RÉPUBLIQUE
FRANÇAISE
Liberté
Égalité
Fraternité

REVISITING BLUETOOTH VULNERABILITIES IN AUTOMOTIVE INFOTAINMENT SYSTEMS

A Remote Code Execution Case Study

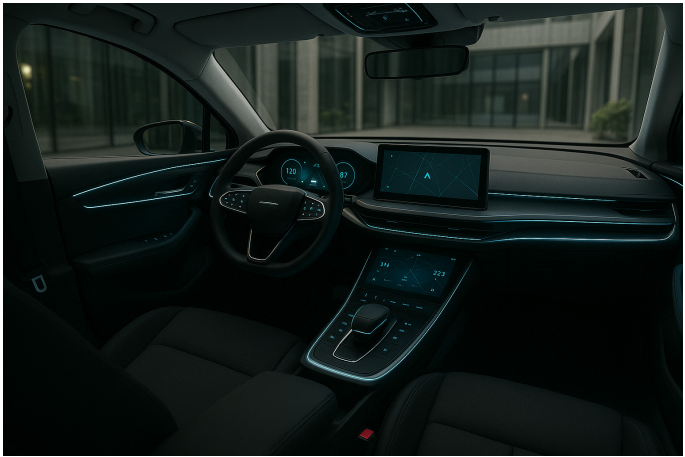
Philippe Trébuchet and **Guillaume Bouffard**
National Cybersecurity Agency of France (ANSSI)
Hardware and Software Architectures Lab

https://cyber.gouv.fr

Increasingly Connected Vehicles

(image generated by ChatGPT-4o)

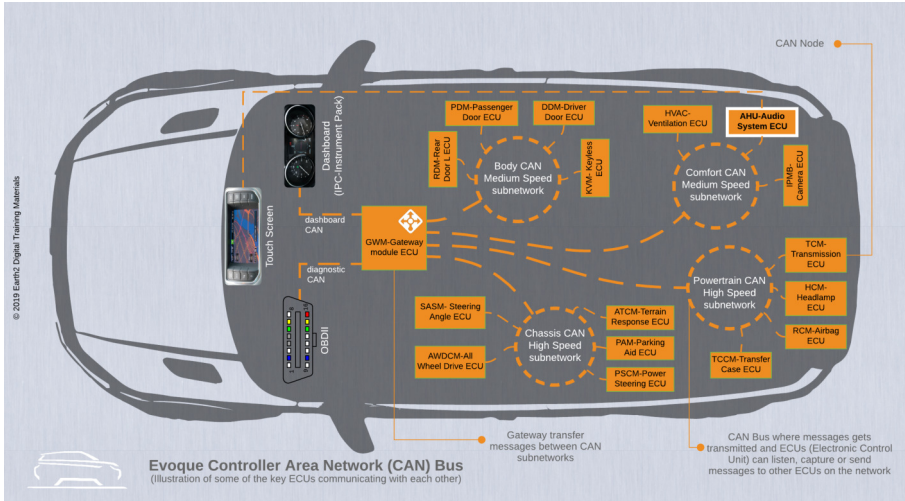
Increasingly Connected Vehicles



An ever-expanding attack surface!

(image generated by ChatGPT-4o)

Example of an IT Architecture for Connected Vehicles



(Source: `https://www.earth2.digital/blog/what-is-vehicle-can-bus-ecu-evoque-adam-ali.html`)

Attacker Model

Local attacker:

- Present inside the vehicle
- Physical access to internal vehicle buses

Attacker Model

Local attacker:

- Present inside the vehicle
- Physical access to internal vehicle buses

Remote attacker:

- Ability to interact with all external sensors
- Ability to interact with all wireless communication devices

Attacker Model

Local attacker:

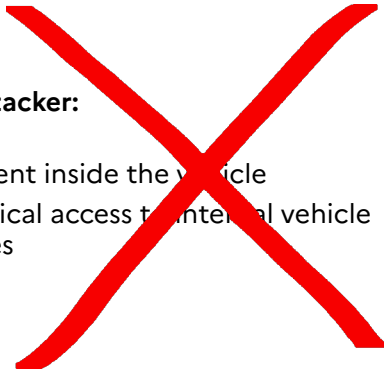
- Present inside the vehicle
- Physical access to internal vehicle buses

Remote attacker:

- Ability to interact with all external sensors
- Ability to interact with all wireless communication devices

Study conducted on a **2020 production vehicle** (representative of those used by public administration) featuring Bluetooth, Wi-Fi, and all premium-class options.

