PLAXIDITYX

FORMERLY ARGUS

Tales from a Penetration Testing Team

Insights from zero-day automotive vulnerabilities discovered in recent years

Amit Geynis

Security Researcher & Team Leader

PLAXIDITYX
GO EVERYWHERE

PLAXIDITY X
GO EVERYWHERE

Introduction

In This Talk

- Found Pre-SOP & already fixed by the responsible parties
- Some of the details were **redacted** to ensure confidentiality of the involved parties

PLAXIDITYX
GO EVERYWHERE

Vulnerability #1:

Arbitrary Remote Code Execution over CAN

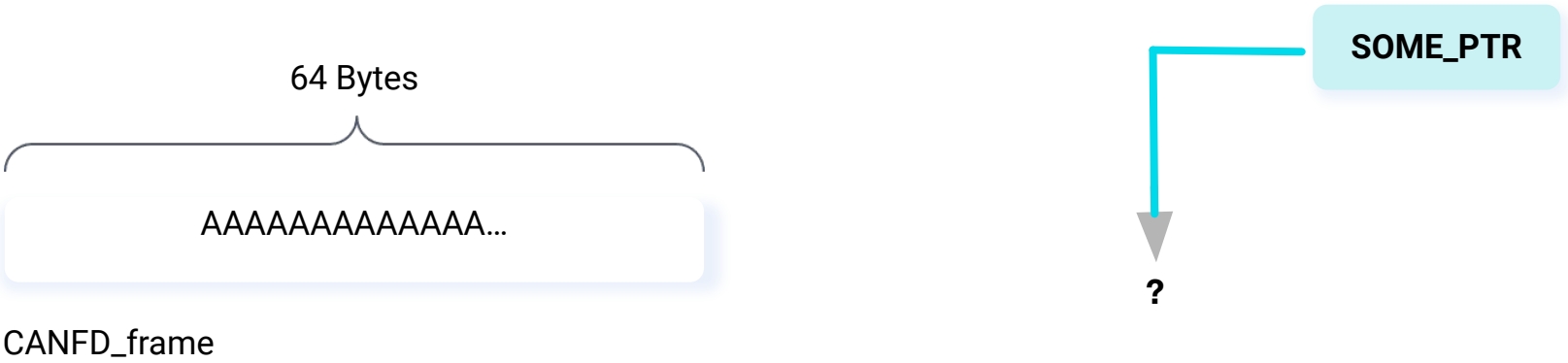
PLAXIDITYX
GO EVERYWHERE

Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)

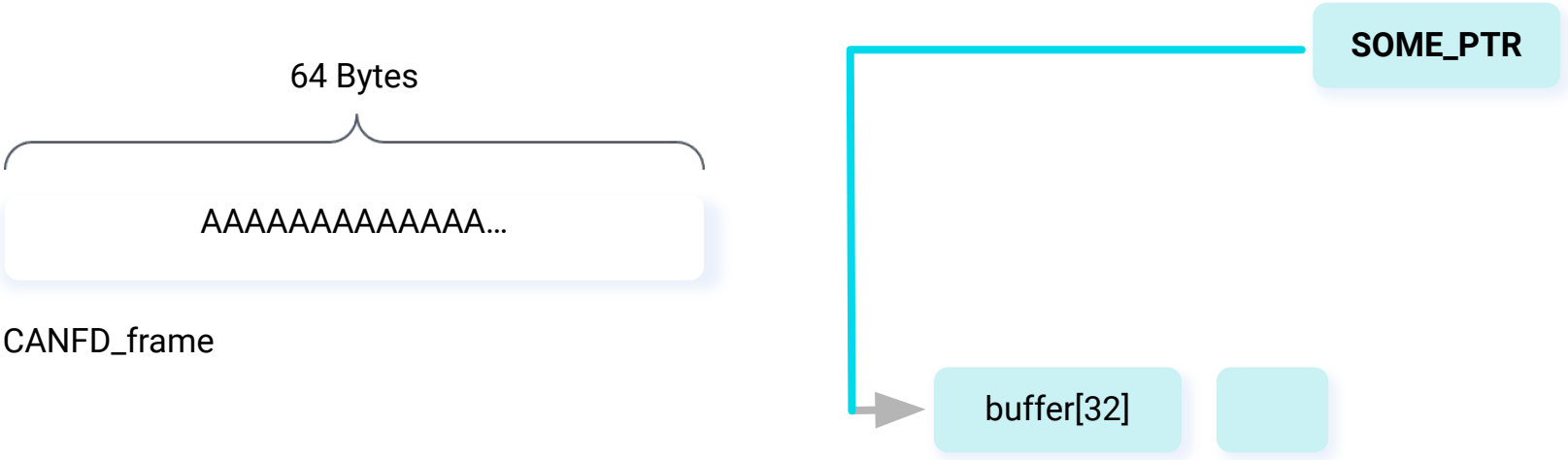
Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)



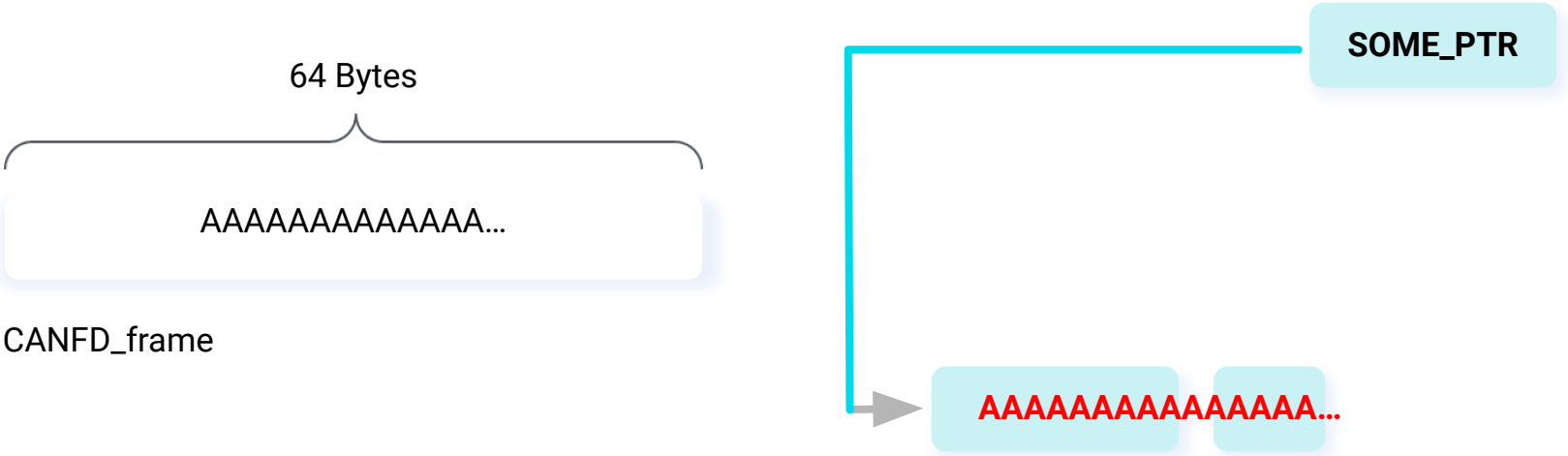
Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)



