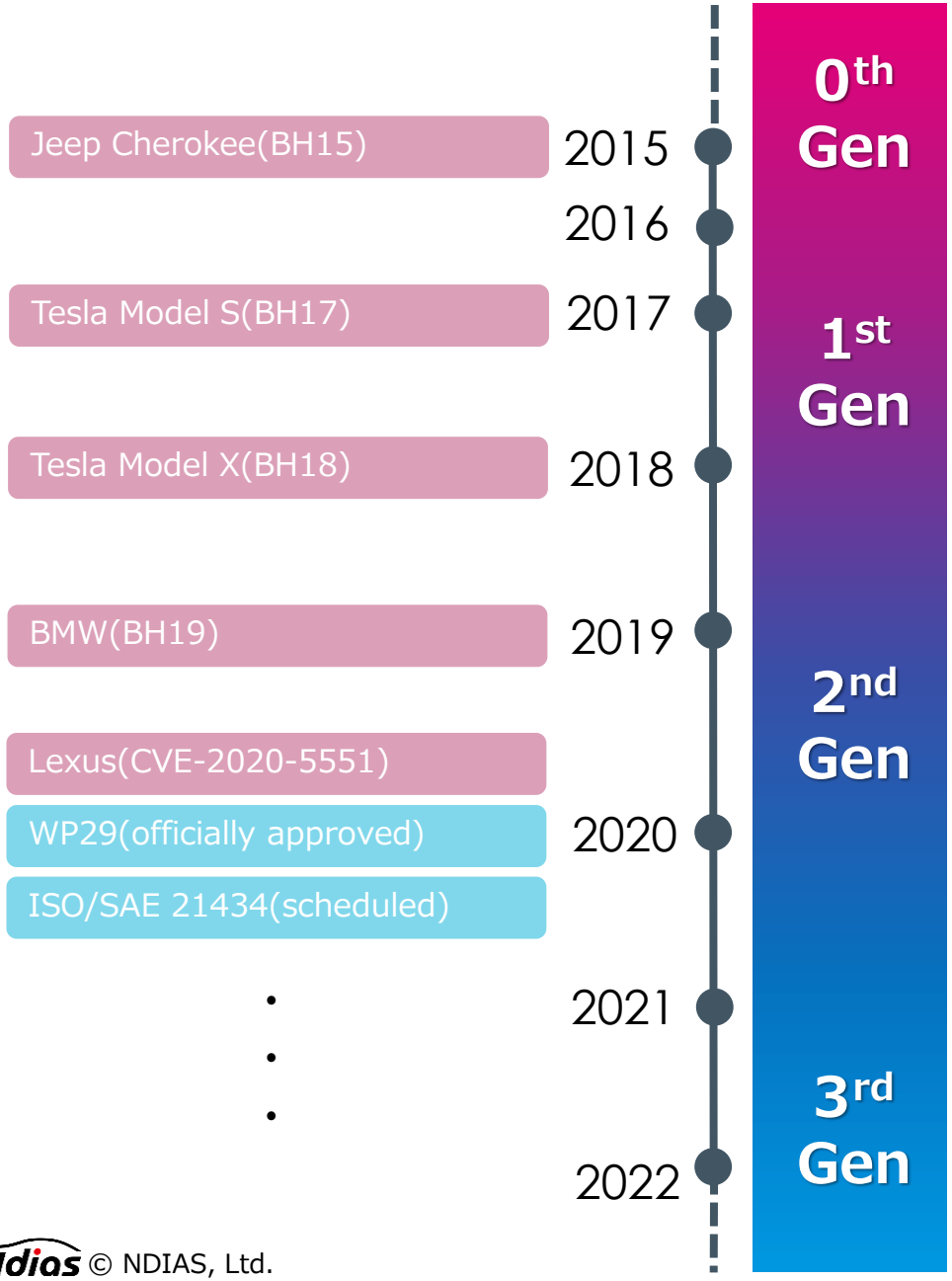
- **Ryoichi Teramura**
  - Manager
  - @trmr105

- **Tatsuya Katsuhara**
  - Director
  - @kthrtty

Background

Car Cybersecurity Generation

Note: it depends on car manufactures.



- Before “Cherokee”(aka B.C.)
- Almost no mitigation, no regulation
- It’s like “Hack the box”

- Simple mitigations
- No changes to the electronic platform

- Electronic platform designed with cybersecurity
- Defense-in-depth
- Not perfect, but moderate

- WP29, ISO/SAE 21434 compliant
- Security by design
- Cybersecurity for pre/post-production

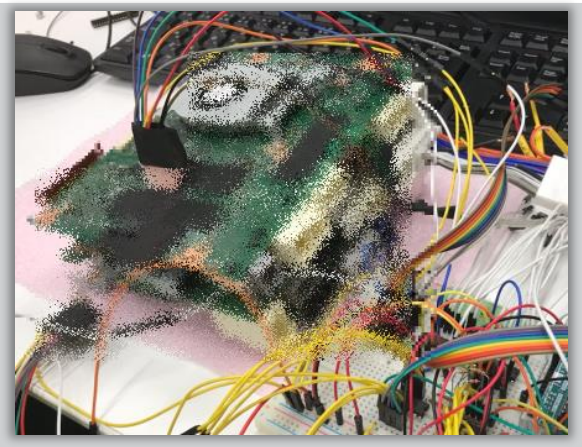
Executive summary

- Introduction of our approaches focusing on high-risk vulnerabilities in the whole “vehicle” perspective
- Trends of vulnerabilities found in development phase for actual 2nd Gen cars and ECUs.
  - Based on more than **40 ECUs** and **300 vulnerabilities**.
- What kinds of mitigations are suit for those vulnerabilities
- Hope that this result helps the developers in car manufacturers and ECU suppliers!

How Testing

Pen test for “vehicle” and “ECUs”

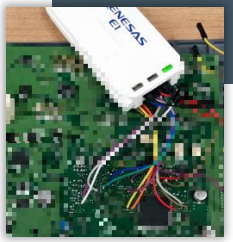
- Car security testing has 2 categories:
  - Penetration test for vehicles
  - Penetration test for ECUs
- Today’s results are mainly based on the test for ECUs.

	Pen test for vehicle	Pen test for ECUs
		
Check point	Which thereat scenarios are realized?	Which vulnerabilities are exsit?
Test type	Integration test	Unit test
Num. of vulns found	A few	A lot

Trial to find vulnerabilities everywhere

- Vulnerabilities are found in everywhere through HW/SW stack
  - e.g., Hardware, Interface driver, Boot strap, OS Kernel, OSS, Application etc.
- We will introduce our approaches for finding vulnerabilities

Scope of HW/SW Stack



Application

IF

OS/OSS/
Off-the-shelf

HW

×

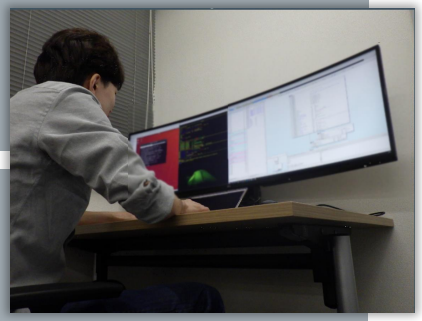
Test methods

Interface test

- Remote I/F (e.g., Cellular)
- Adjacent I/F (e.g., Wi-Fi, Bluetooth, LF/RF, NFC)
- Local I/F (e.g., USB, CAN, Ethernet, EV)
- Physical I/F (e.g., JTAG, UART, SPI)

Software analysis

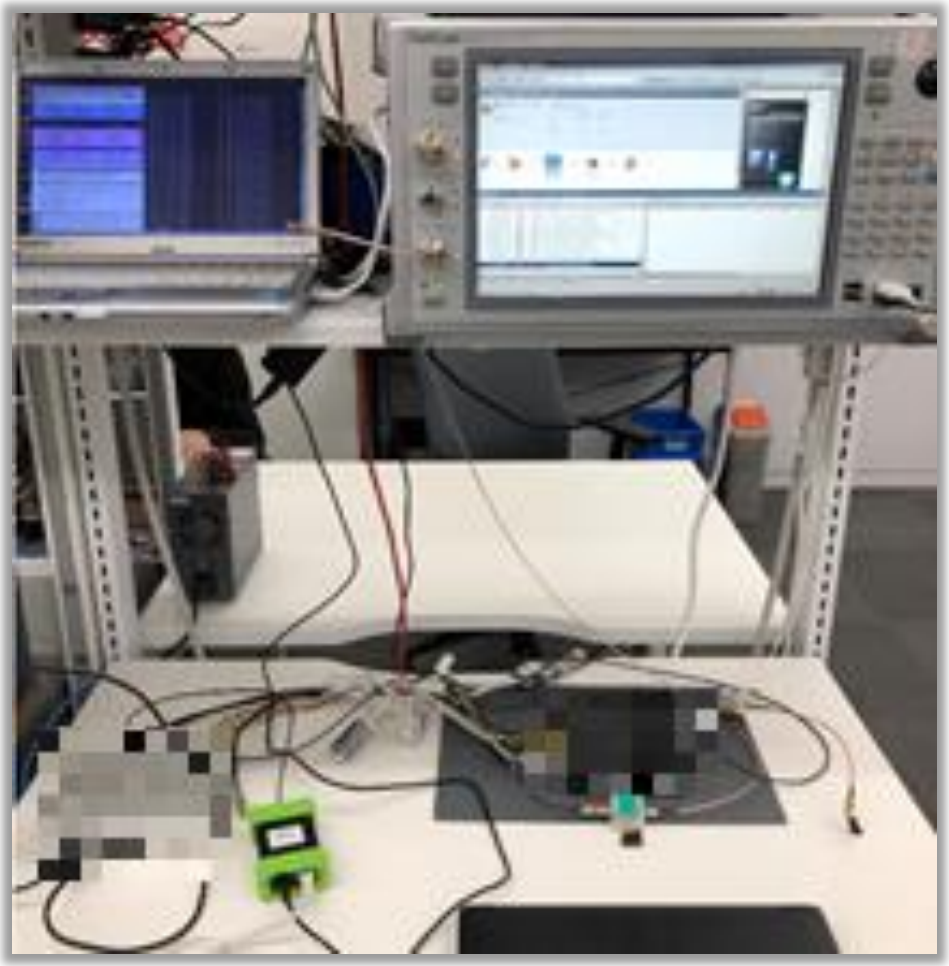
- Dynamic software analysis
- Static software analysis



Interface Test : Remote（Cellular)

- Penetrating wireless interface
  - Check functions and messages over Cellular(2G/3G/4G)
    - LTE stack test
    - Port scan
    - MITM
  - Check a lot of things
    - Authentication bypass
    - MITM attack
    - Unintended service remaining
    - Replay attack
    - Confidentiality of sensitive data

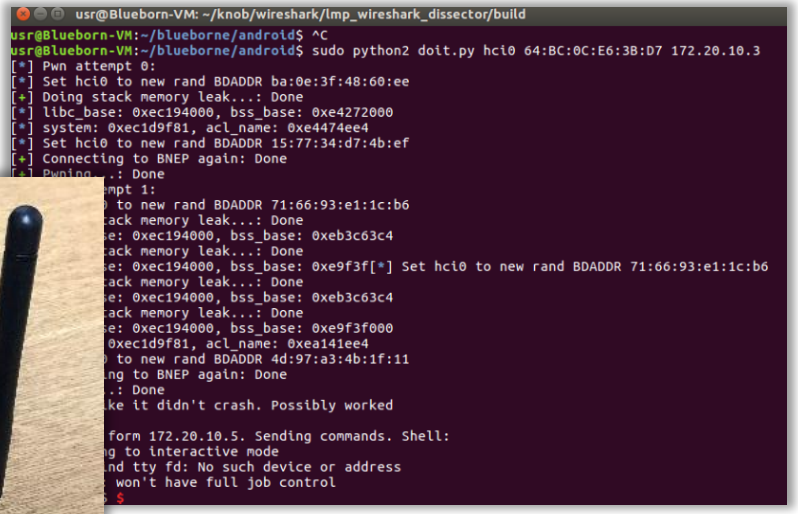
Anritsu MD8475A



Interface Test : Adjacent（wireless LAN, Bluetooth, …）

- Penetrating adjacent interface (wireless LAN, Bluetooth, …)
  - Check functions and messages over adjacent network
    - Wireless LAN protocol and Bluetooth stack test
    - Well-known vulnerability check
    - Port scan
    - MITM
    - Sniffing（for Bluetooth）
  - Check a lot of things
    - Authentication bypass
    - MITM attack
    - Unintended service remaining
    - Replay attack
    - Confidentiality of sensitive data
    - Well-known vulnerability remaining