Attacker Model



Local attacker:

- Present inside the vehicle
- Physical access to internal vehicle buses

Remote attacker:

- Ability to interact with all external sensors
- Ability to interact with all wireless communication devices

Study conducted on a **2020 production vehicle** (representative of those used by public administration) featuring Bluetooth, Wi-Fi, and all premium-class options.

In-Vehicle Infotainment Software Architecture

Interfaces

- Wi-Fi
- GSM
- Bluetooth

In-Vehicle Infotainment Software Architecture

Interfaces

- Wi-Fi
- GSM
- Bluetooth

Security Functions

- **ASLR (*Address Space Layout Randomization*)**
  - randomizes memory locations (code, stack, libraries)
  - makes ROP/JOP attacks more difficult
  - strongly depends on available entropy
- **W⊕X (*Write XOR Execute* – DEP)**
  - prevents execution of data in memory (e.g., stack, heap)
  - blocks execution of injected shellcodes

In-Vehicle Infotainment Software Architecture

Interfaces

- Wi-Fi: disabled by default.
- GSM: uncontrolled traffic.
- Bluetooth

Security Functions

- **ASLR (*Address Space Layout Randomization*)**
  - randomizes memory locations (code, stack, libraries)
  - makes ROP/JOP attacks more difficult
  - strongly depends on available entropy
- **W⊕X (*Write XOR Execute* – DEP)**
  - prevents execution of data in memory (e.g., stack, heap)
  - blocks execution of injected shellcodes

In-Vehicle Infotainment Software Architecture

Interfaces

- Wi-Fi: disabled by default.
- GSM: uncontrolled traffic.
- **Bluetooth**

Security Functions

- **ASLR (*Address Space Layout Randomization*)**
  - randomizes memory locations (code, stack, libraries)
  - makes ROP/JOP attacks more difficult
  - strongly depends on available entropy
- **W⊕X (*Write XOR Execute* – DEP)**
  - prevents execution of data in memory (e.g., stack, heap)
  - blocks execution of injected shellcodes

In-Vehicle Infotainment Software Architecture

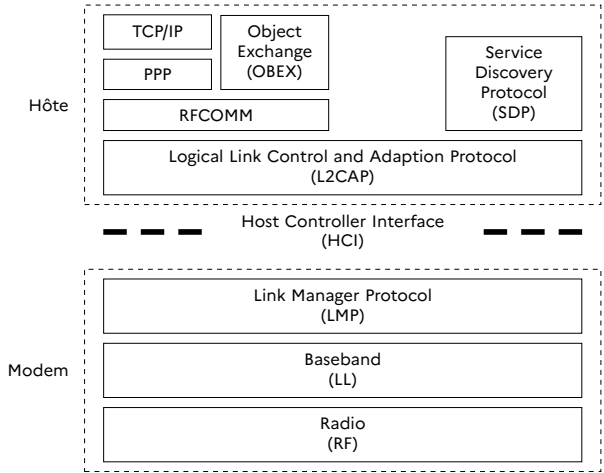
Interfaces

- Wi-Fi: disabled by default.
- GSM: uncontrolled traffic.
- **Bluetooth** moved outside kernel: `BluegoService` (Java service)
  - JNI and `libbluego.so`.
  - single public CVE for this library (`CVE-2018-20378`).