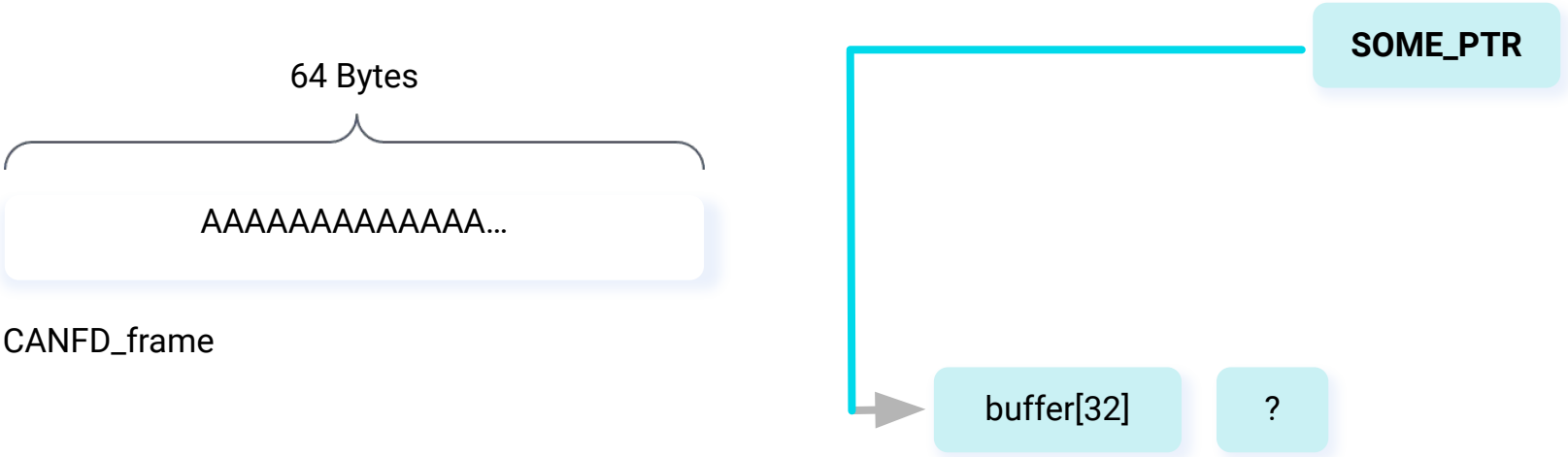
Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)



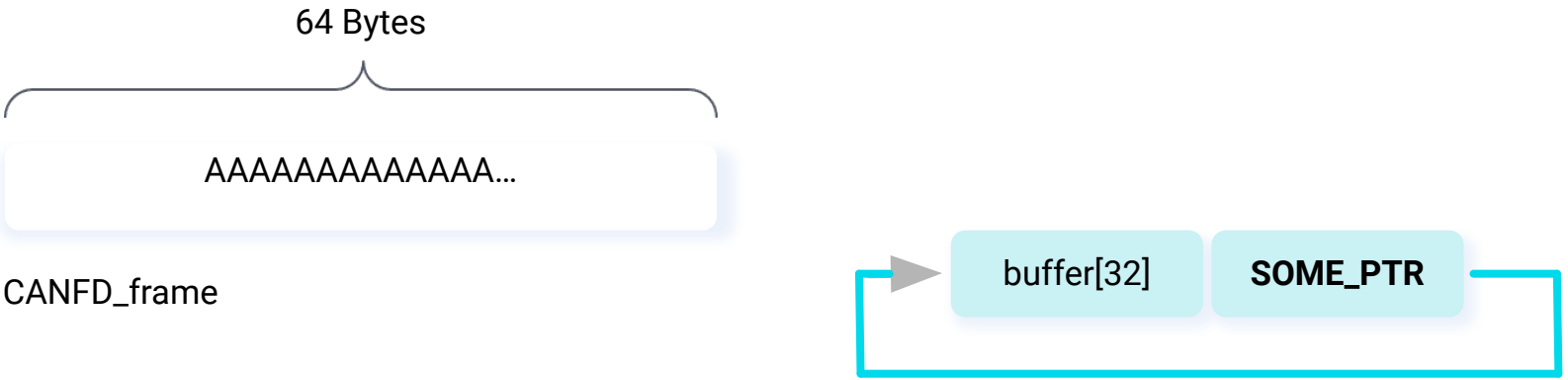
Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)



Vulnerability #1

memcpy(**SOME_PTR**, CANFD_frame, 64)



Vulnerability #1

memcpy(**AAAAAAAA**, CANFD_frame, 64)

64 Bytes

AAAAAAAAAAAAA...

CANFD_frame

AAAAAAAAAAAAAAA...

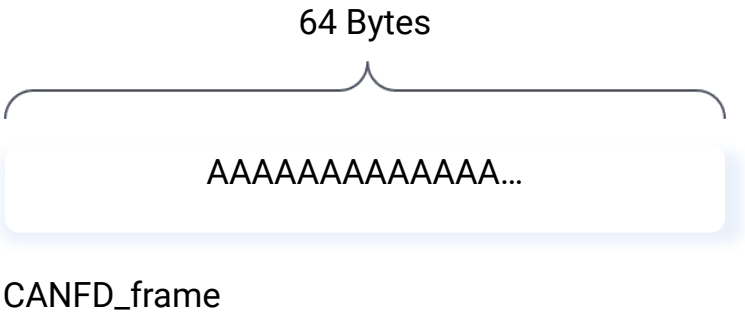
Vulnerability #1

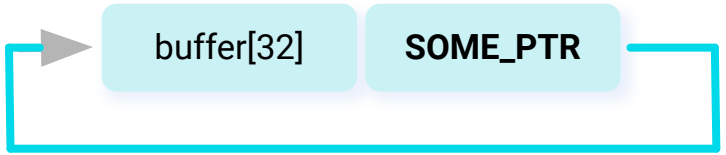
ISR(){

...

memcpy(**SOME_PTR**, CANFD_frame, 64)

...





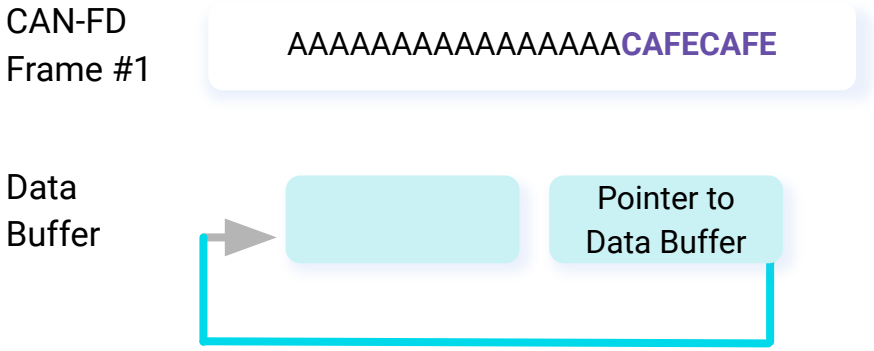
Vulnerability #1

```
ISR(){
...
  memcpy(SOME_PTR, CANFD_frame, 64)
...
```