Well-known vulnerability check about Bluetooth



BLE Sniffer

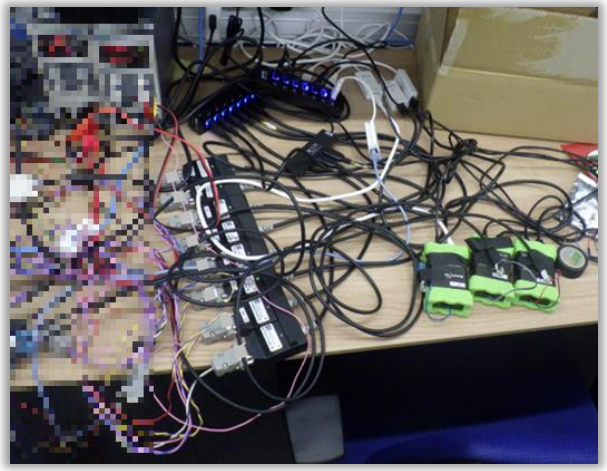
LF/RF

NFC R/W

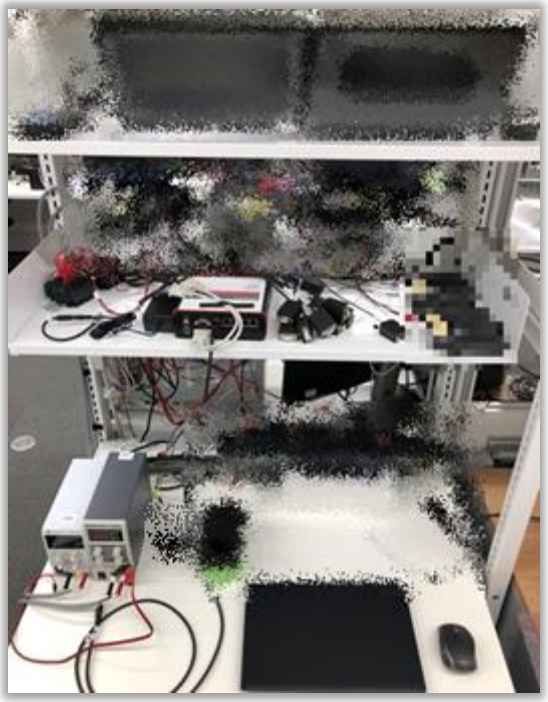
Interface Test : Local（USB, CAN, Ethernet, …）

- Penetrating local interface
  - Check functions and messages over local interface
    - Fuzzing
    - Port scan
    - Well-known vulnerability check
    - Using UDS or not?
  - Check a lot of things
    - Unintended service remaining
    - Replay attack
    - Security access
    - Message Authentication Code
    - Confidentiality of sensitive data
    - Well-known vulnerability remaining

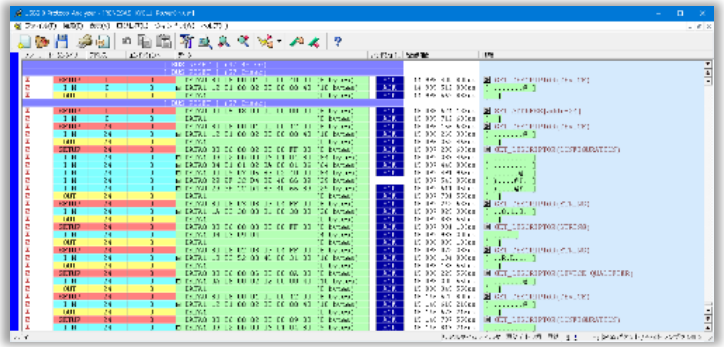
CAN and Ethernet



Test bench

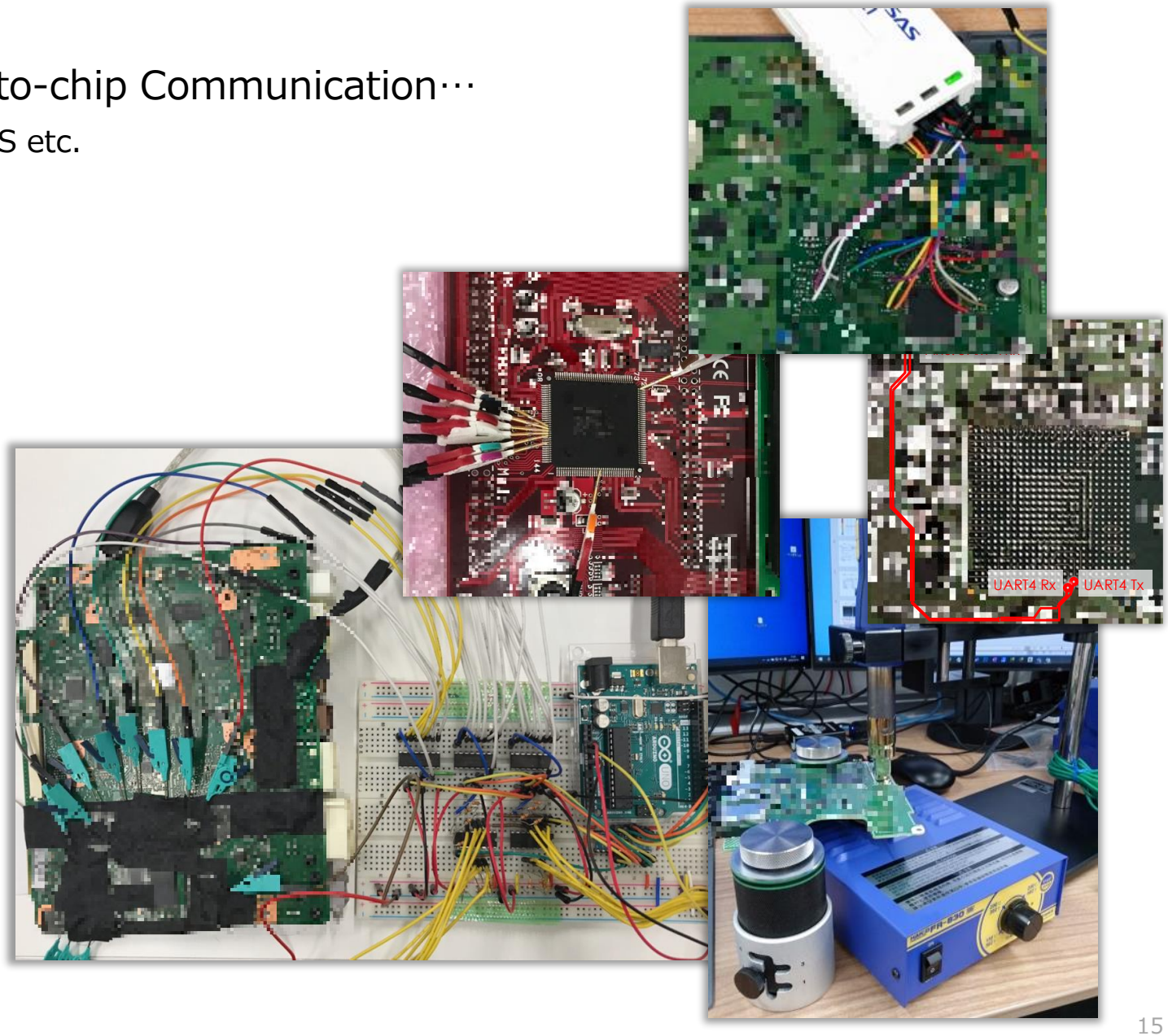


USB Analyzer



Physical Interface Test

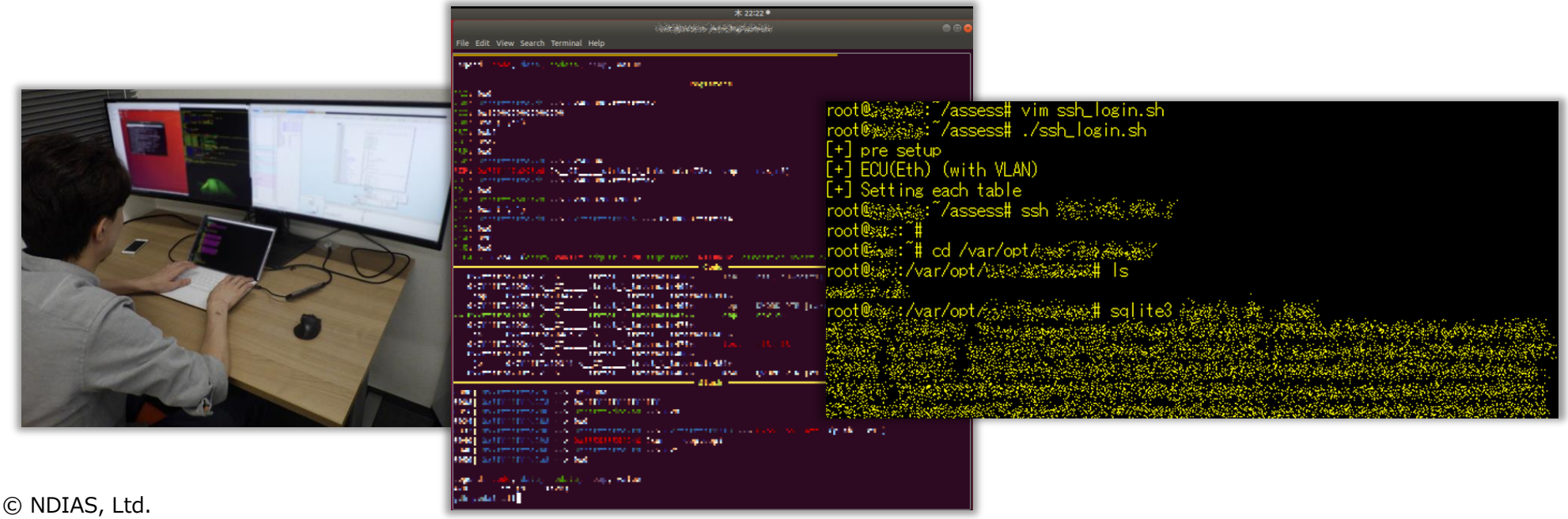
- Penetrating HW
  - Check Debug port, Chip-to-chip Communication…
    - JTAG, SPI, UART, eMMC, UFS etc.
    - Do you like soldering?
  - Check a lot of things
    - Firmware extraction
    - Insecure credentials



Dynamic Software analysis

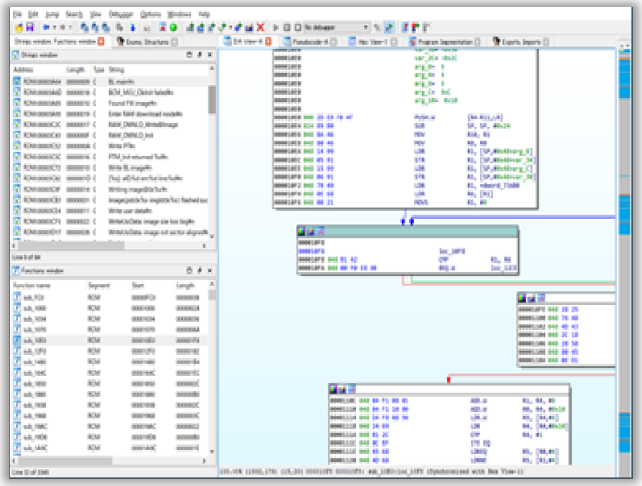
- Analyze running processes on ECUs

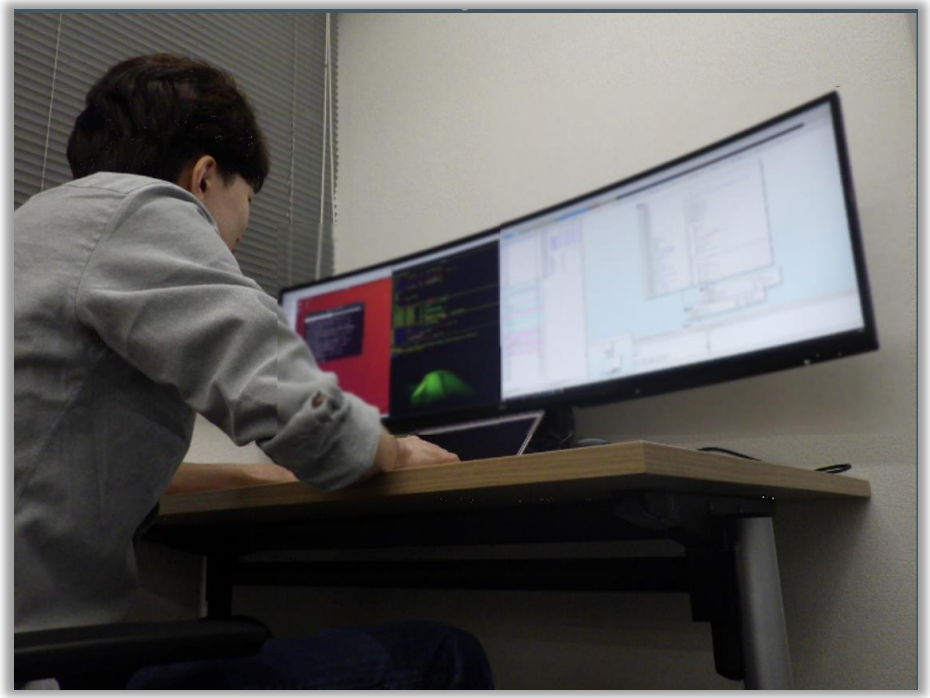
- The dynamic analysis tells us a lot of things
  - Running processes on ECU
  - Behavior and logs during service running
  - Monitoring behavior differences with the config modified



Static Software analysis

- Analyze binary files (Firmware, Software,…)
- Input: Firmware (or unlocked ECU)
  - We can extract firmware from unlocked ECU
    - (Sometimes) we can get it from locked ECU
- Firmware is good alternative of source code and debug console
- The binary file tells us a lot of things
  - Well-known vulnerabilities on OS/OSS/Off-the-shelf products
    - Better alternative to the vulnerability scanner
  - Debug port and service
    - Better alternative to the port scanner
  - Insufficient Security Configuration
  - Hardcoded key, password, certifications
  - Supplier’s debug port…