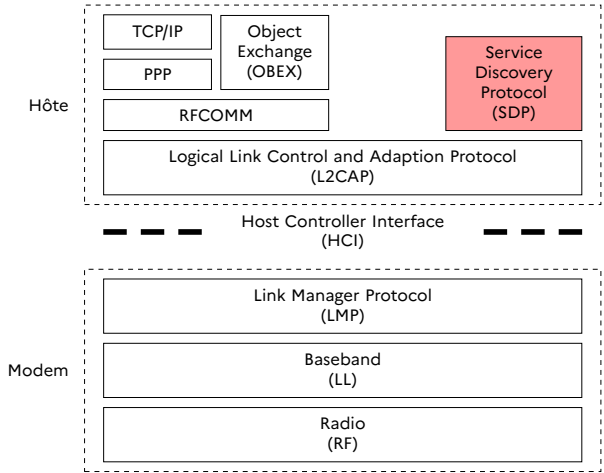
Security Functions

- **ASLR (*Address Space Layout Randomization*)**
  - randomizes memory locations (code, stack, libraries)
  - makes ROP/JOP attacks more difficult
  - strongly depends on available entropy
- **W⊕X (*Write XOR Execute* – DEP)**
  - prevents execution of data in memory (e.g., stack, heap)
  - blocks execution of injected shellcodes

The Bluetooth Stack



[a part of] # The Bluetooth Stack



Protocol usable before pairing

Libbluego: CVE-2018-20378

`libbluego` version 2.X (`2.6.0-rc1`) is found in the studied ECU.

- CVE published on 03/29/2019 under reference `CVE-2018-20378`
- Invalid MTU in the L2CAP layer.
- Targets: OpenSynergy Blue SDK[a] versions **3.2** to 6.0.
- Closed-source library (explaining the relatively low CVSS score)

[a] **See: `https://www.opensynergy.com/blue-sdk/`**

"The L2CAP signaling channel implementation and SDP server implementation in OpenSynergy Blue SDK 3.2 through 6.0 allow remote, unauthenticated attackers to execute arbitrary code or cause a denial of service via malicious L2CAP configuration requests, in conjunction with crafted SDP communication over maliciously configured L2CAP channels. The attacker must have connectivity over the Bluetooth physical layer, and must be able to send raw L2CAP frames."

CVSS scores for CVE-2018-20378

Base Score	Base Severity	CVSS Vector	Exploitability Score	Impact Score	Source
5.4	MEDIUM	AV:A/AC:M/Au:N/C:P/I:P/A:P	5.5	6.4	nvd@nist.gov
7.5	HIGH	CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H	1.6	5.9	nvd@nist.gov

Libbluego: Overview of CVE-2018-20378

Analysis of CVE-2018-20378

- The authors *claim* to have developed a proof of concept using JOP *(Jump Oriented Programming)*.
- No details of the exploitation are public, even in related publications!

Libbluego: Overview of CVE-2018-20378

Analysis of CVE-2018-20378

- The authors *claim* to have developed a proof of concept using JOP *(Jump Oriented Programming)*.
- No details of the exploitation are public, even in related publications!

- **Can this CVE be transposed to the target vehicle?**

Libbluego: Overview of CVE-2018-20378

Analysis of CVE-2018-20378

- The authors *claim* to have developed a proof of concept using JOP *(Jump Oriented Programming)*.
- No details of the exploitation are public, even in related publications!

- **Can this CVE be transposed to the target vehicle? Yes!**
  - … but it's far from straightforward
  - … **the CVE details do not apply to the target!**
    - (since the target embeds a version older than the oldest vulnerable version)

Analysis of CVE-2018-20378

- The authors *claim* to have developed a proof of concept using JOP *(Jump Oriented Programming)*.
- No details of the exploitation are public, even in related publications!