Interrupt Service Routine (ISR) -> **Highest Privileges**

Step 1

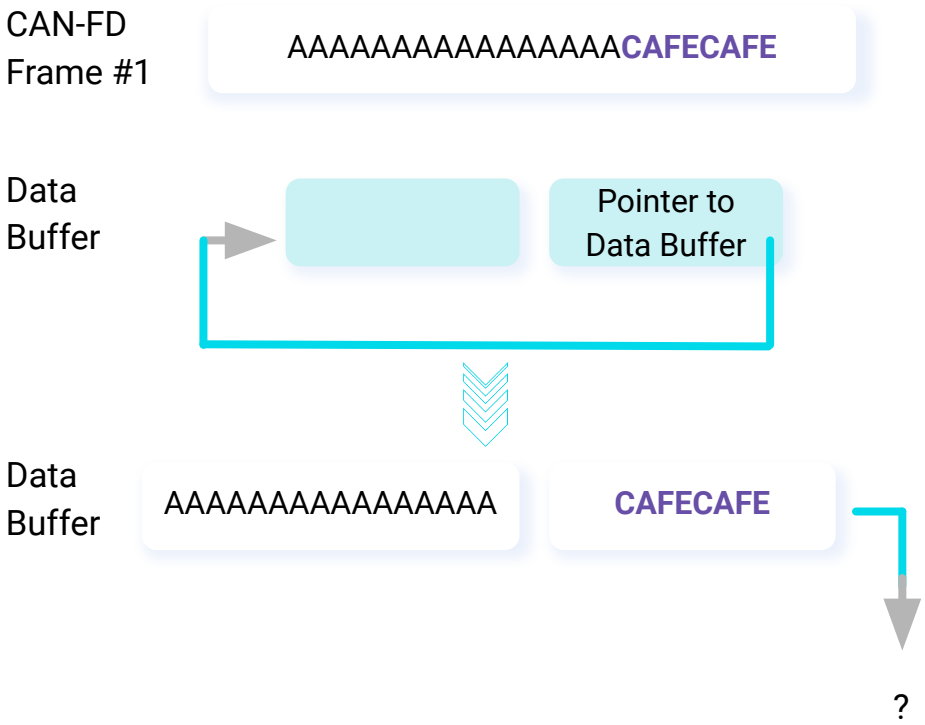
Controlling the destination pointer



PLAXIDITYX
GO EVERYWHERE

Step 1

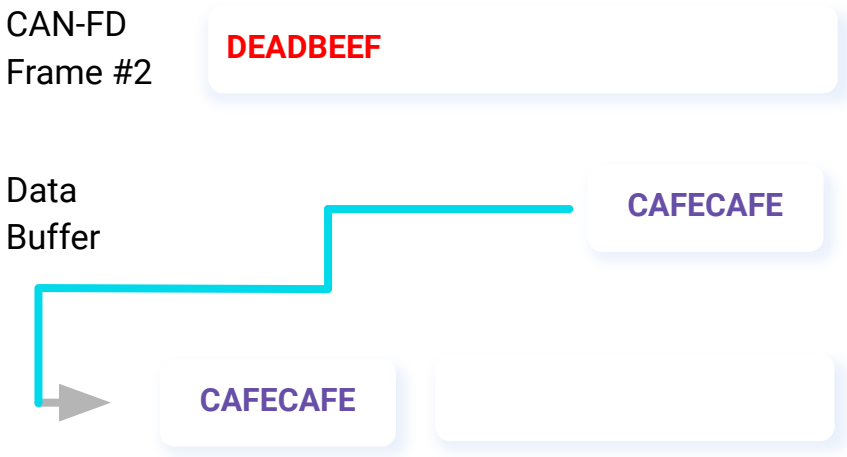
Controlling the destination pointer



PLAXIDITYX
GO EVERYWHERE

Step 2

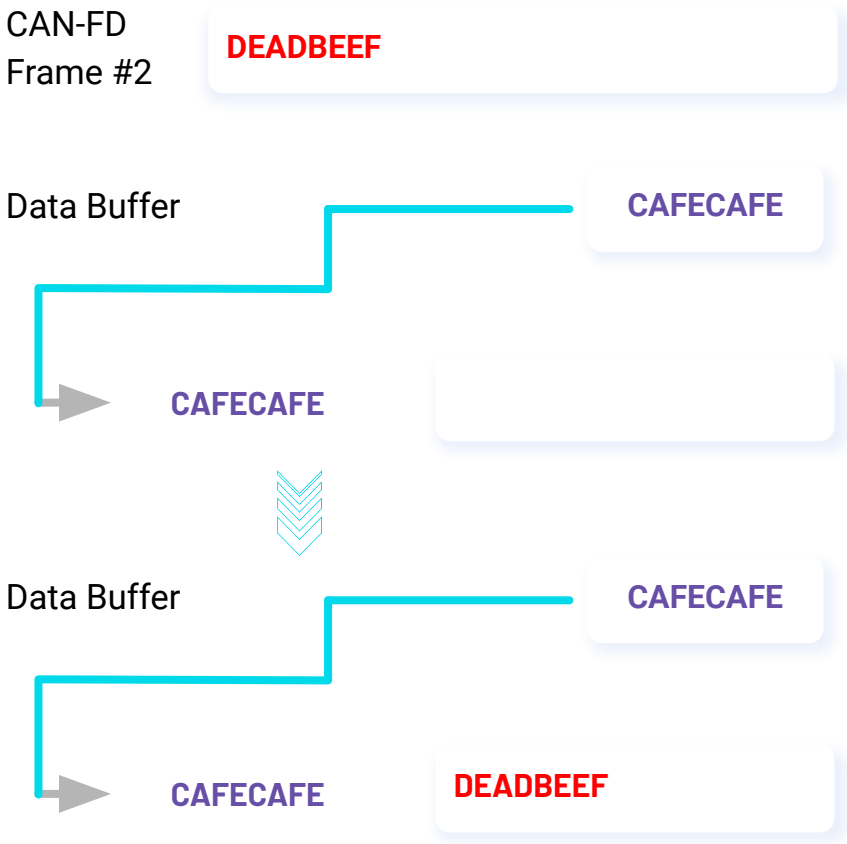
Writing



PLAXIDITYX
GO EVERYWHERE

Step 2

Writing



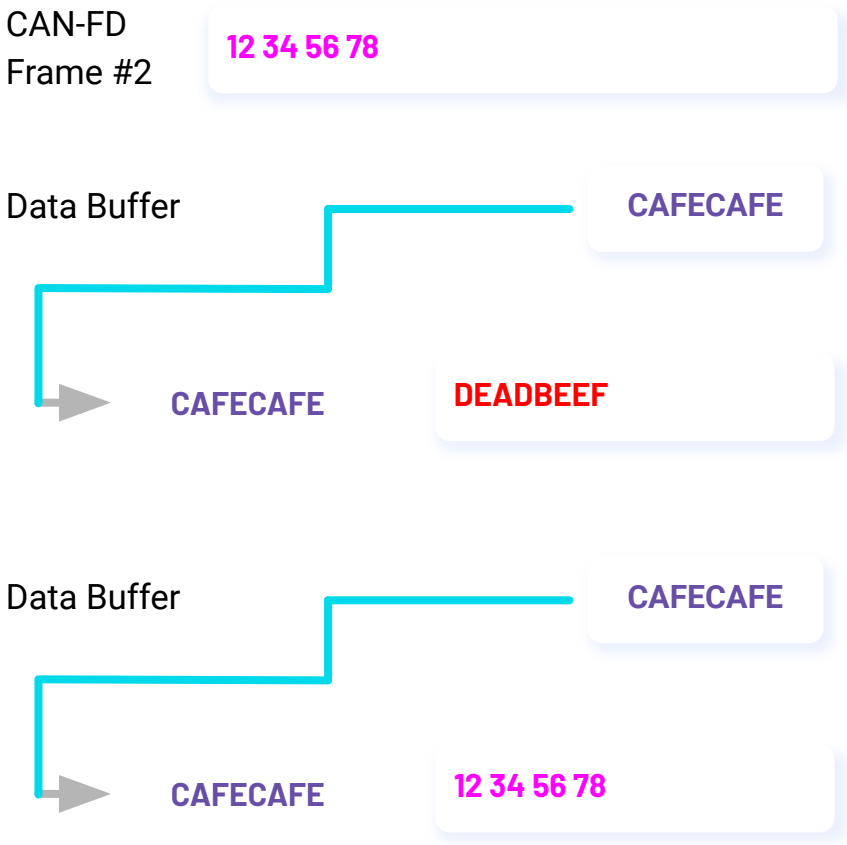
PLAXIDITYX
GO EVERYWHERE

Note

PLAXIDITYX
GO EVERYWHERE

Note

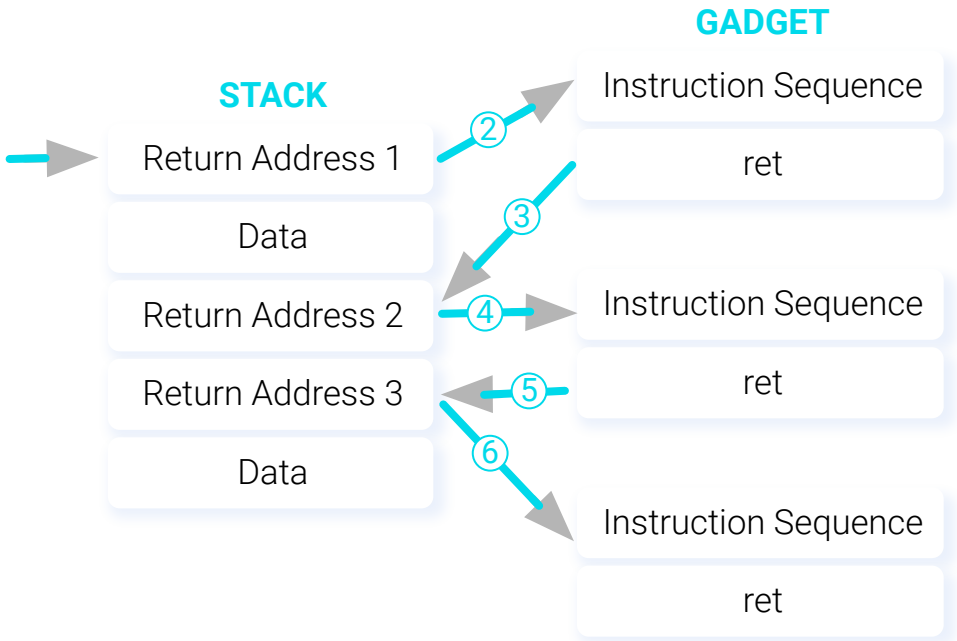
Writing once



PLAXIDITYX
GO EVERYWHERE

Vulnerability #1 - Conclusion

- Arbitrary Write via CAN-FD messages