Wrap up for test methods

Test method		Target elements	Check points
Interface test	Remote interface	Cellular etc.	Check functions and packets over cellular network to find vulnerabilities of cellular dependency.
	Adjacent interface	Wireless LAN, Bluetooth etc.	Check functions and packets over cellular network to find vulnerabilities of wireless Lan and Bluetooth dependency.
	Local interface	USB, CAN, Ethernet etc.	Check functions and packets over cellular network to find vulnerabilities of USB, CAN, and ethernet dependency.
	Physical interface	JTAG, UART, SPI etc.	Check signal on JTAG, UART, SPI and access them to extract firmware and credentials.
Software test	Dynamic analysis	OS/OSS/Off-the-shelf, Application	Analyze running processes on ECU to know behavior and their vulnerabilities.
	Static analysis	OS/OSS/Off-the-shelf, Application	Analyze binary files to find vulnerabilities

How much risks are there from whole “vehicle” perspective

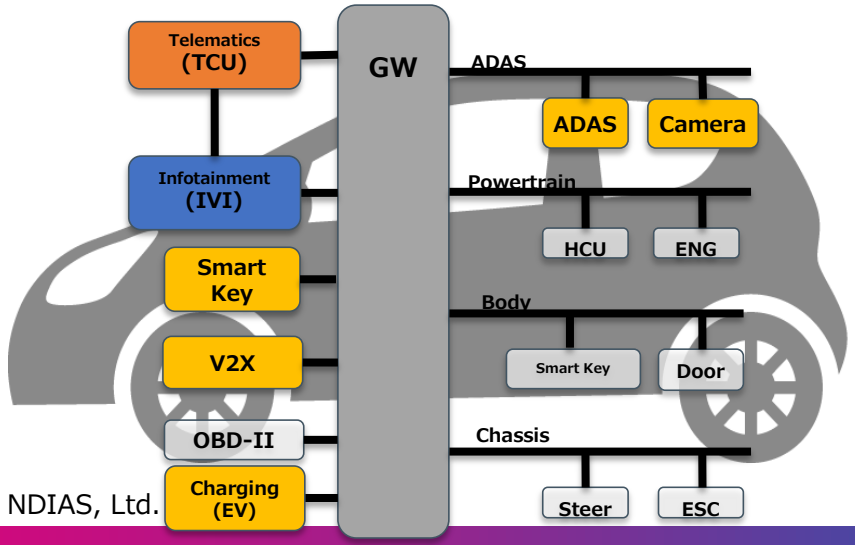
- Risk scores we use calculated by “Damage impact” * “Attack feasibility”
  - Damage impact: The impact for **“vehicle”** when the vulnerability is exploited
  - Attack Feasibility: The attack feasibility for **“vehicle”** when the vulnerability is exploited
- Risks of the attack scenarios related to vulnerabilities of internal NW tend to be decreasing
  - Such vulnerabilities are typically found on ECU and CAN-bus behind GW
  - Of course, risks related to components are also important in “defense-in-depth” perspective, however…
- Hope to find more high-risk vulnerabilities
  - Focusing on “vehicle” is one of the approaches to achieve that.

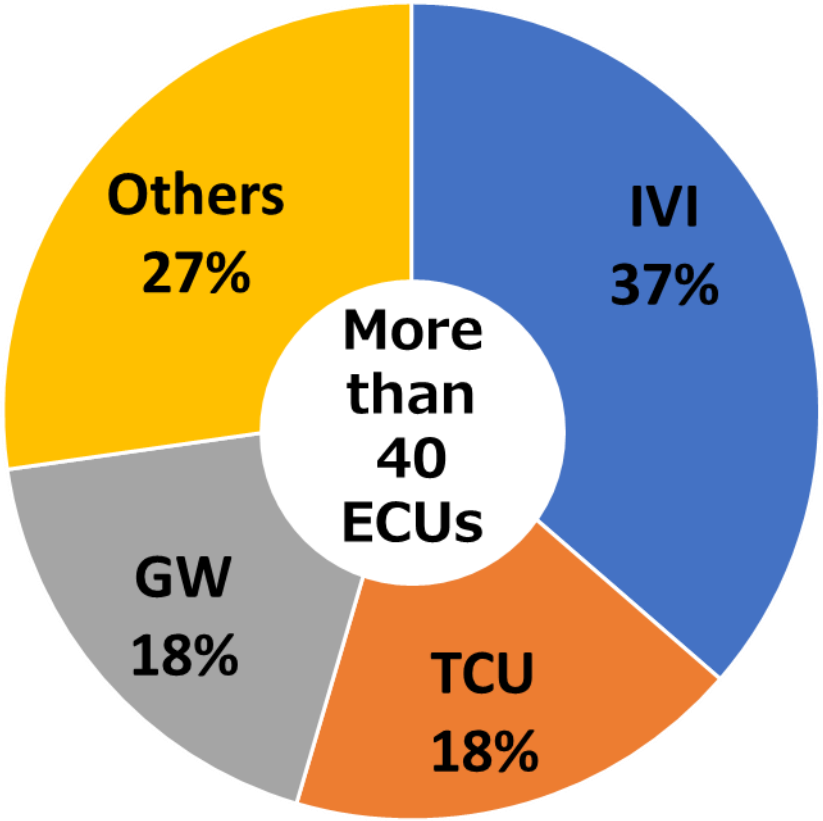
Risk Score Matrix (example)		**Attack feasibility**			
		Very easy	Easy	Moderate	Difficult
Damage impact for vehicle	Severe	High	Medium	Medium	Low
	Major	Medium	Medium	Low	Low
	Moderate	Low	Low	Low	Information
	Negligible	Information	Information	Information	Information

Vulnerability trends in Vehicles

Data set

- Our analysis are based on
  - **More than 300 vulnerabilities** we found
  - **More than 40 ECUs** developed by **more than 10 auto manufacturers and suppliers**.
- The target ECUs are classified to 4 categories
  - #1. IVI 37% of target ECUs
    - It has Adjacent I/F (such as WLAN, Bluetooth etc.)
  - #2. TCU 18% of target ECUs
    - It has Remote I/F (such as Cellular etc.)
  - #3. GW 18% of target ECUs
    - It has local I/F (such as OBD-II etc.)
  - #4. Others 27%
    - Including ADAS, Smart key, Charging(EV), V2X…

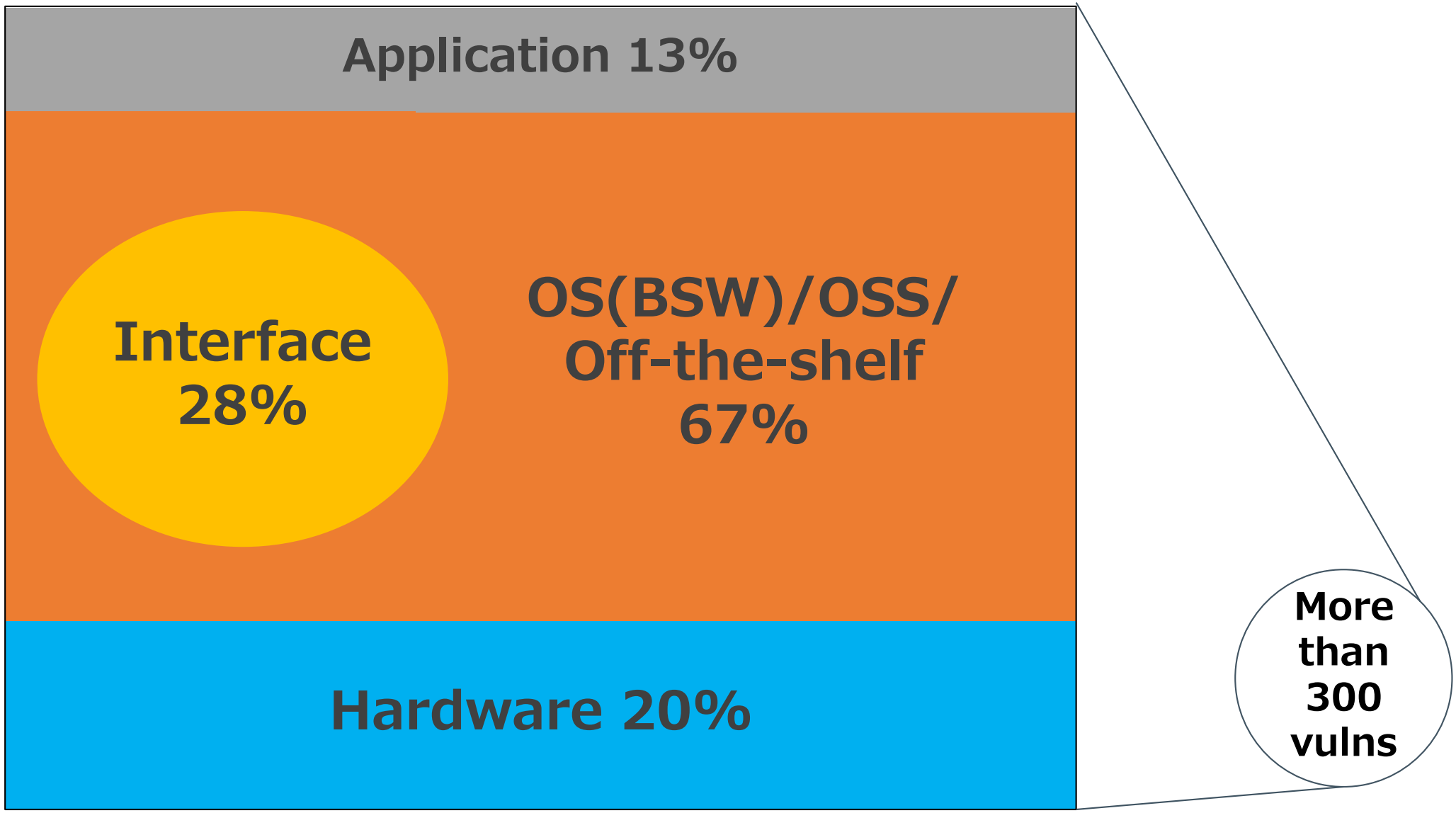




Proportion of target ECUs

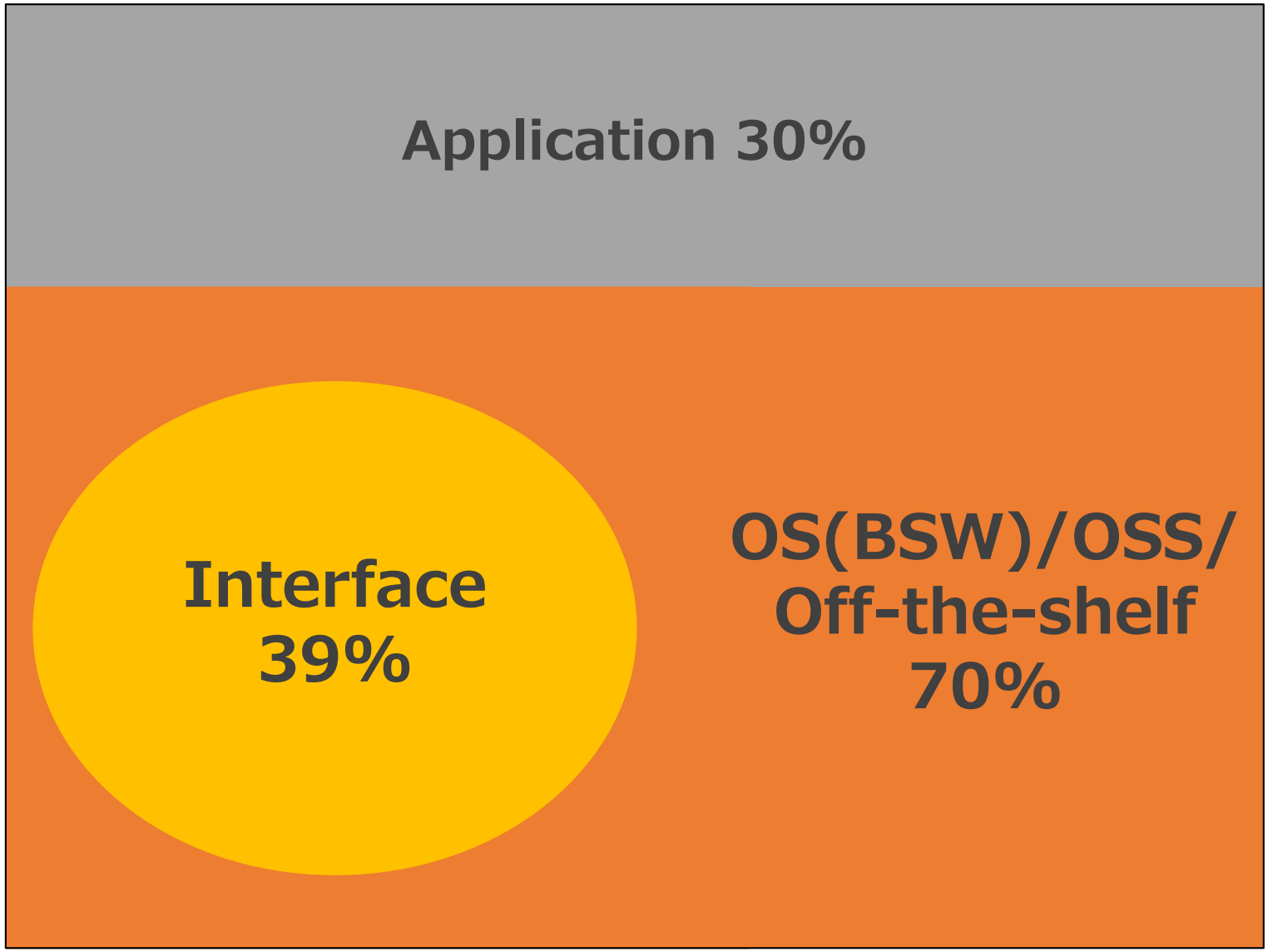
Around 70% of vulns are in OS (BSW), OSS, Off-the-shelf