- **Can this CVE be transposed to the target vehicle? Yes!**
  - … but it's far from straightforward
  - … **the CVE details do not apply to the target!**
    - (since the target embeds a version older than the oldest vulnerable version)
  - **the actual exploitation turns out to be completely different** from the one **suggested** by the authors

General view of the retained attack scenario

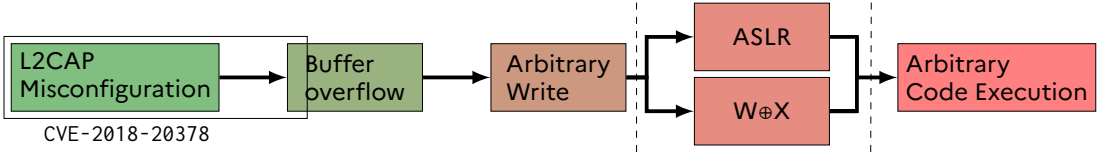
L2CAP Misconfiguration ⇢ Arbitrary Code Execution

General view of the retained attack scenario

Step 1
Buffer overflow exploitation

Step 2
Bypassing security mechanisms

Step 3
Arbitrary code execution

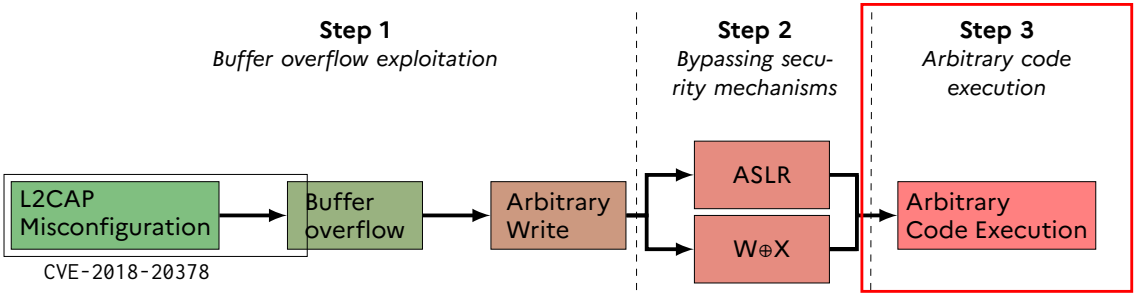


General view of the retained attack scenario

Step 1
Buffer overflow exploitation

Step 2
Bypassing security mechanisms

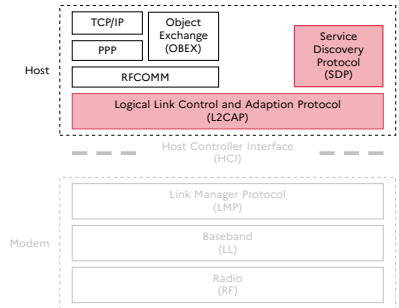
Step 3
Arbitrary code execution



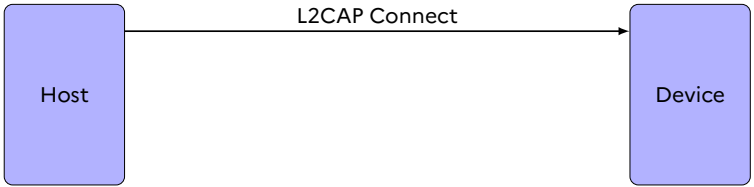
Unsecured Channel Management *(SDP only)*

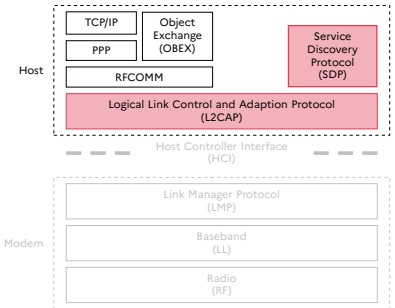
Host

Device

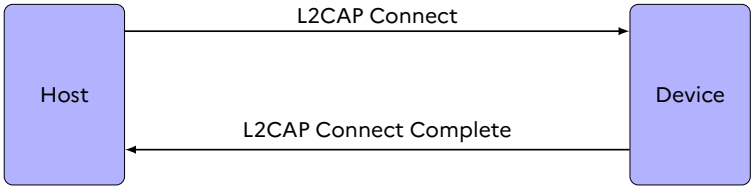


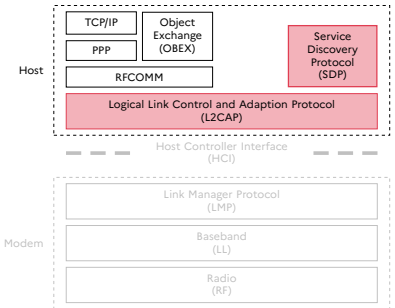
Unsecured Channel Management *(SDP only)*