- Writing directly to the return address on the stack
- Bypassing stack canaries
- Use Return-Oriented-Programming to get the desired code flow



Springer: ROP-Hunt: Detecting Return-Oriented Programming Attacks in Applications

Vulnerability #2:

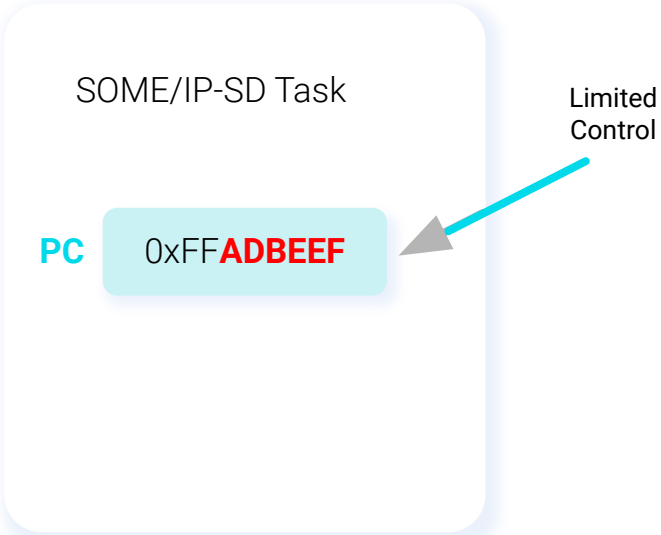
RCE over IPsec and SOME/IP-SD

PLAXIDITYX
GO EVERYWHERE

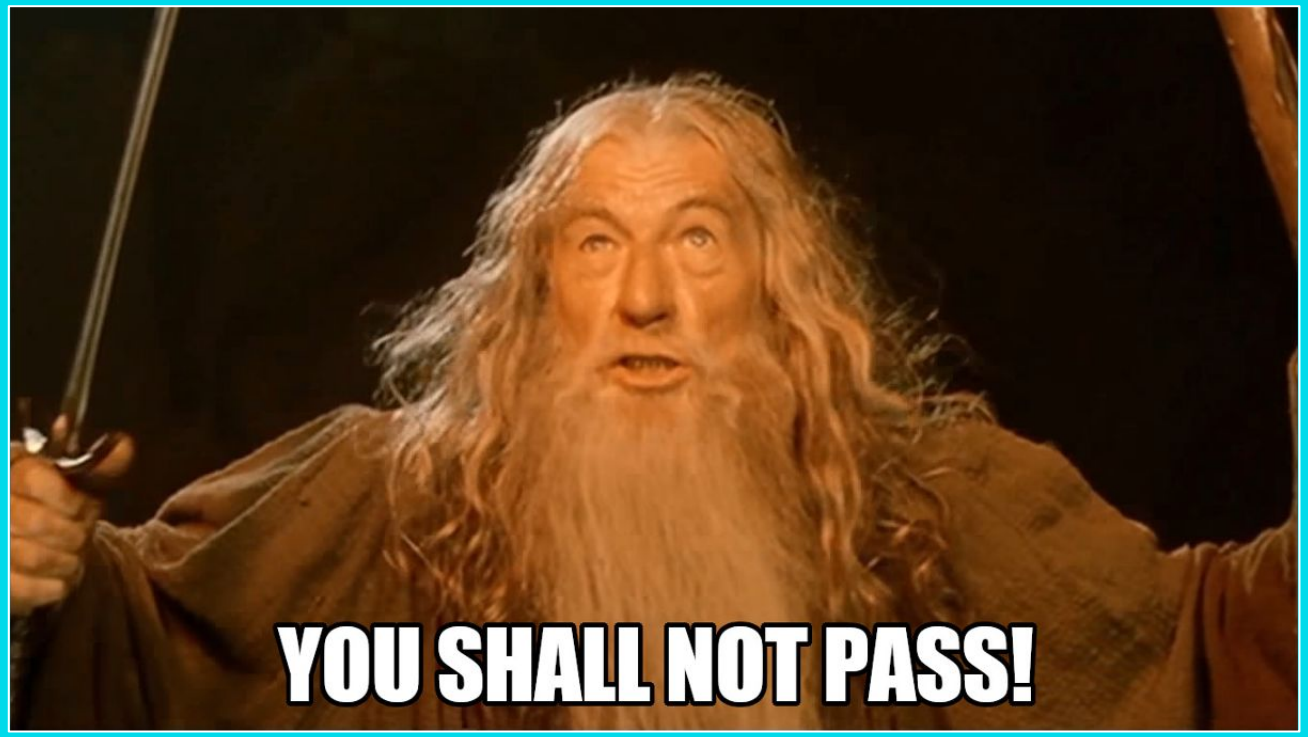
Fuzzing SOME/IP-SD

- We used our interface fuzzing tool to find issues in the SOME/IP-SD protocol handling
- The fuzzer successfully crashed the system
- A stack overflow caused the program counter to point to an invalid memory area
- No stack canaries were added as a countermeasure in the vulnerable function