Proportion of vulnerability detected locations in the ECU structure

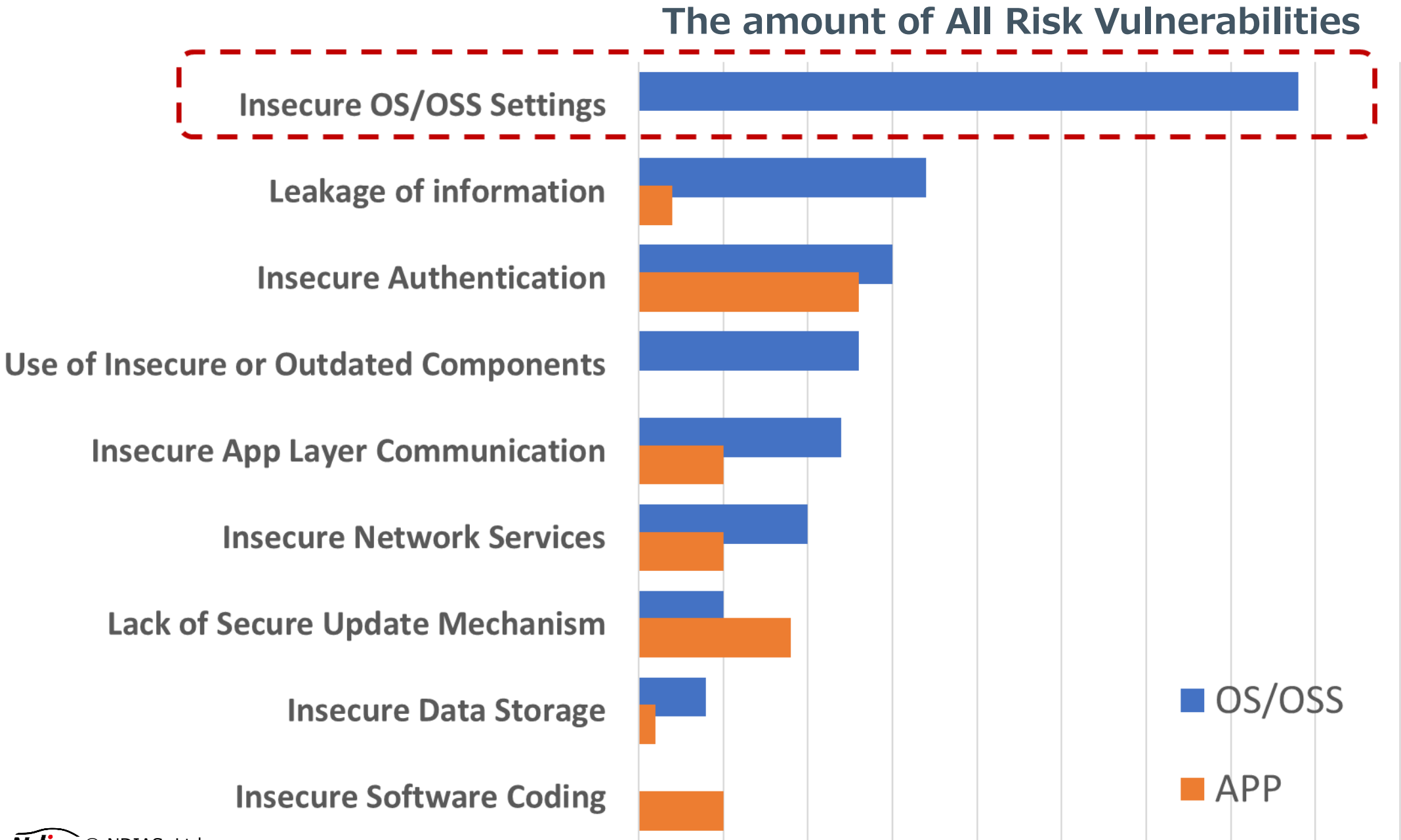


Most of high risk vulns are in Software (not hardware)

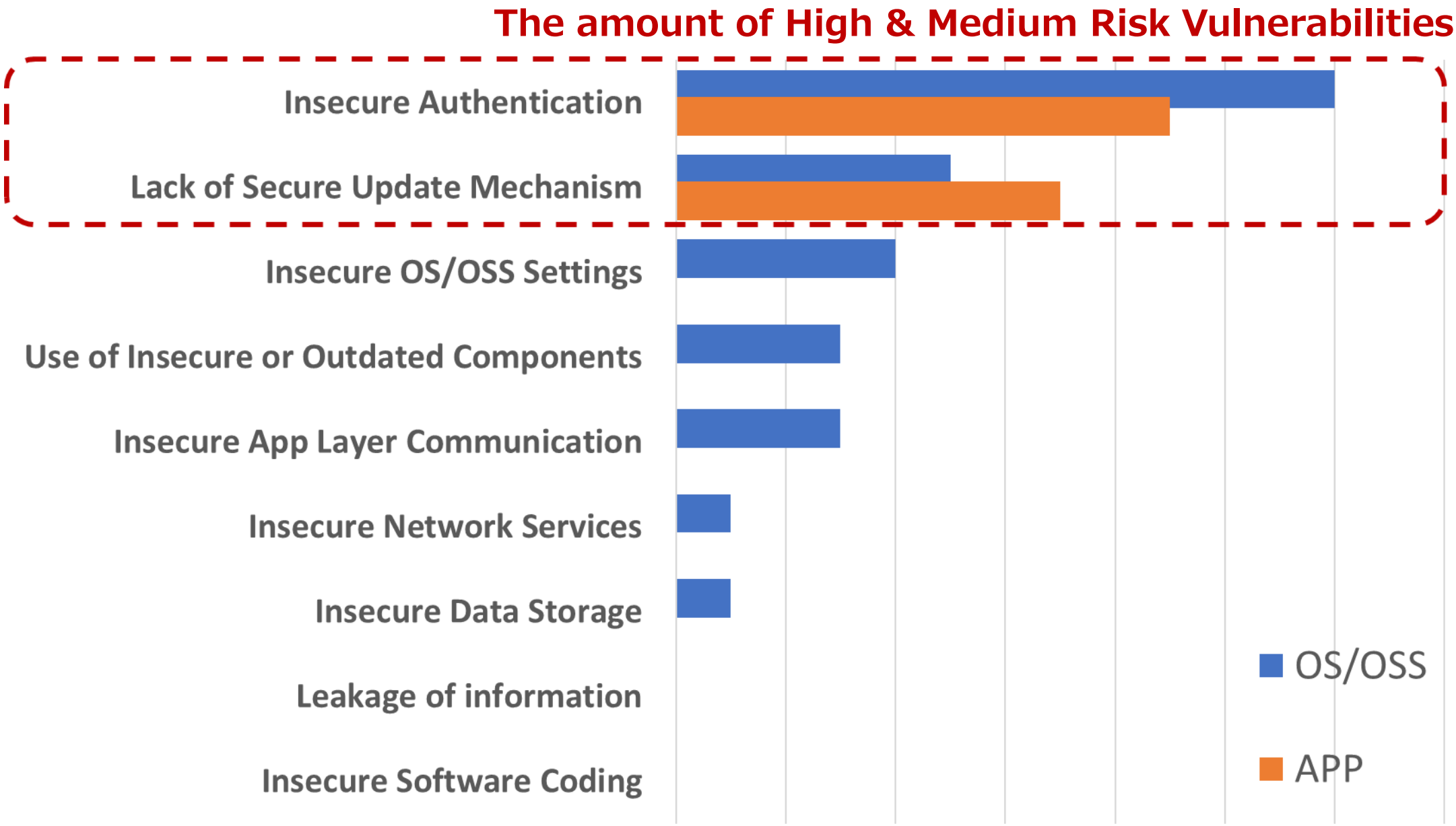
Proportion of vulnerability(only high risk) detected locations in the ECU structure



The most often detected is “Insecure OS/OSS settings”