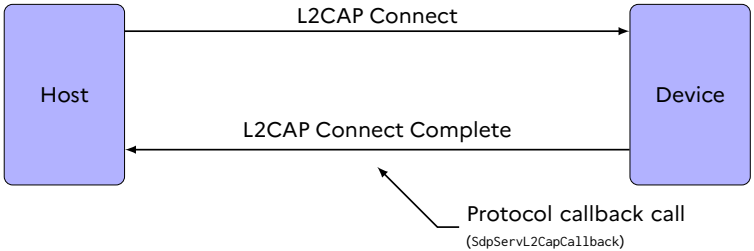


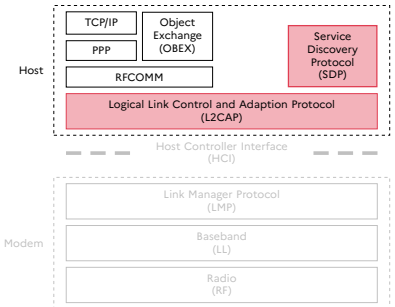
Unsecured Channel Management *(SDP only)*





Unsecured Channel Management *(SDP only)*

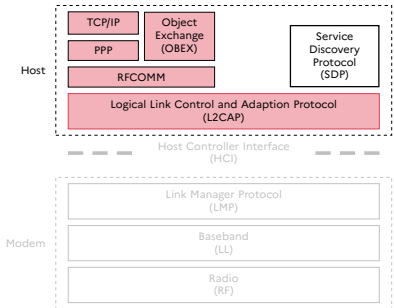




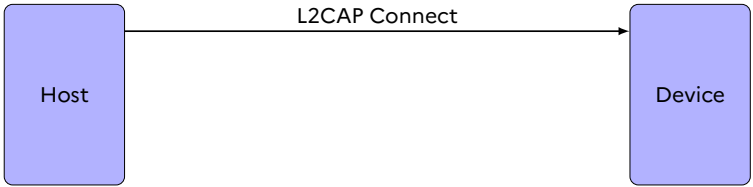
Secured Channel Management

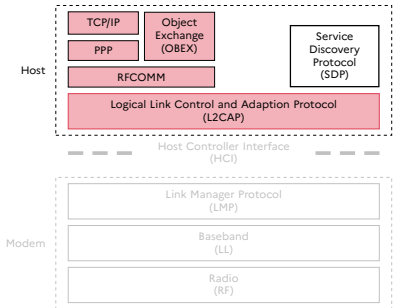
Host

Device

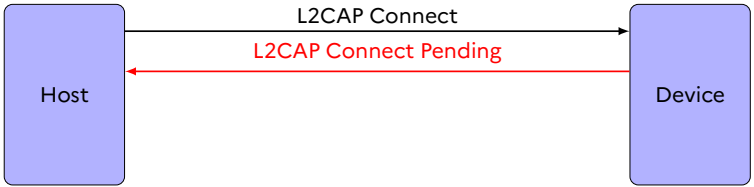


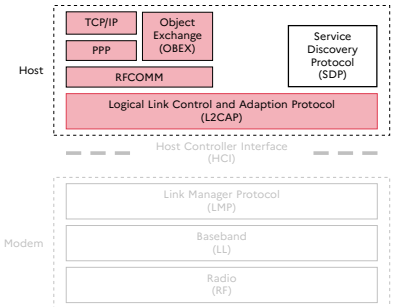
Secured Channel Management



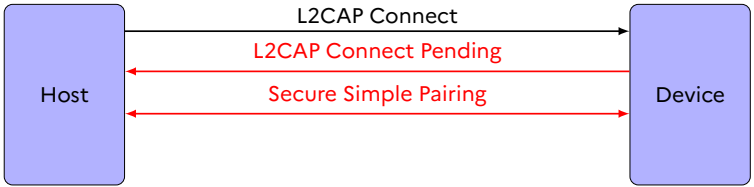