Vulnerability #2 - The SD task Problem



PLAXIDITYX
GO EVERYWHERE

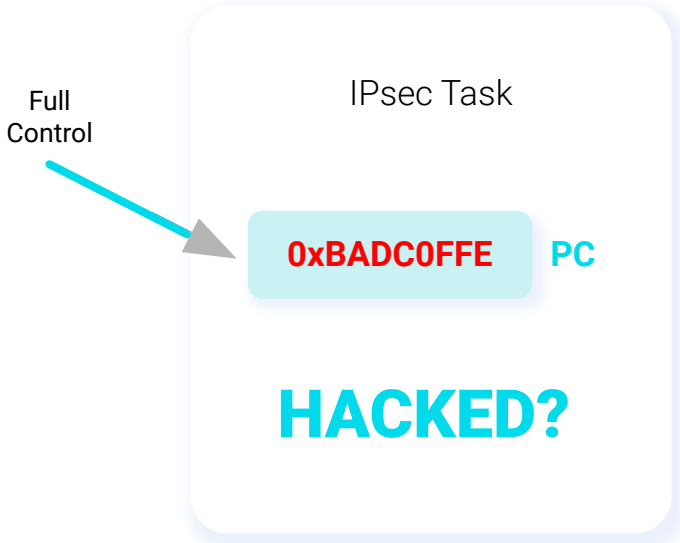


PLAXIDITYX
GO EVERYWHERE

Fuzzing IPsec

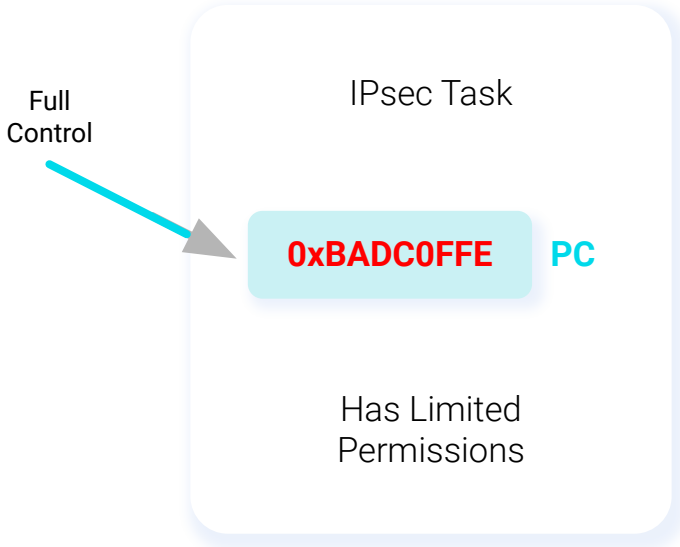
- IPsec was also part of the TCP/IP communication stack
- We used our interface fuzzing tool again
- Once more, we found a stack overflow that caused the program counter to point to an invalid memory area
- This happens as part of the authentication process (**No need to be authenticated!**)
- No stack canaries in the vulnerable function, again!