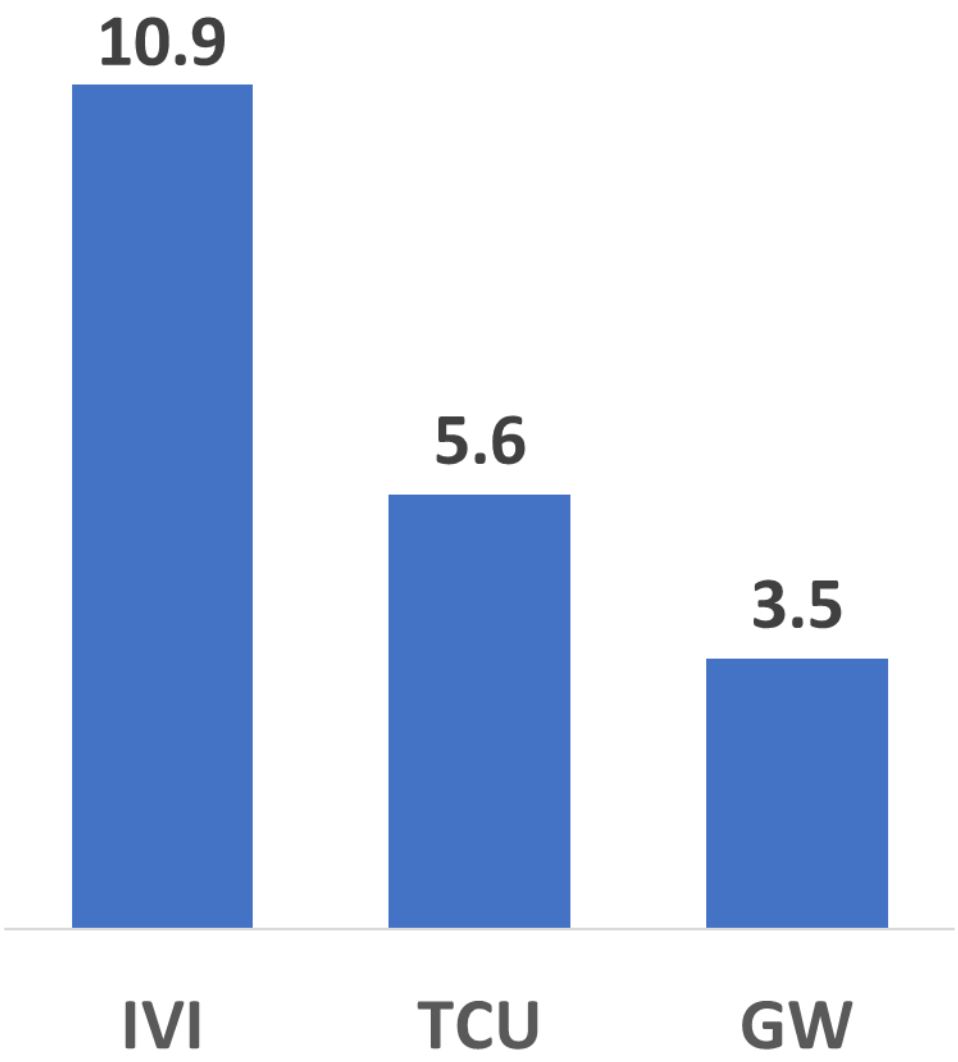


Authentication & SW Update security are “very” important

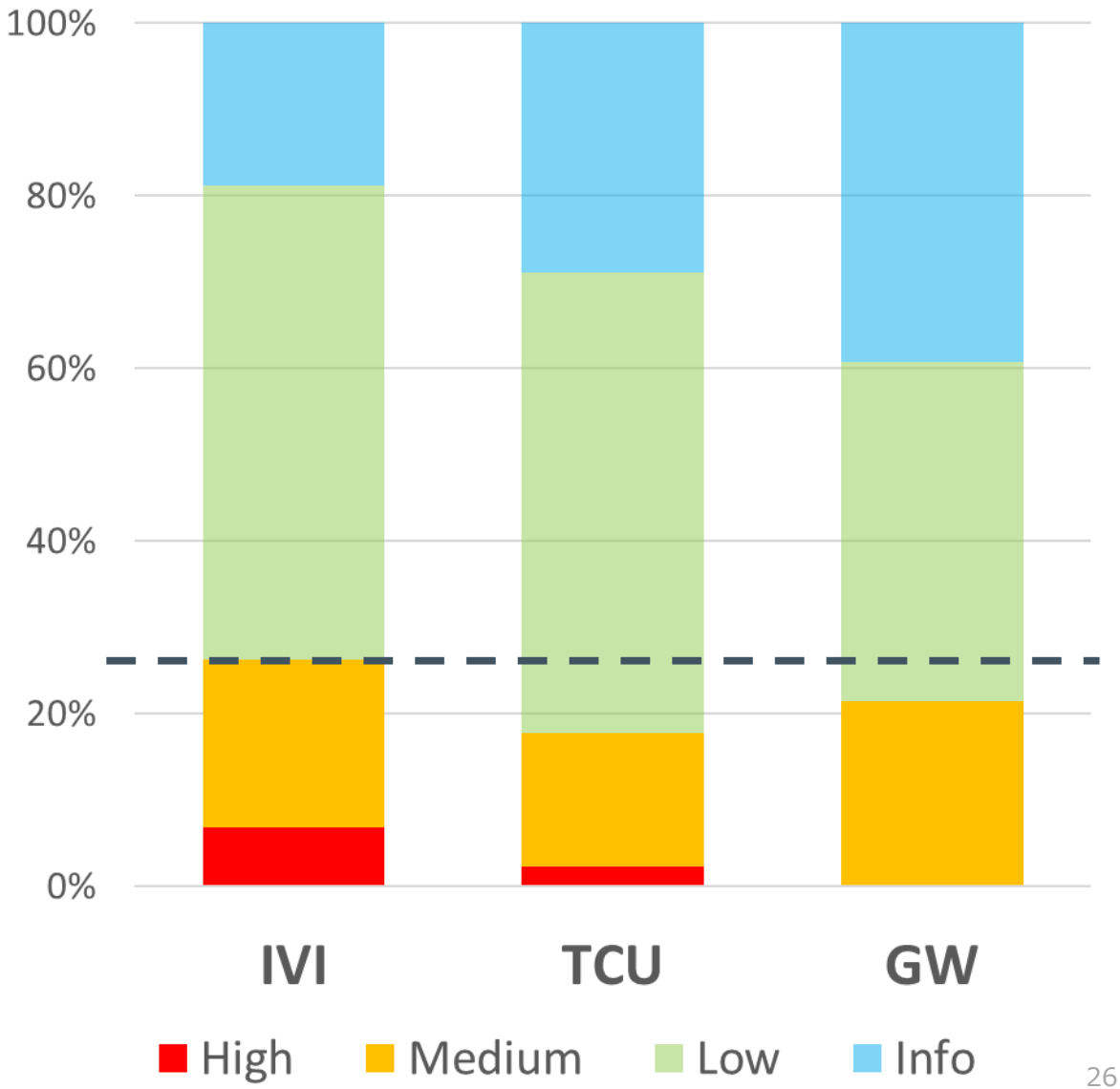


More complex ECU, more vulns

Average number of vulnerabilities

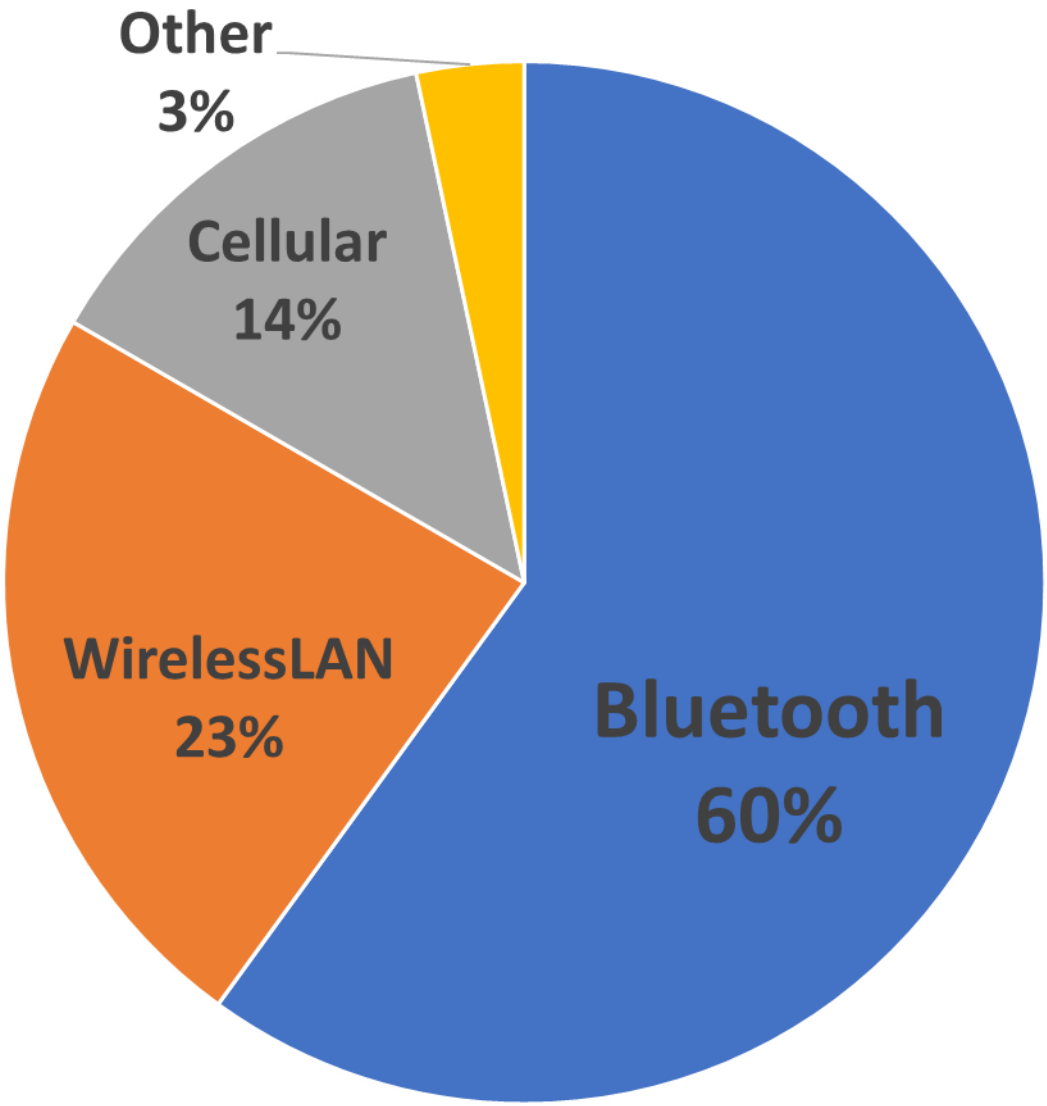


Proportion of vulnerabilities per risk score



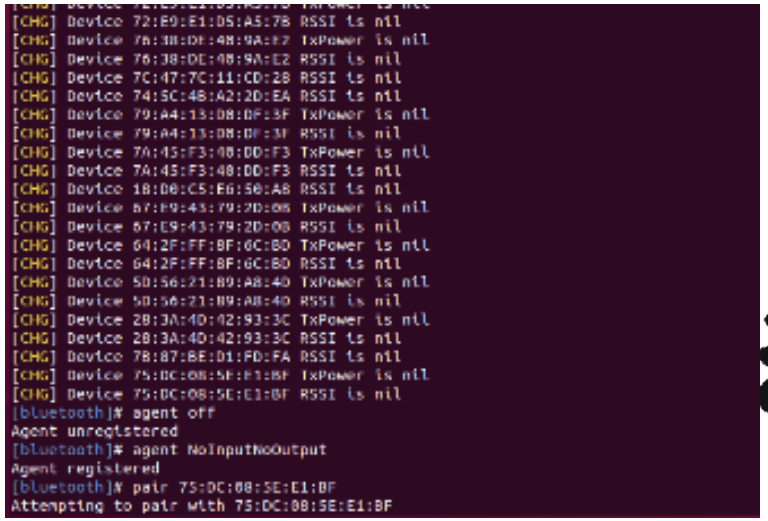
Bluetooth vulnerabilities are the most detected in remote attack I/F

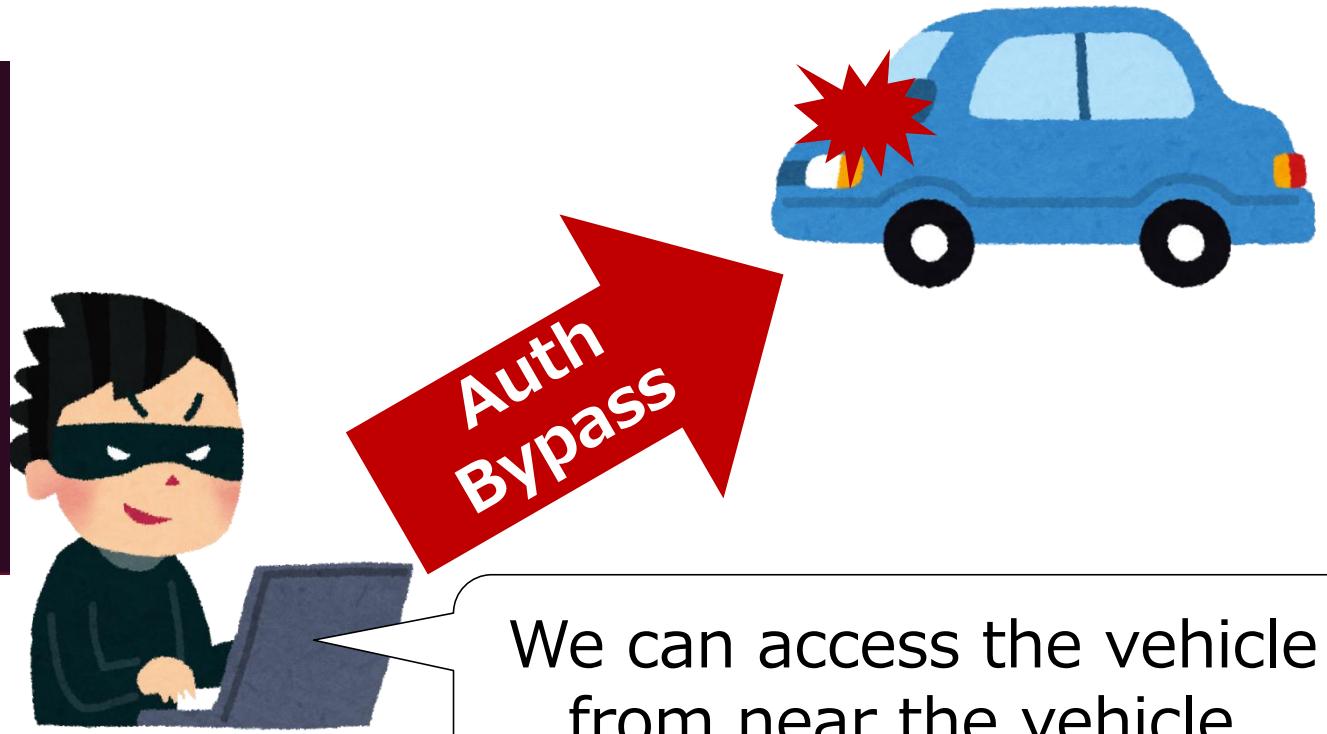
Proportion of vulnerabilities we found in remote attack I/F



Authentication Bypass Vulnerability (Bluetooth)

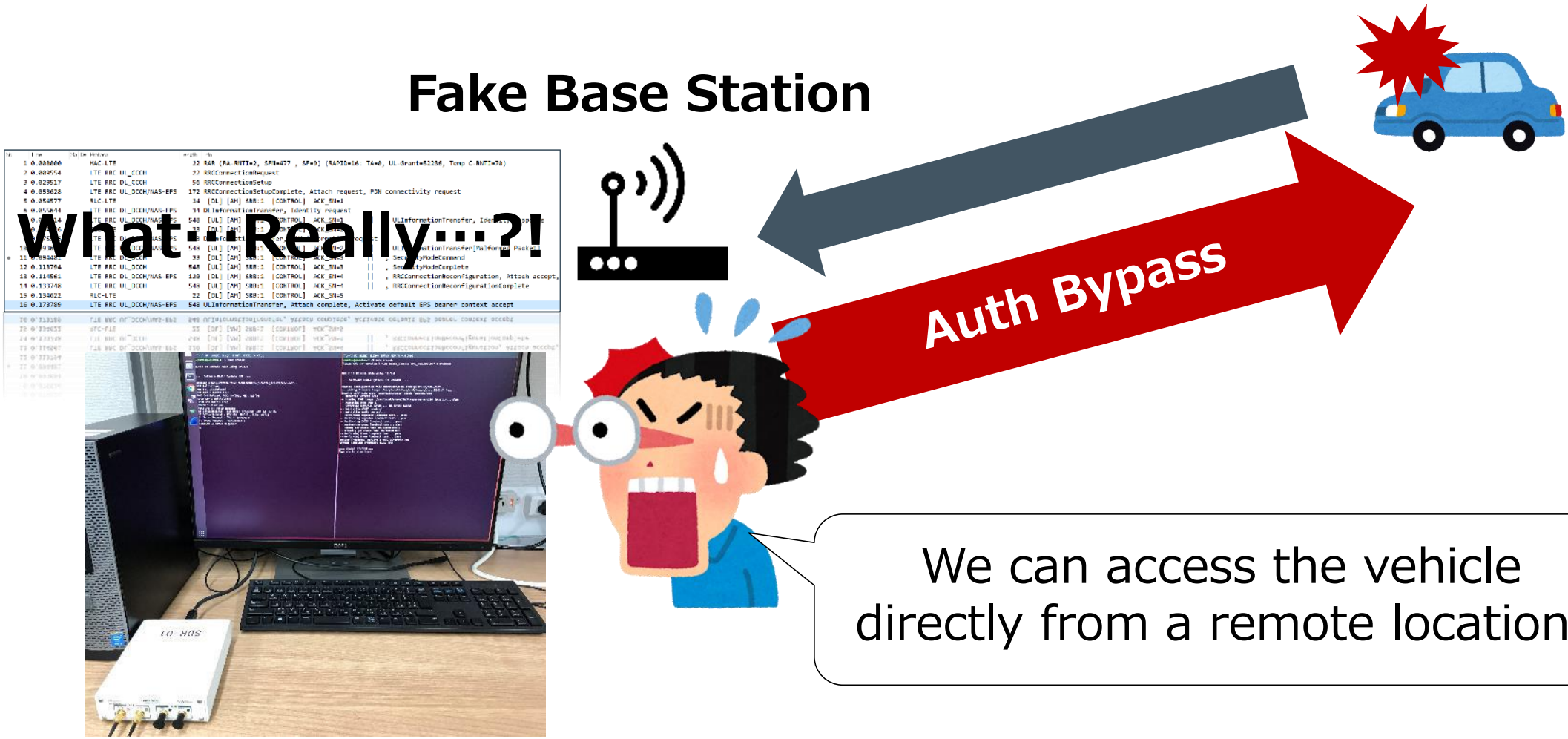
- Authentication Bypass
  - PIN Mode (0000,1234,9999,etc…)
  - CarsBlues
  - IO Capability:NoInputNoOutput