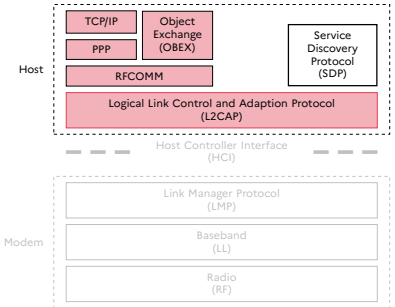
Secured Channel Management



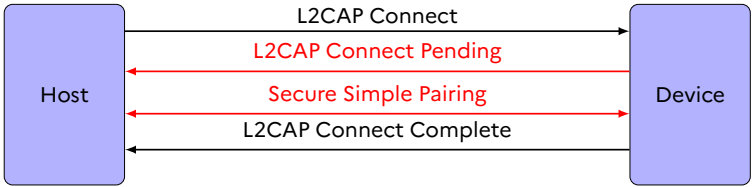


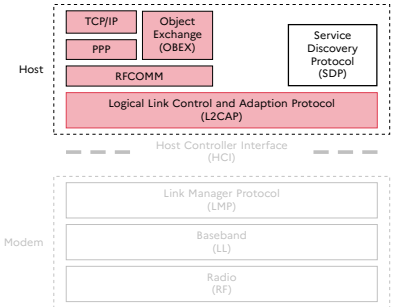
Secured Channel Management



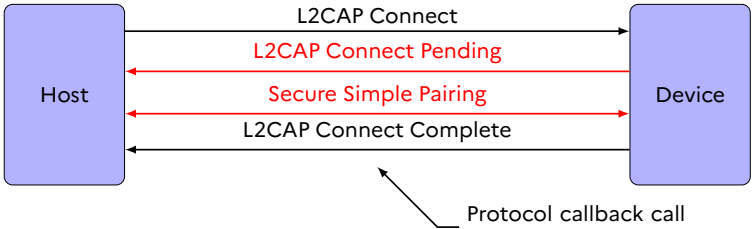


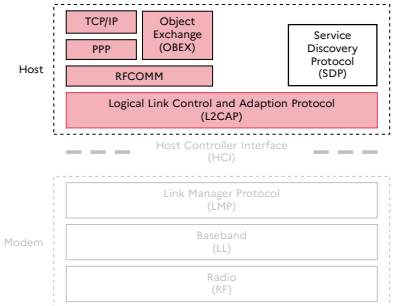
Secured Channel Management



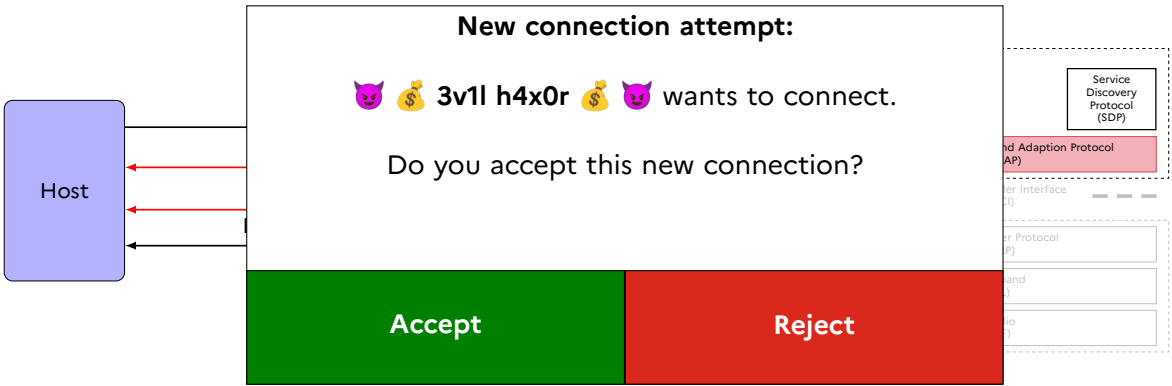


Secured Channel Management

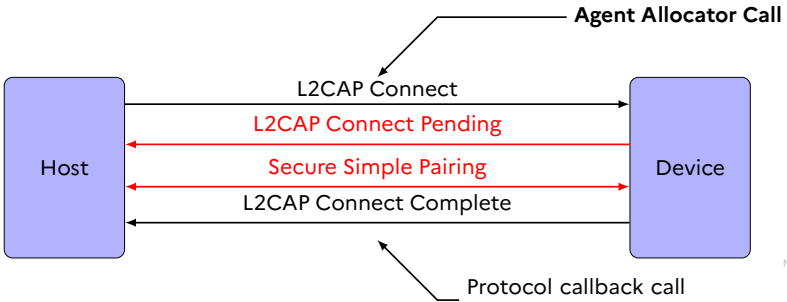


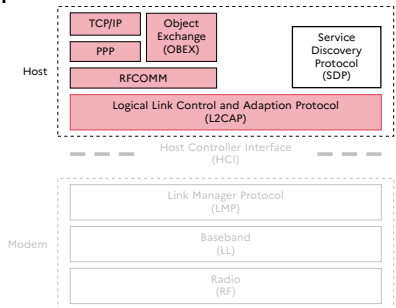


Secured Channel Management



Secured Channel Management