Vulnerability #2 - Full Control



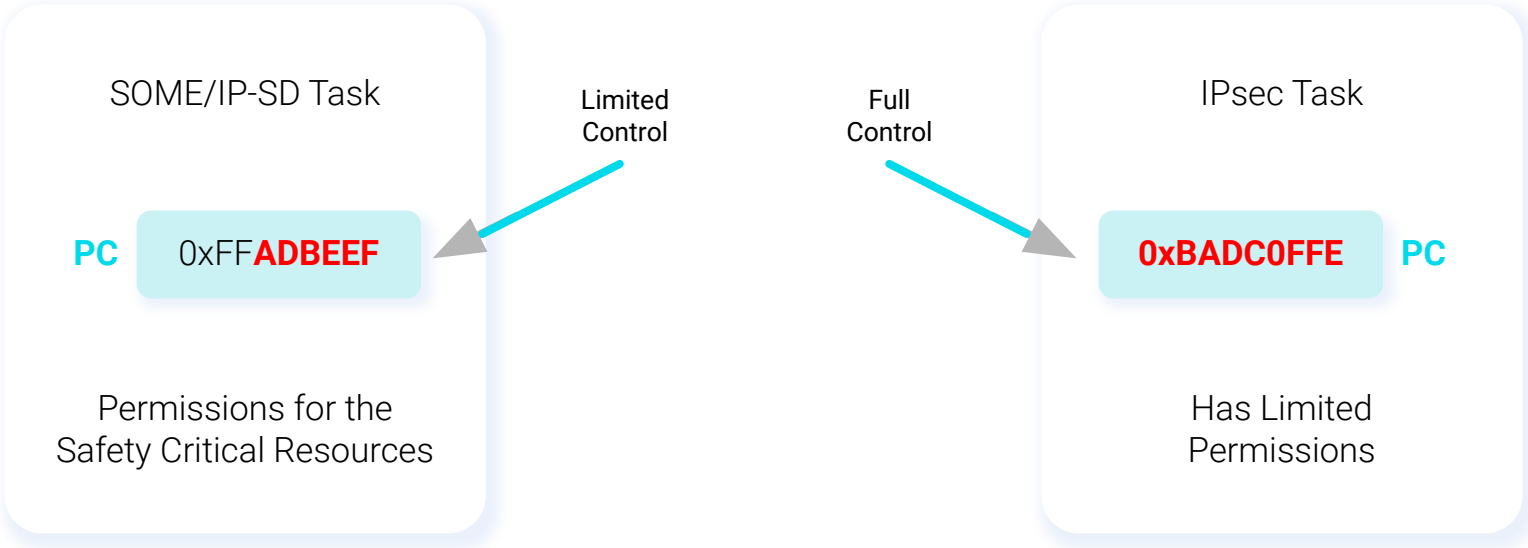
PLAXIDITYX
GO EVERYWHERE

Vulnerability #2 - The IPsec Task Problem

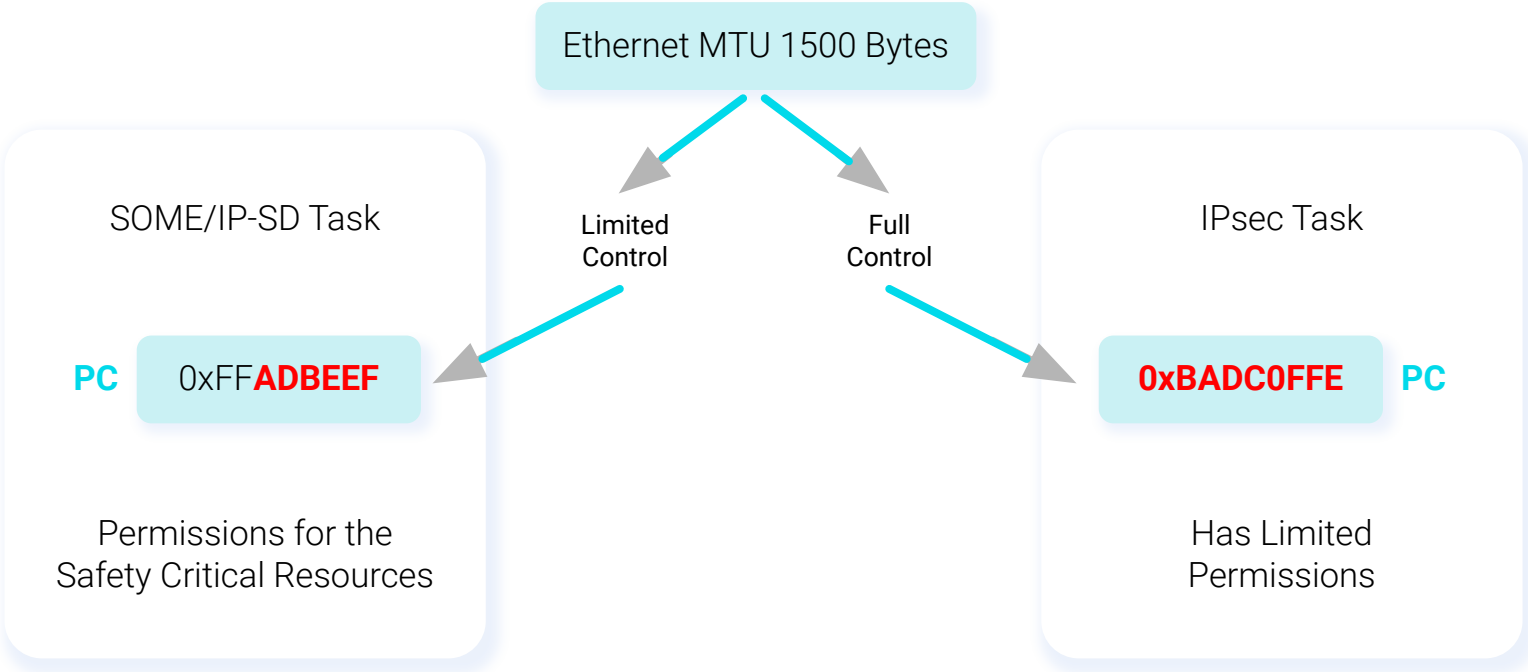


PLAXIDITYX
GO EVERYWHERE

Vulnerability #2 - The Problem

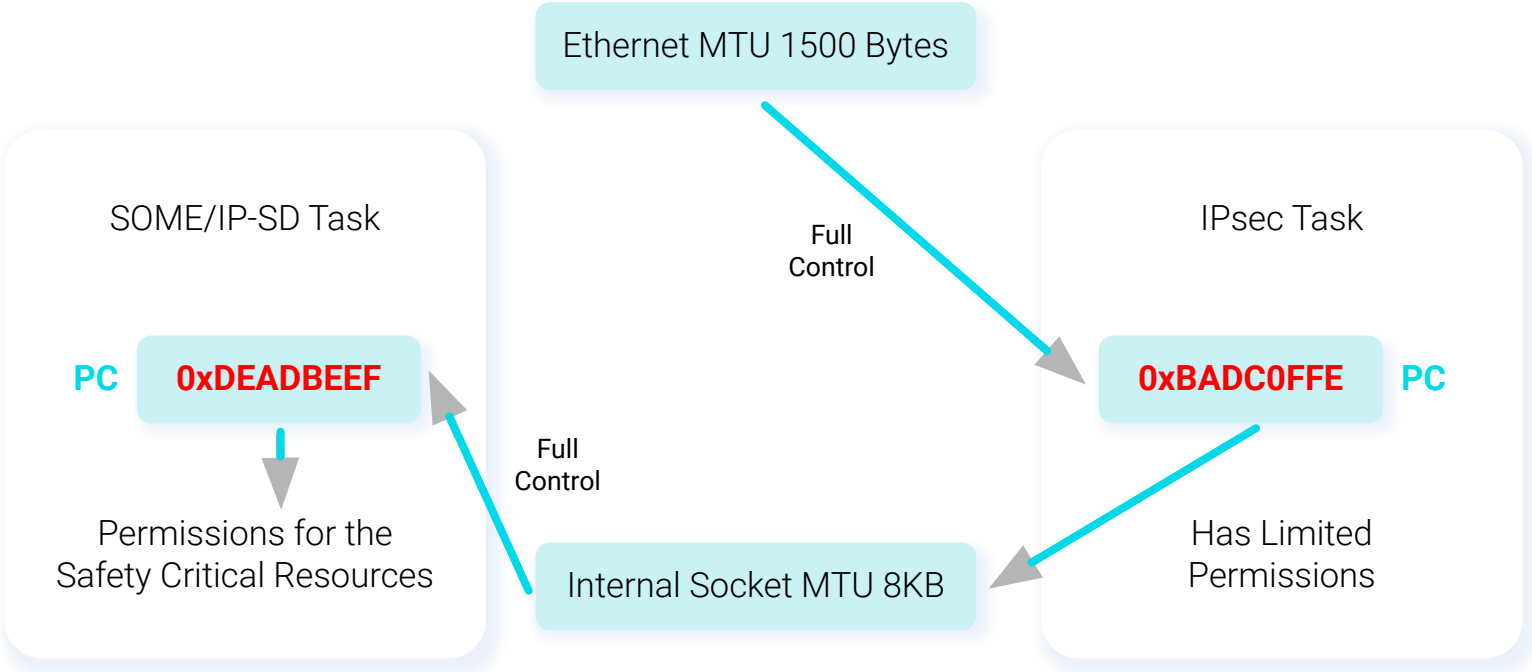


Vulnerability #2 - The Problem



PLAXIDITYX
GO EVERYWHERE

Vulnerability #2 - Chaining Vulnerabilities



PLAXIDITYX
GO EVERYWHERE

AND THE UNION OF THE TWO TASKS

PLAXIDITYX
GO EVERYWHERE

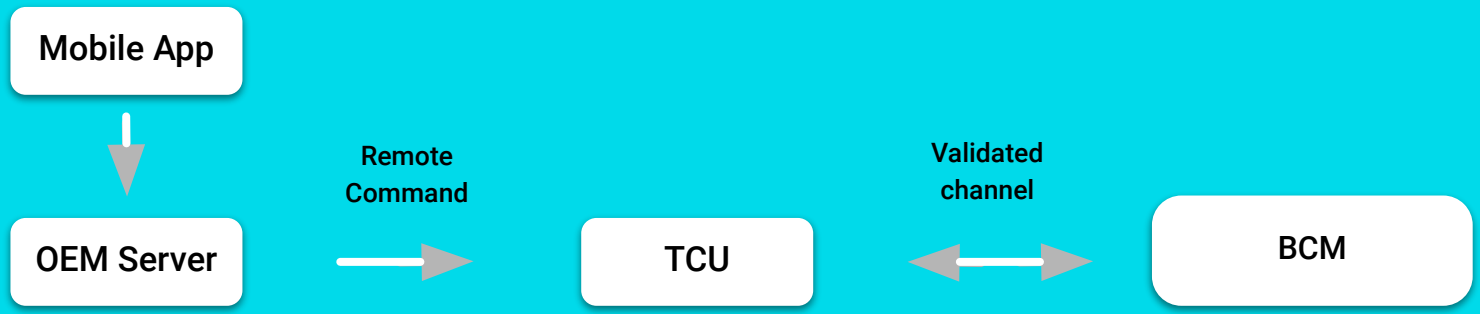
Vulnerability #3:

Shaky Cryptography

PLAXIDITYX
GO EVERYWHERE

Vulnerability #3 - Authentication of Remote Commands

- TCU can issue ‘instructions’ to the BCM based on received ‘remote commands’
- BCM validates instructions through a unique pre-shared key and AES in order to prove they came from a valid TCU source.