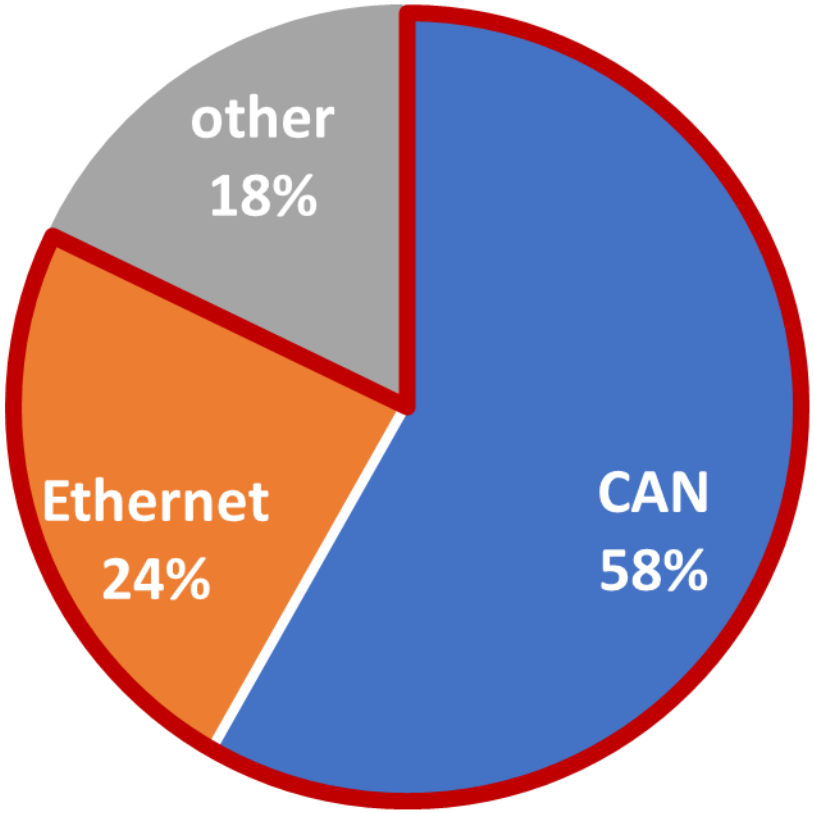
Authentication Bypass Vulnerability (Cellular I/F)

- A very rare case, but a very serious problem.
  - Using the test code for auth bypass based on srsLTE.



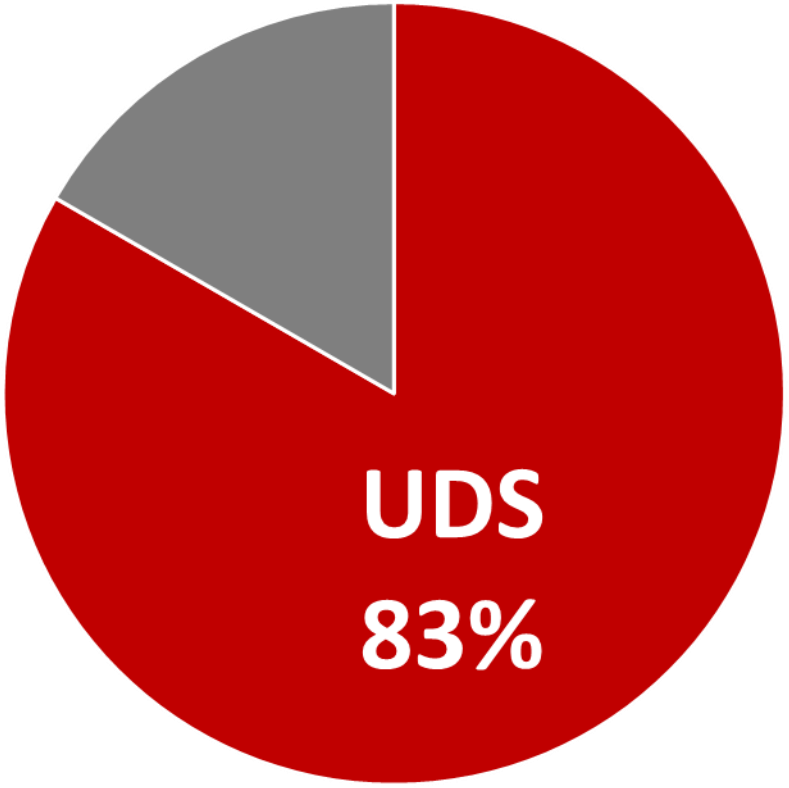
83% of High & Medium vulnerabilities of local “NW” I/F are related to UDS

Vulnerabilities in local I/F



Focus on
High & Medium
vulnerabilities

Type of Vulnerability in local “NW” I/F

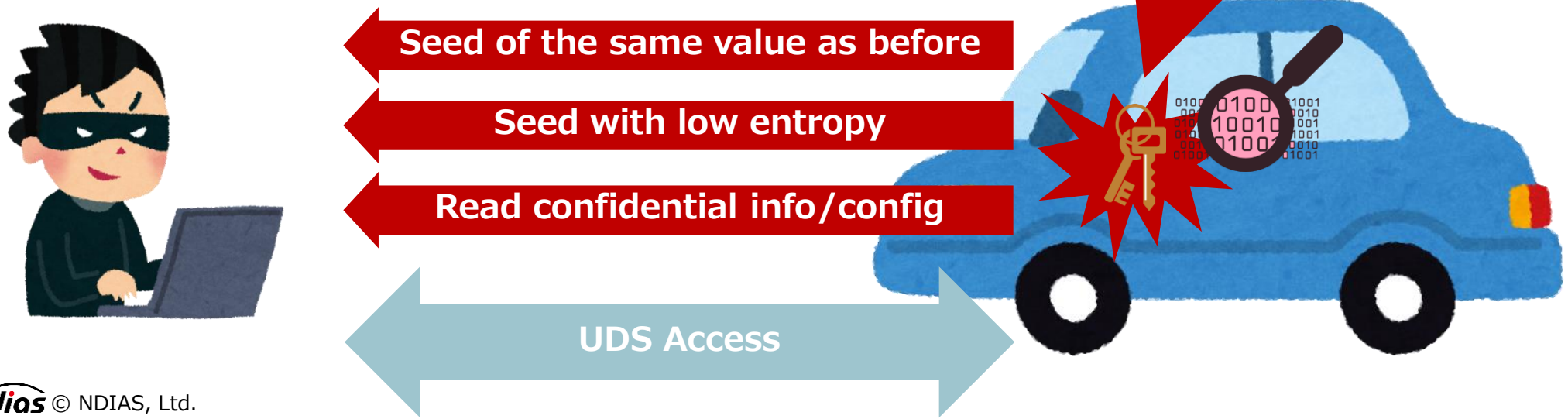


Broken SecurityAccess protection (CAN/Ethernet)

- What matters is “UDS” protocol
  - 2nd gen cars start to support “Message Authentication Code” to secure messaging via CAN/Ethernet.
  - By contrast, for UDS, “security access” is still primary countermeasure for security.

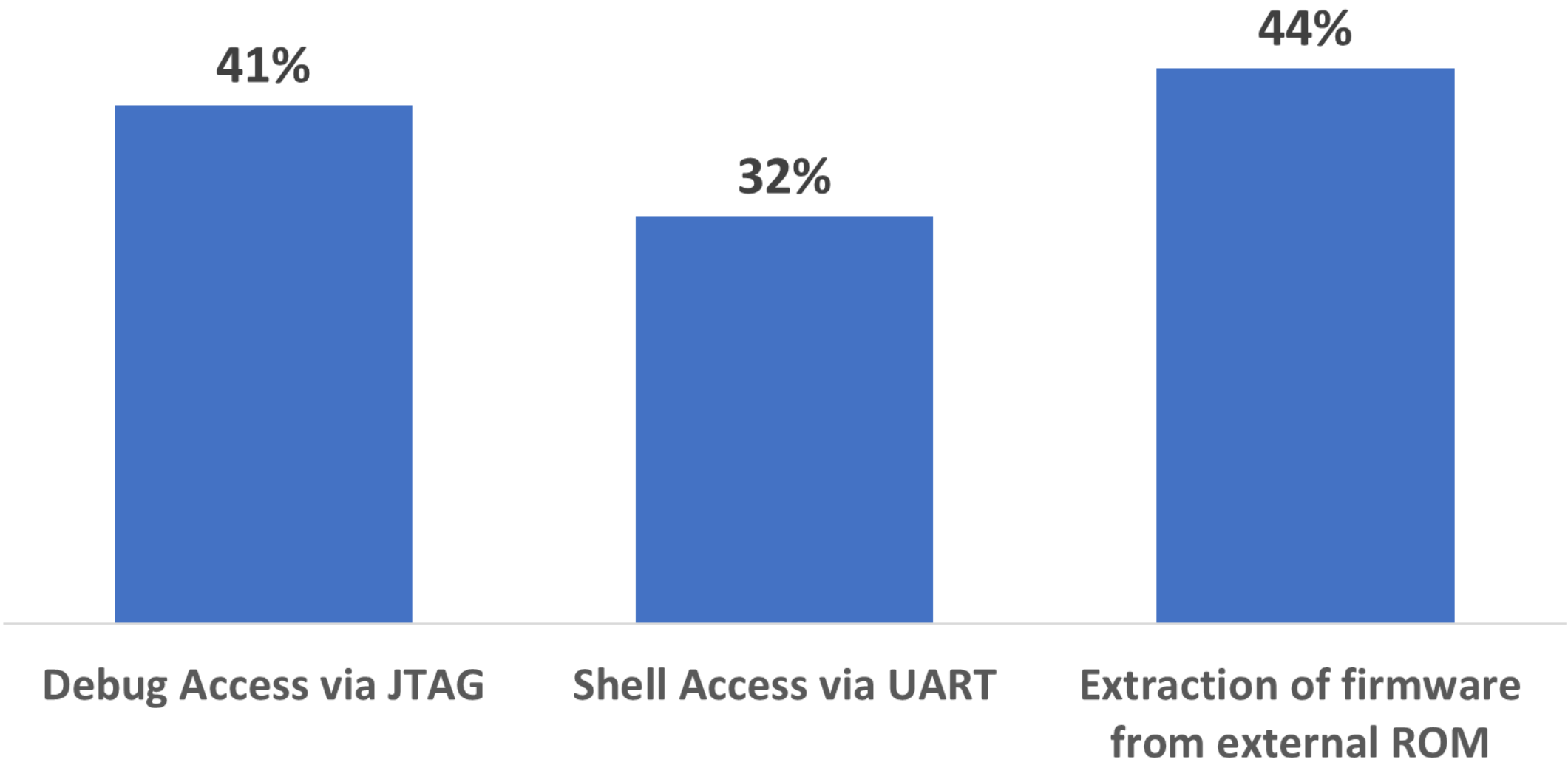
- Improper specifications or implementations of security access:
  - Lack of entropy for “seed” generation
  - After resetting ECU, getting same random sequence
  - “Read by ID/Address” is accessible to secrets without security access

- Improper protection of credentials:
  - Underutilized hardware security module

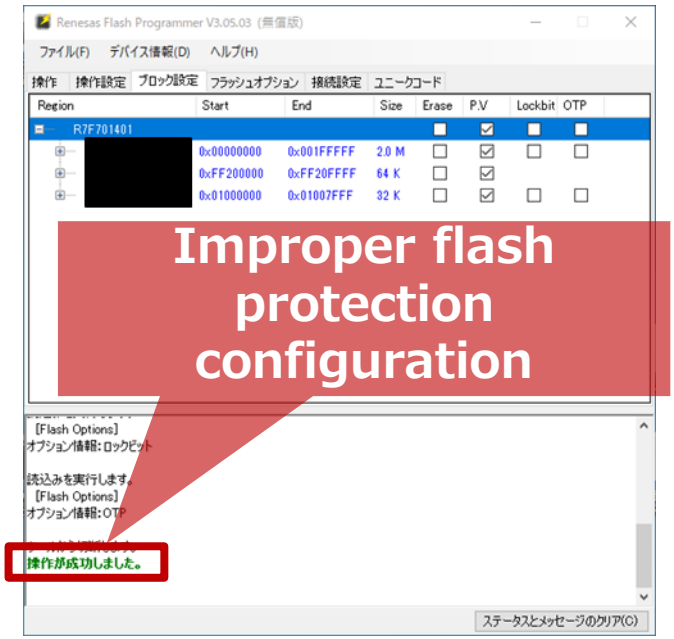


Some ECUs are still unlocked - physical debug interfaces

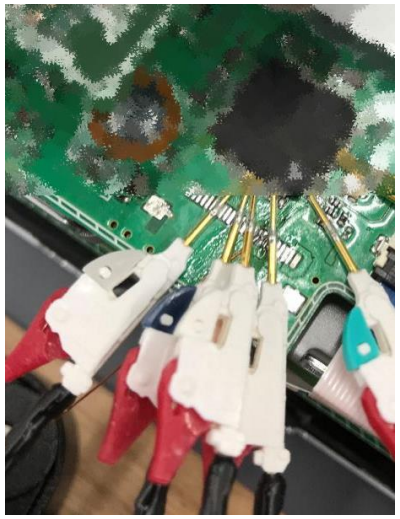
Proportion of ECUs with an unlocked physical debug interface