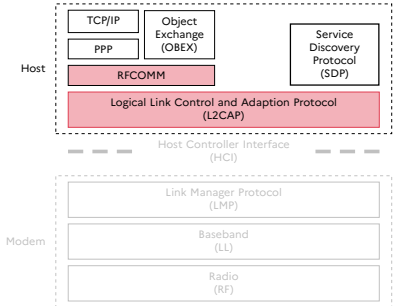




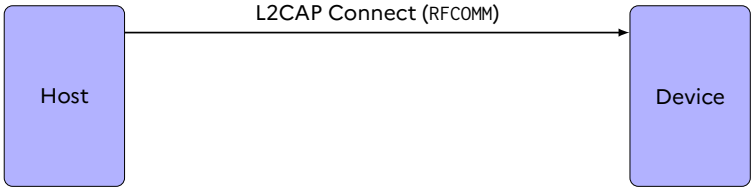
Triggering Execution

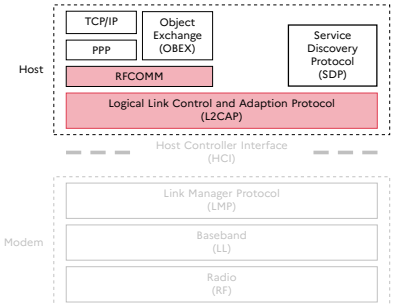
Host

Device

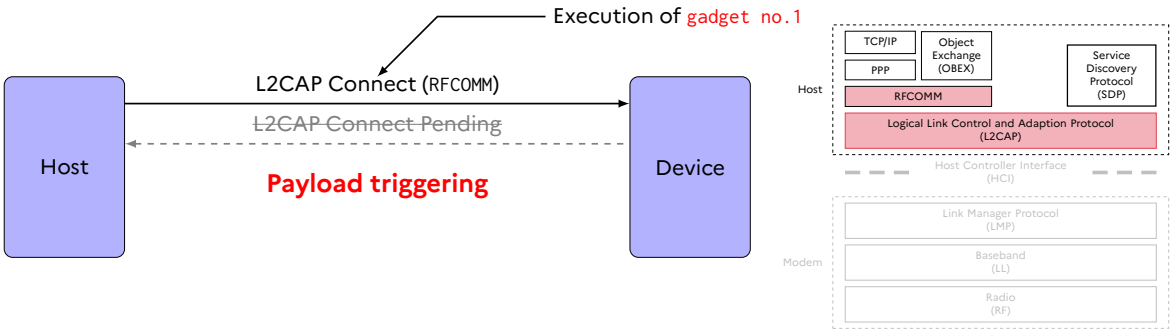


Triggering Execution





Triggering Execution



Security Impact

Vulnerability Consequences

- Arbitrary code execution on the infotainment system
  - within a process with system privileges.
- Ability to execute CAN commands from the arbitrary code
  - **remote execution of CAN commands**