PLAXIDITYX
GO EVERYWHERE

Typical Cryptography Related Gaps

Non Random -

“Random” Values

Derivable Keys

From Communication

Get / Set Key

Functions

Hard Coded

Master Key

Easily

Disabled

Brute - Forceable

Keys

PLAXIDITYX
GO EVERYWHERE

What Have We Found In This Setup

Non Random -

“Random” Values

Derivable Keys

From Communication

Get / Set Key

Functions

Hard Coded

Master Key

Easily

Disabled

Brute - Forceable

Keys

PLAXIDITYX
GO EVERYWHERE

What Have We Found In This Setup

Non Random -

“Random” Values

Derivable Keys

From Communication

Get / Set Key

Functions

Hard Coded

Master Key

Easily

Disabled

Brute - Forceable

Keys

PLAXIDITYX
GO EVERYWHERE

What Have We Found In This Setup

Non Random -

"Random" Values

Derivable Keys

From Communication

Get / Set Key

Functions

Hard Coded

Master Key

Easily

Disabled

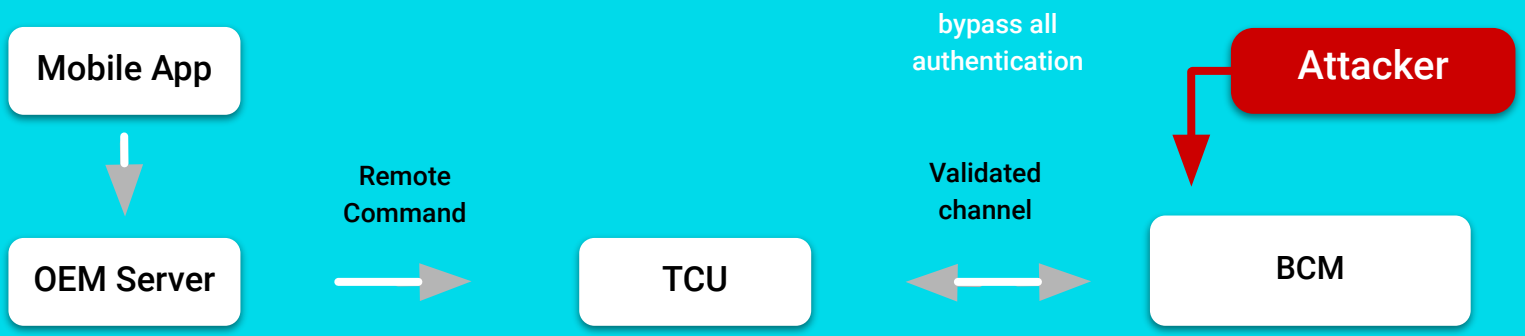
Brute - Forceable

Keys

PLAXIDITYX
GO EVERYWHERE

Vulnerability #3 - Authentication of Remote Commands

- Using any of the above vulnerabilities we were able to bypass the validation and impersonate the TCU, without any prior knowledge of any key material



PLAXIDITYX
GO EVERYWHERE

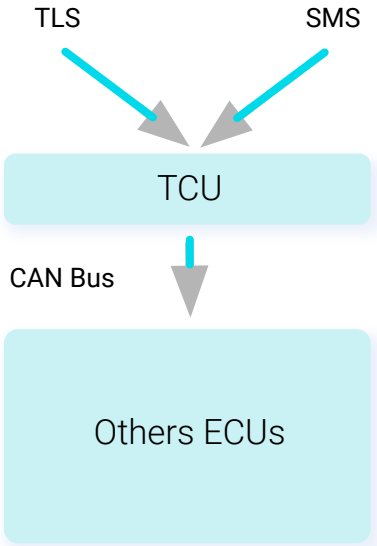
Vulnerability #4:

Remote and Persistent Vulnerability via Cellular Connection

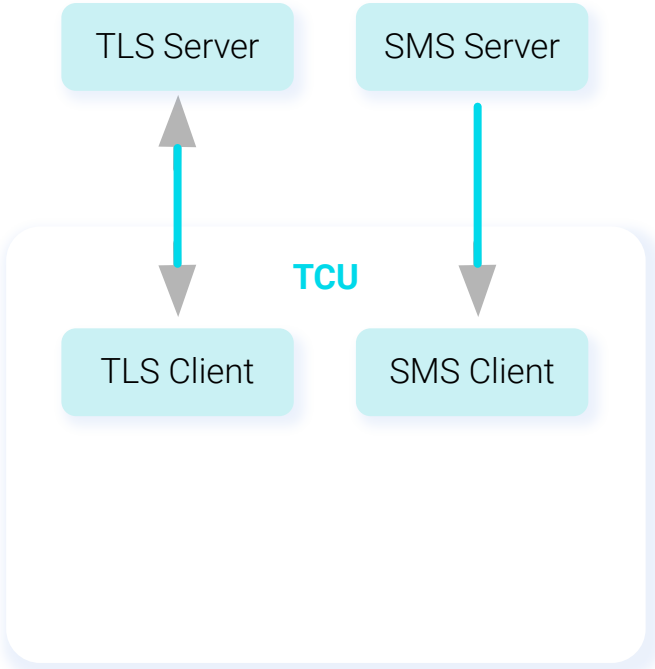
PLAXIDITYX
GO EVERYWHERE

Vulnerability #4 - Setup

- TCU used in a heavy duty vehicle with a cellular modem
- Communication with the backend via secured TLS
- Connected directly to safety critical CAN bus
- Can get binary SMS to trigger communication with the backend