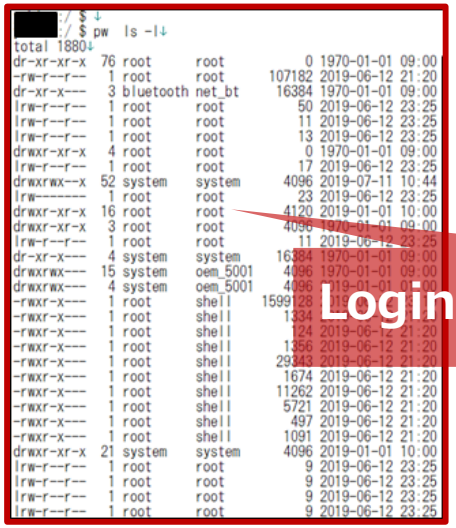


Physical debug interfaces(such as JTAG, UART) are unlocked



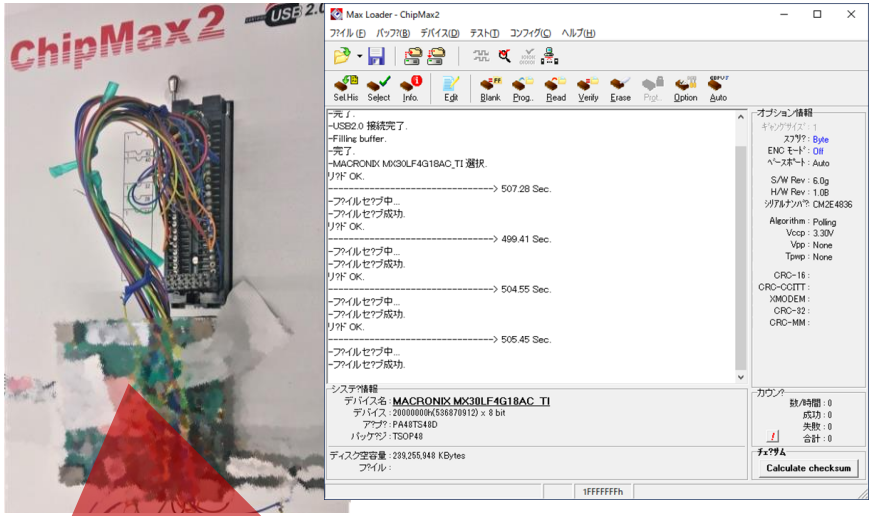
Improper flash protection configuration





Login to console via UART

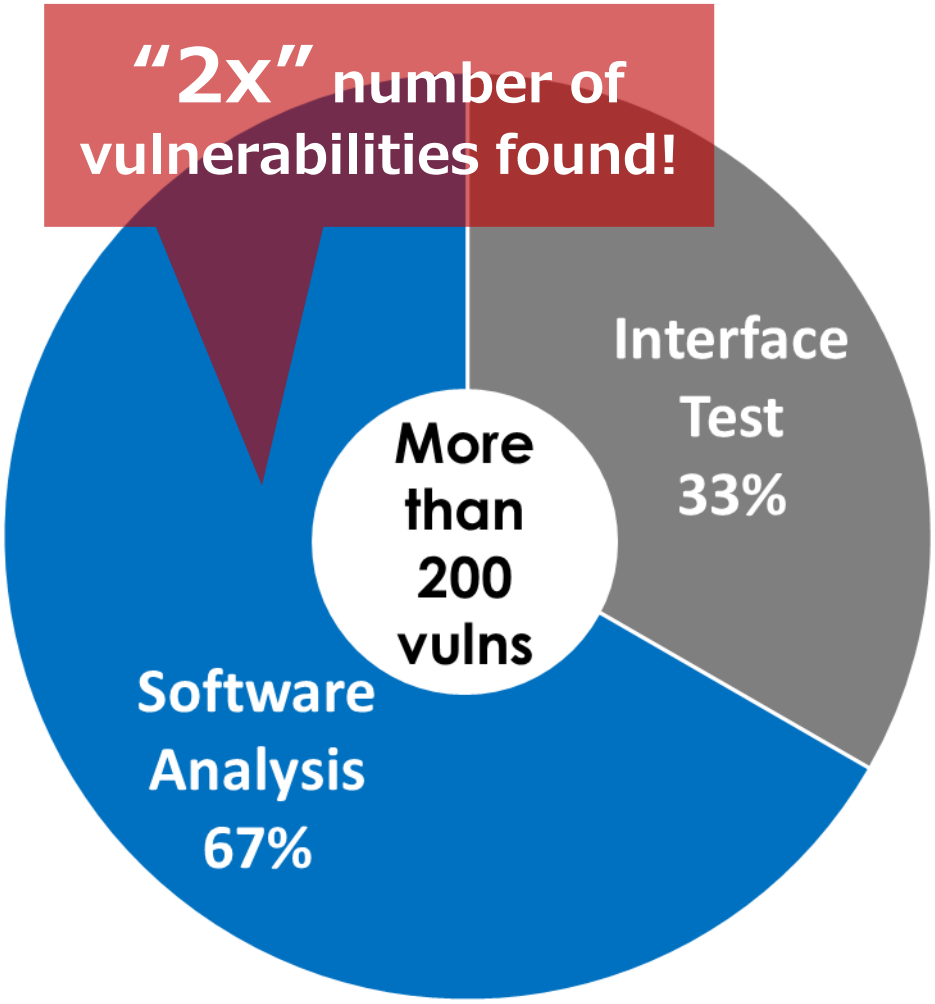
Locked, but improper password use results in firmware extraction



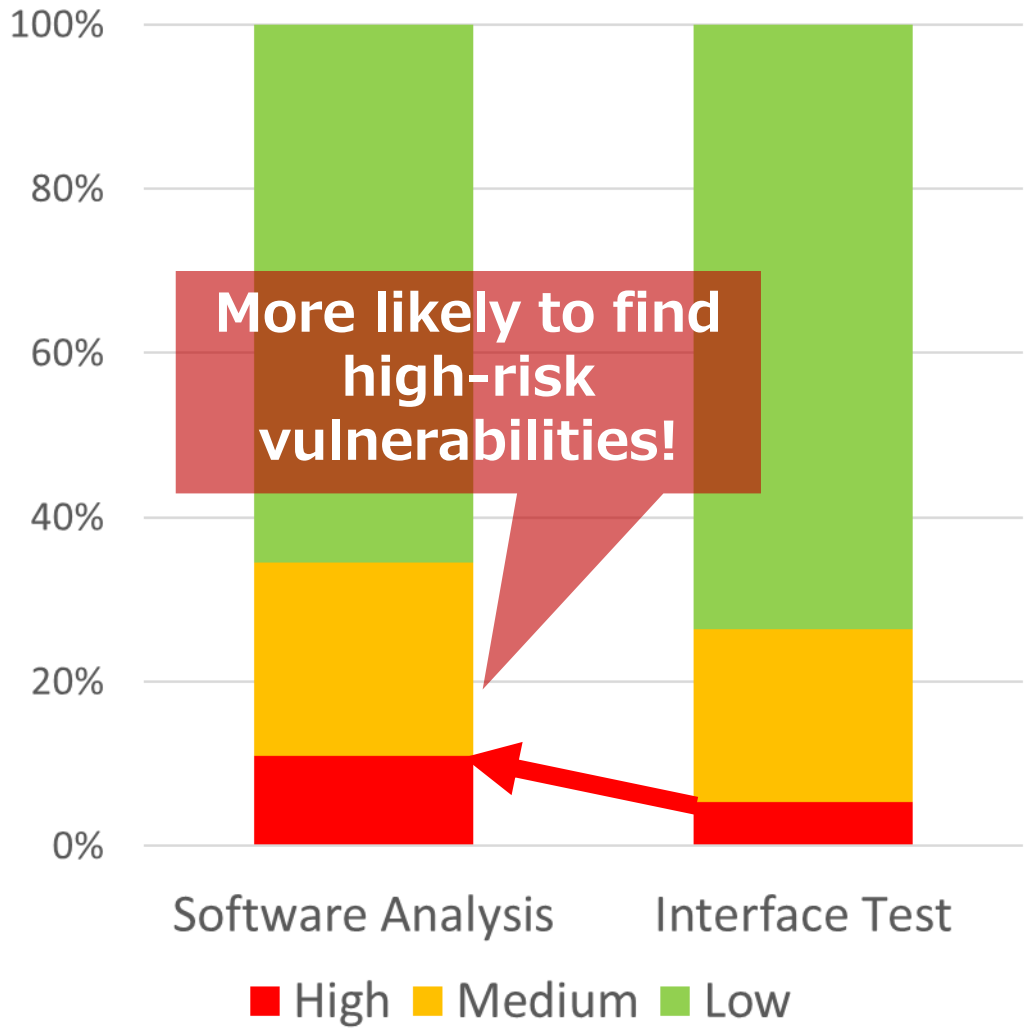
Extract firmware from eMMC

Software analysis is a powerful method to find high-risk vulns

The proportion of nums of vulns found by each test

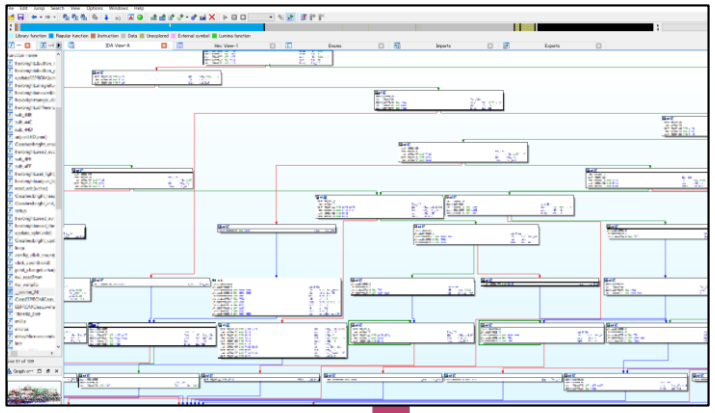


The proportion of the risk score of vulns found by each test

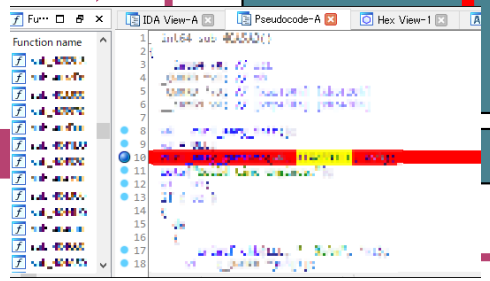


How to analyze software

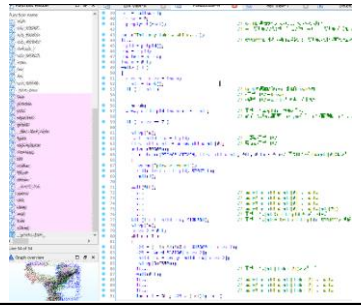
- 1. Identify the target code in huge amount of ECU binary
- 2. Analyze the data flow and security functions
- 3. Dig deeper and deeper



① Identify



TLS communication



③Dig deeper

Update process on container

TLS communication
Signature verification
Encryption

② Analyze

Proxy

Update process out of container

2nd signature verification

Container

Com. bus

Host OS

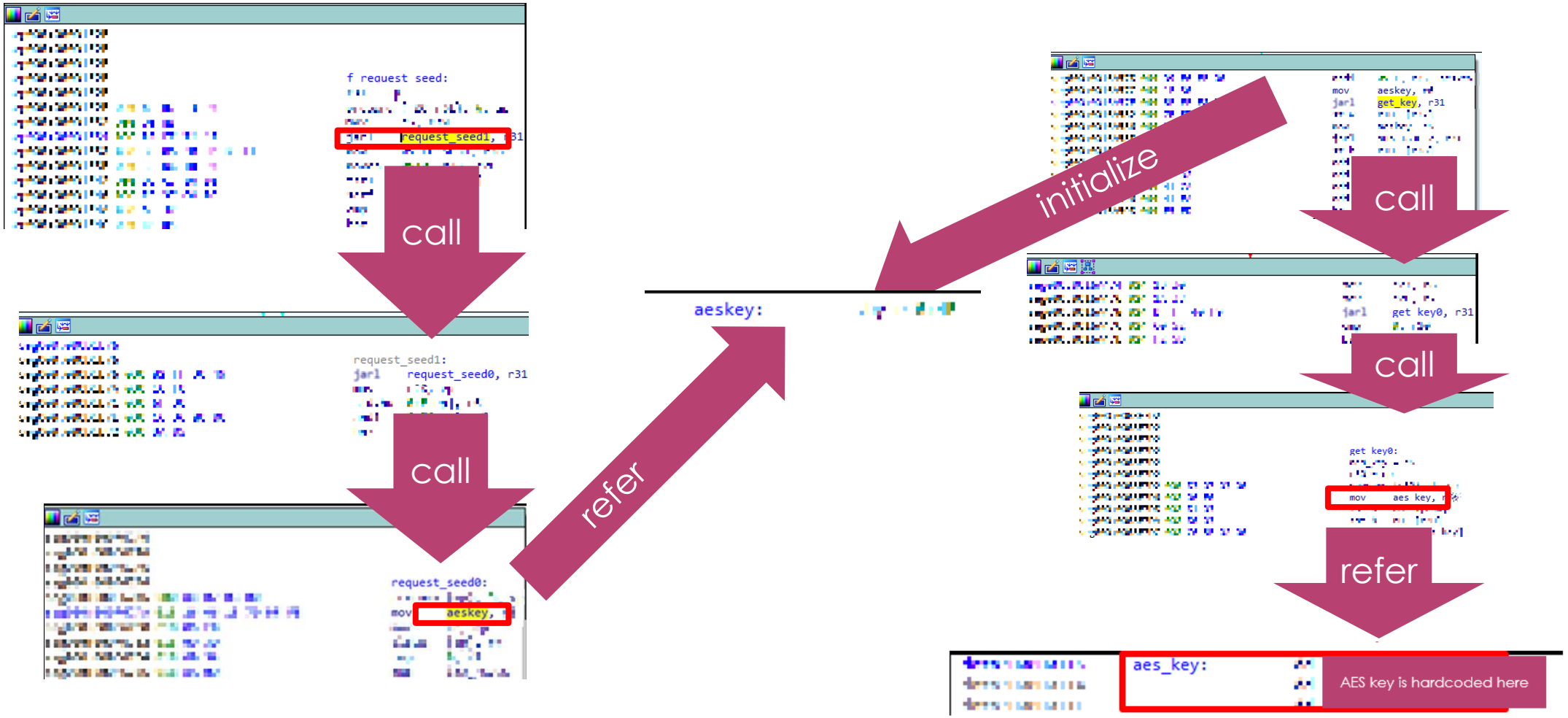
Socket communication

Hardware

Example of software arch.