Security Impact

Vulnerability Consequences

- Arbitrary code execution on the infotainment system
  - within a process with system privileges.
- Ability to execute CAN commands from the arbitrary code
  - **remote execution of CAN commands**

Impact on other vehicles of different models and brands not updated

Security Impact

Vulnerability Consequences

- Arbitrary code execution on the infotainment system
  - within a process with system privileges.
- Ability to execute CAN commands from the arbitrary code
  - **remote execution of CAN commands**

Impact on other vehicles of different models and brands not updated

Responsible Disclosure

- Work conducted between September 2023 and April 2024.
- The manufacturer was notified in **September 2024**.
- Positive reception and very rapid response from them.
- The manufacturer states that critical CAN commands are filtered.

This vulnerability was patched by the target vehicle manufacturer.

Conclusion

- Analysis of an embedded Bluetooth stack:
  - vehicle representative of 2020 models and **still in circulation**.
- Presence of a vulnerability referenced by `CVE-2018-20378`:
  - initially considered not exploitable in the embedded version of the library.
  - public exploitation details are not directly applicable.
- Analysis conducted in **black box** conditions.
- Proposal of an original exploitation scenario:
  - **remote** and **no user interaction** attack.

Conclusion

- Analysis of an embedded Bluetooth stack:
  - vehicle representative of 2020 models and **still in circulation**.
- Presence of a vulnerability referenced by `CVE-2018-20378`:
  - initially considered not exploitable in the embedded version of the library.
  - public exploitation details are not directly applicable.
- Analysis conducted in **black box** conditions.
- Proposal of an original exploitation scenario:
  - **remote** and **no user interaction** attack.
- Patch deployed by the manufacturer!

Conclusion

- Analysis of an embedded Bluetooth stack:
  - vehicle representative of 2020 models and **still in circulation**.
- Presence of a vulnerability referenced by `CVE-2018-20378`:
  - initially considered not exploitable in the embedded version of the library.
  - public exploitation details are not directly applicable.
- Analysis conducted in **black box** conditions.
- Proposal of an original exploitation scenario:
  - **remote** and **no user interaction** attack.
- Patch deployed by the manufacturer!

Vehicle security has a direct impact on their operational safety.

Conclusion

- Analysis of an embedded Bluetooth stack:
  - vehicle representative of 2020 models and **still in circulation**.
- Presence of a vulnerability referenced by `CVE-2018-20378`:
  - initially considered not exploitable in the embedded version of the library.
  - public exploitation details are not directly applicable.
- Analysis conducted in **black box** conditions.
- Proposal of an original exploitation scenario:
  - **remote** and **no user interaction** attack.
- Patch deployed by the manufacturer!

Vehicle security has a direct impact on their operational safety.

The attack surface continues to expand:

- Electric vehicles: *Vehicle-to-Charger* (V2C) interfaces.
- *Vehicle-to-Vehicle* (V2V) communication, still rarely deployed but already critical.

Conclusion

- Analysis of an embedded Bluetooth stack:
  - vehicle representative of 2020 models and **still in circulation**.
- Presence of a vulnerability referenced by `CVE-2018-20378`:
  - initially considered not exploitable in the embedded version of the library.
  - public exploitation details are not directly applicable.
- Analysis conducted in **black-box conditions**.
- [illegible]
- [illegible]

[Much] more detail in the associated ACSW 2026 paper!

Vehicle security has a direct impact on their operational safety.

The attack surface continues to expand:

- Electric vehicles: *Vehicle-to-Charger* (V2C) interfaces.
- *Vehicle-to-Vehicle* (V2V) communication, still rarely deployed but already critical.

Questions?

Philippe Trébuchet & Guillaume Bouffard
National Cybersecurity Agency of France (ANSSI)
Hardware and Software Architectures Lab
Paris, France

1 Introduction
2 Firmware Security Analysis
3 Remote Code Execution
4 Conclusion and Future Work