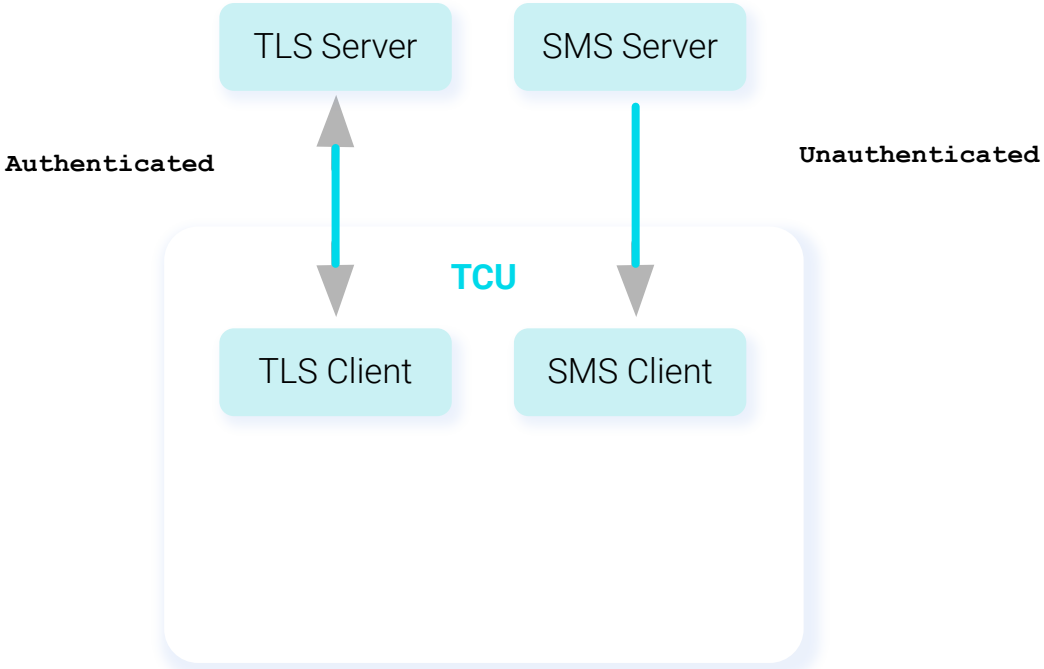


Vulnerability #4 - Setup



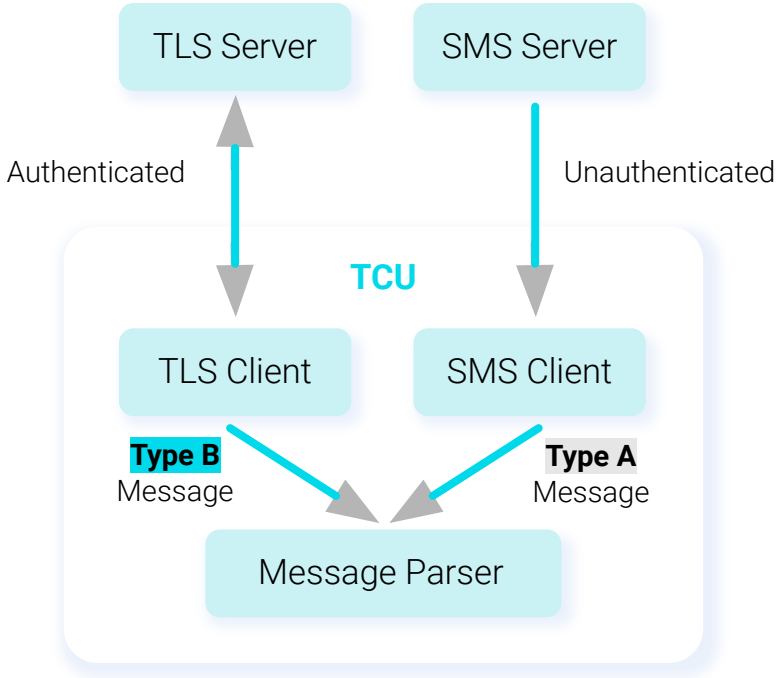
PLAXIDITYX
GO EVERYWHERE

Vulnerability #4 - Connections

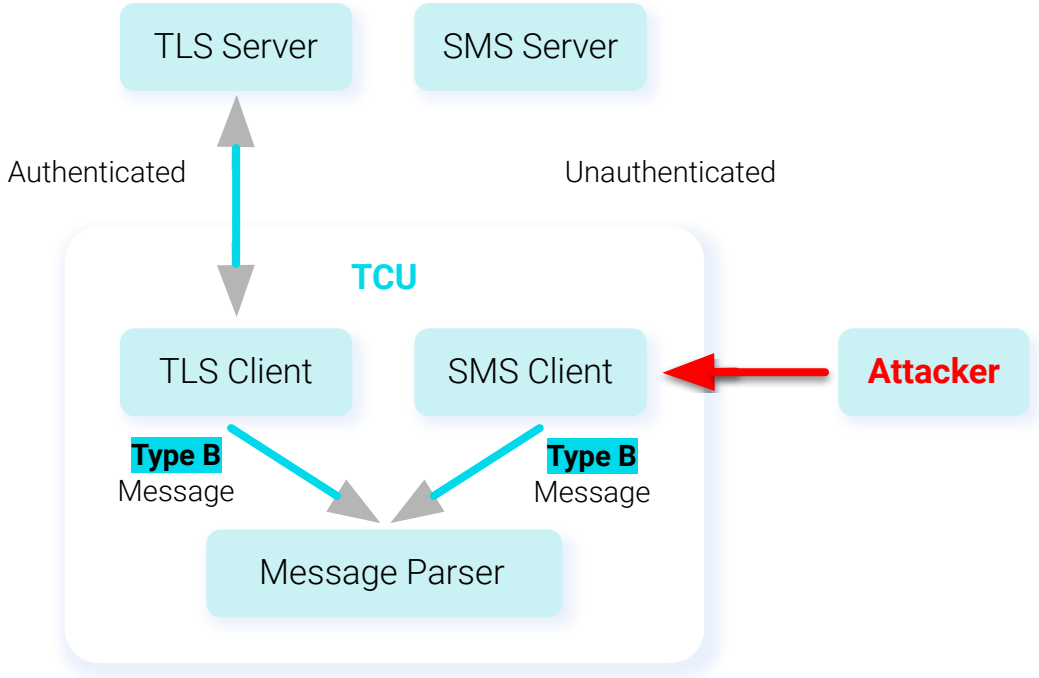


PLAXIDITYX
GO EVERYWHERE

Vulnerability #4 - Messages Types



Vulnerability #4 - Messages Types



Vulnerability #4

PLAXIDITYX
GO EVERYWHERE

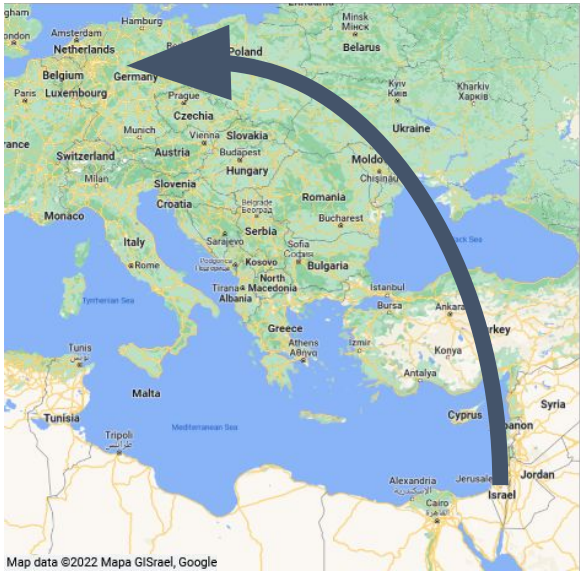
Vulnerability #4

PLAXIDITYX
GO EVERYWHERE

Vulnerability #4 - The Next Step

- Attacker can use a **backend command to replace** a file within the system
- Some files on the system are **tagged with a digital-signature**
- But some of the partitions in the file system **did not have any validation**
- A subset of those were not flagged as **“noexec”**
- Therefore, an attacker **can inject an arbitrary executable** to any of these partitions

Vulnerability #4 - End Game



- This vulnerability enabled us to run a full demo of injecting CAN messages to impact the vehicle via an unauthenticated binary SMS message. From **Tel Aviv**, **Israel** to **Europe**.

Conclusions

PLAXIDITYX
GO EVERYWHERE

Conclusions

- High severity **Zero-Day vulnerabilities are still common in multiple types** of ECUs (some safety-critical)
- More **vulnerabilities are caught Pre-SOP** nowadays
- These vulnerabilities are the **result of faulty implementations of protocols** and interfaces
- Security of modern, **complex vehicle systems requires a holistic approach** with layered security controls and vast security auditing

Conclusions

- Augmented security awareness during the development process
- Fuzzing Tools
- Don’t underestimate a limited security issue
- Hardening!
- Penetration testing, by security researchers

PLAXIDITYX
GO EVERYWHERE

Thank you

PLAXIDITYX
GO EVERYWHERE