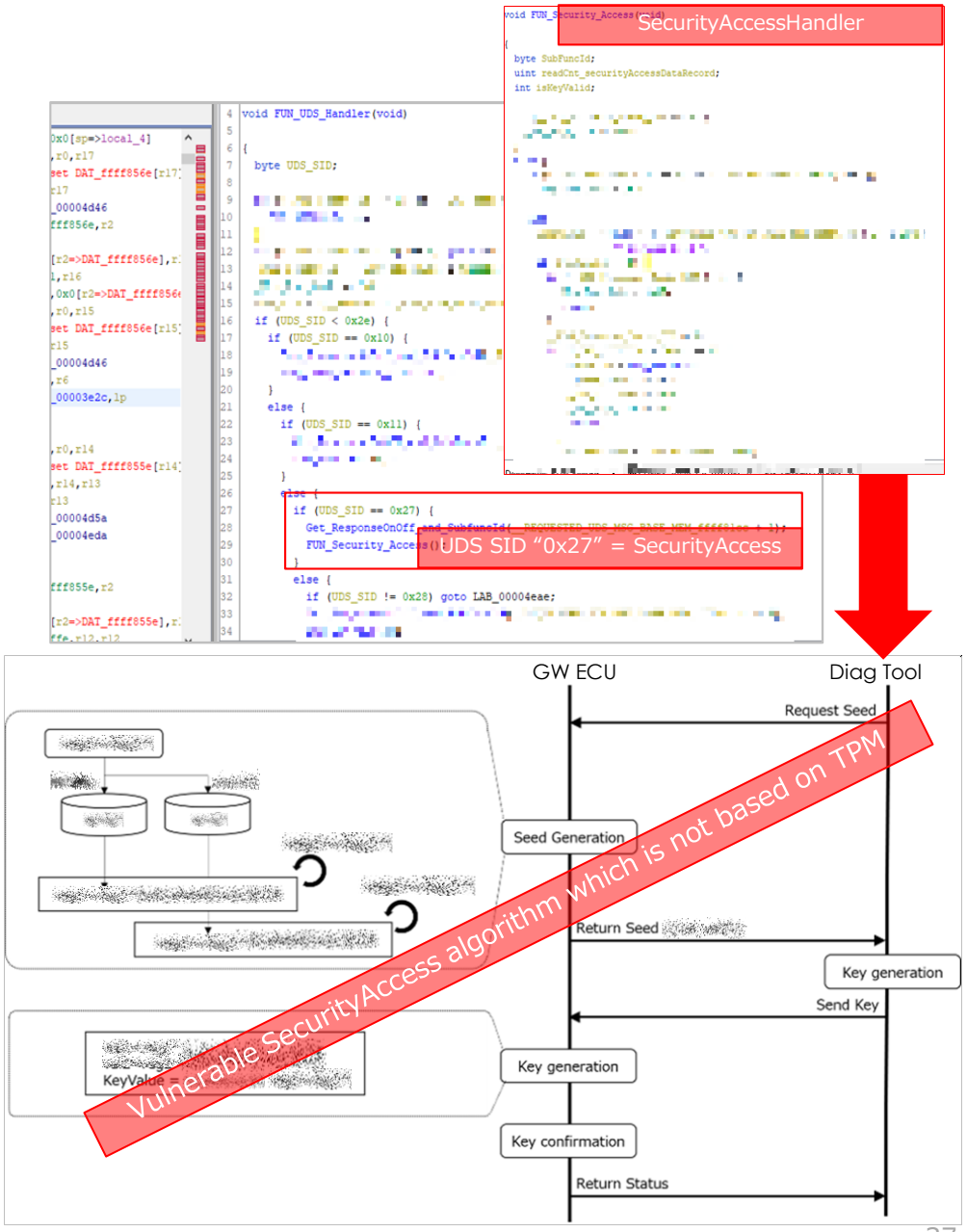
SW analysis example: Insecure key management

- The credentials must NOT be hardcoded
- HSM is now mandatory to protect your private keys

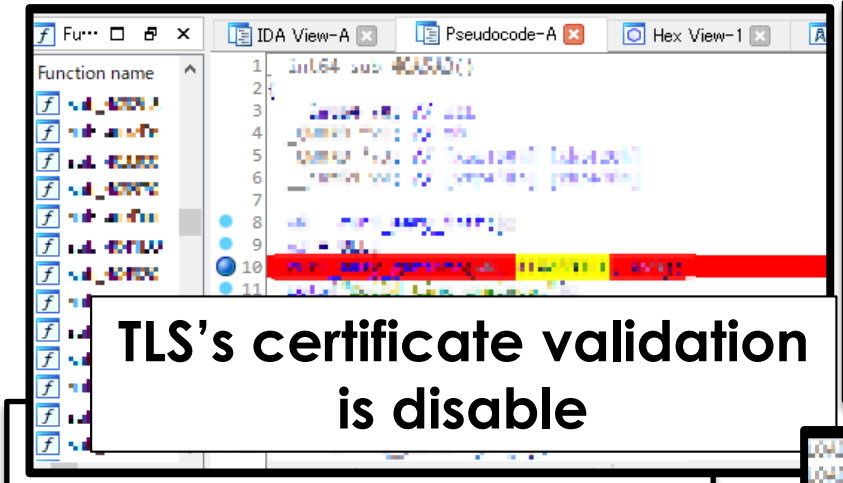


SW analysis example: Insecure “SecurityAccess” algorithm

- “Old generation” MCU is known for the lack of TPM feature.
- An MCU firmware with improper lock was successfully extracted
- Static analysis results in identifying vulnerable “SecurityAccess” algorithm and secret key in the firmware binary.



Software tells us many many things



TLS's certificate validation is disable

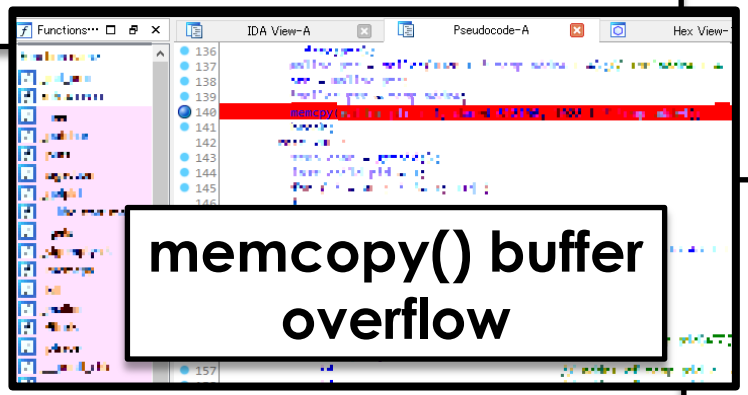
Timing attack to bypass the signature validation

Older version of library is used

Improper implementation of authentication protocol

Hardcoded Secret key

Insufficient entropy of the seed of PRNG generator



memcopy() buffer overflow

Directory traversal

Improper error handling

Insecure sensitive data management

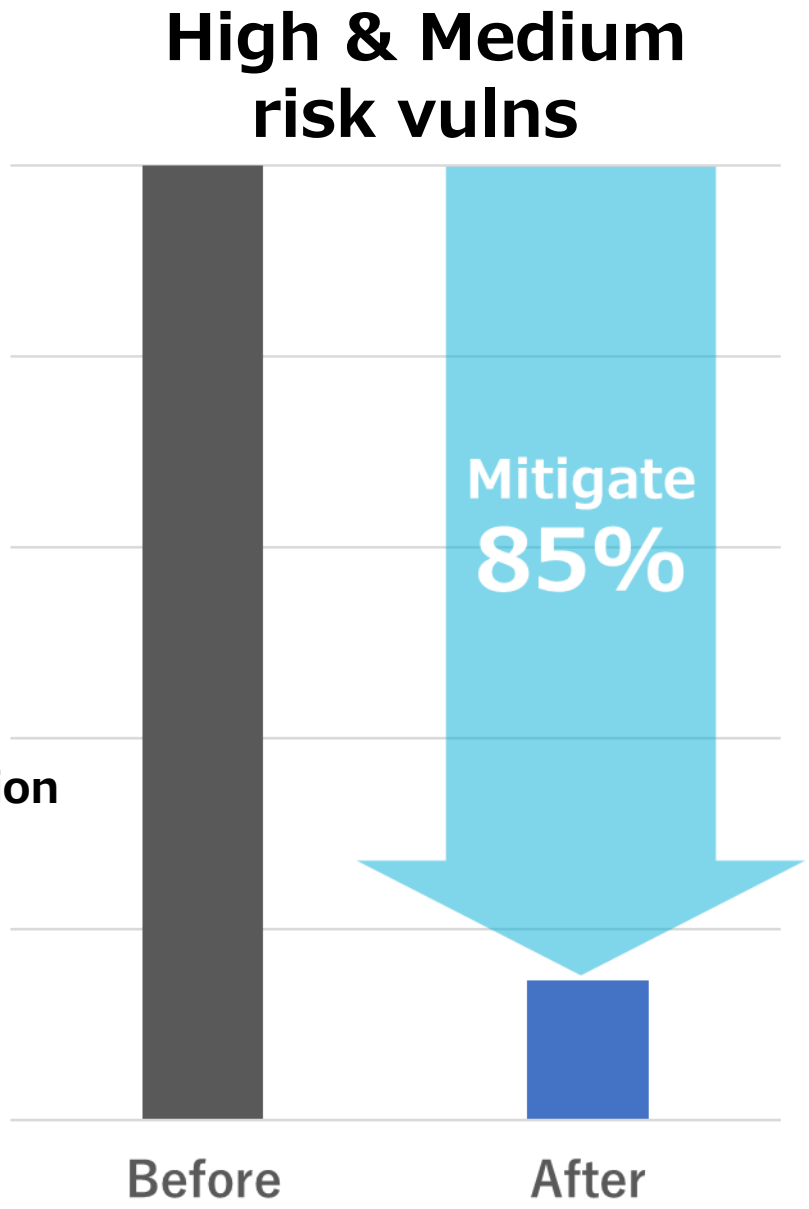
Does 2nd gen car face the cybersecurity threat?

- The answer is “Secure” in most cases.
- It is difficult to realize the threat scenarios for 2nd gen car because
  - 2nd gen car has good architecture from the view of defense in depth
  - 2nd gen car has many security functions.
    - Secure boot, HSM, FOTA, MSG authentications, Good authentication protocol, strictly fire wall…
  - 2nd gen car are installed better hardened OS and off-the-shelf software.
  - 2nd gen car’s application has applied a lot of patches
    - We feel it through software analysis. Developers are good work.

- “Unsecure” cases are very rare.
  - All vulnerabilities we found must be fixed before commercial version.

5 solutions can mitigate 85% of high & medium risk vulns

1. **Keep credentials and private keys in secure element such as Trust Zone, TPM and HSM**
2. **Verify the signature of firmware data strictly**
3. **Use appropriate PRNG and cryptography**
4. **Pay attention to three things about the OS/OSS/off-the-shelf products**
   - **Use OSS properly**
     - e.g., never skip the server certificate verification for TLS.
   - **Software version control/Managing software composition**
   - **Secure configuration of the OS/OSS/off-the-shelf**
     - e.g., Firewall, Permissions, etc..
5. **Lock the physical debug interfaces such as JTAG and UART properly**



Conclusion & Perspective

Conclusion & Perspective

- Q. Is it easy to hack the car?
  - A. For 2nd gen car, answer is “Difficult”.
  - We can find some vulnerabilities through our work.
  - However in a lot of cases, they didn’t realize the high-risk threat scenario.

- Q. What test is good?
  - A. Software analysis
  - But it needs the firmware (of unlocked ECU) and test time
  - Recommend the combined test

- Q. What is need to develop secure cars
  - A. Good relation between OEM, supplier and security service provider.
  - New security function is definitely good. But relationship is more important.

Thank you !

- **For more information, visit & ask us**

  https://ndias.jp/

  info@ndias.jp

- **Follow us on**
  - Ryosuke Uematsu @tansokun920
  - Shogo Nakao
  - Ryoichi Teramura @trmr105
  - Tatsuya Katsuhara @kthrtty