FROM RESEARCH TO INDUSTRY

Towards Fault Analysis of Firmware Updaters

L. Morel[1] & **M-L Potet**[3]
D. Couroussé[1] & **L. Mounier**[3] & **L. Maingault**[2]

lionel.morel@cea.fr

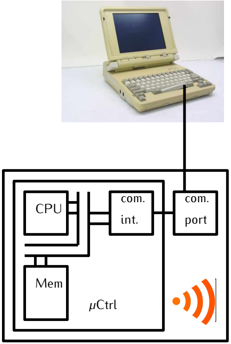
[1]Univ Grenoble Alpes, CEA, List,
[2]Univ Grenoble Alpes, CEA, Leti
[3]Verimag
F-38000 Grenoble, France

leti & list

IoT Platforms

We are interested in platforms that :

- are **small** (wearables, in-body, appliances, etc.).
- have **low computational power** : 25 to a few 100s MIPS.
- have **low memory capacity**: few Kb to few Mb.
- Let's suppose **no HW security** component.
- **No full-fledged OS**.



What's a Bootloader?

- Minimal piece of software to **start** the platform
- Usually **sets up peripherals, memory, etc.**
- ... and jumps to OS's or application's `main`

```
ResetHandler(){
  timerInit();
  memInit();
  comInit();
  encryptionInit();
  main();
}
```

What's a Firmware?

Depends ...

It can mean:

- The low-level interface with HW, ie drivers
- The OS-like code around the "functionnal" heart of the application
- The whole code (app + drivers + glue code)

For now, we will consider the **update of the whole software present on the platform**.

What's a Firmware Updater (FU)?

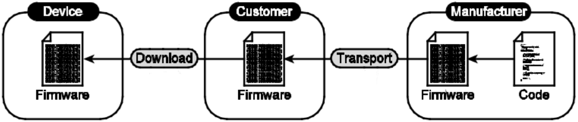
- Functionality to update the code of the platform firmware
- New functionalities, bug fixes, vulnerability fixes, etc.
- Implies code within the bootloader and ....
- .... code on a remote system (base station, gateway, etc)

- **The attack surface is quite large ... !**

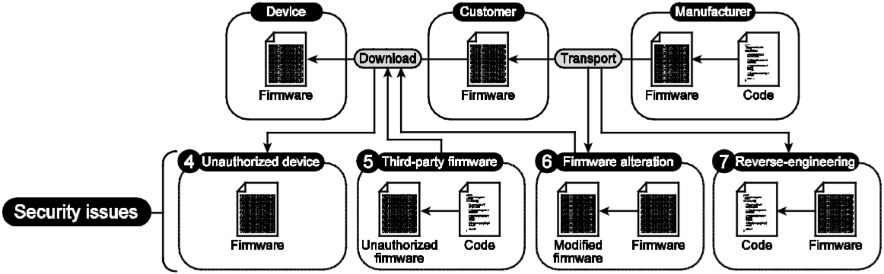
Why study FUs?

- Achille's heel of the whole system: if you control FU, you control the ~~world~~ platform.
- SoA essentially deals with "logical" security, not with physical attacks.
- Good intermediate level of complexity.

Typical Firmware Upgrading Flow [Atmel2013]



Corresponding Security Issues [Atmel2013]



Attack Objectives

- Execute firmware on **unauthorized** device
- Load **3rd-party** (malicious?) firmware
- **Alter** the firmware, inc.:
  - **Prevent** future **updates**
    eg to prevent future protections against security flaws.
  - Bring the device to **DoS**
- **Reverse-engineer** the firmware

Existing SW Protections

Integrity

Check against FW alteration	Hash Signature MAC

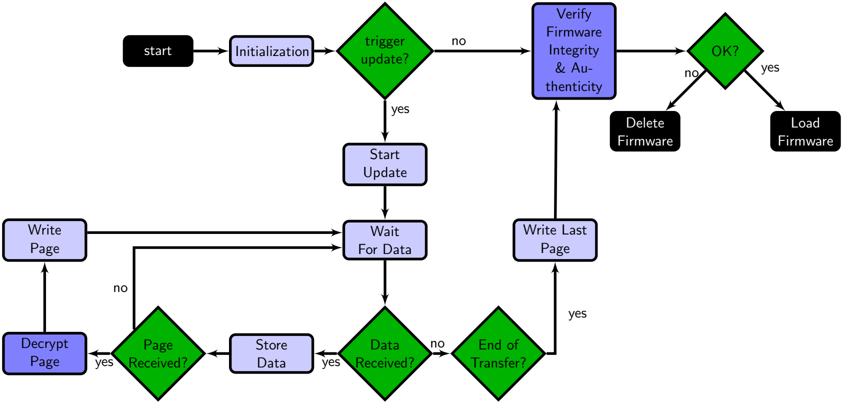
Authenticity

Check against 3rd-party FW	Signature MAC
Execute only on authorized device	Encryption

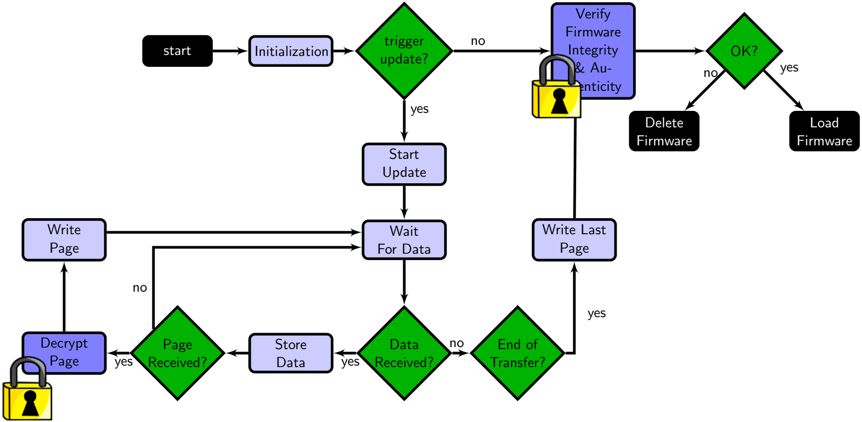
Confidentiality

Make reverse-engineering hard	Encryption

Secured BL-FU Control Flow [Atmel2013]



Secured BL-FU Control Flow [Atmel2013]



These protections are only efficient to protect against “logical” attacker ....

What about physical attacks?

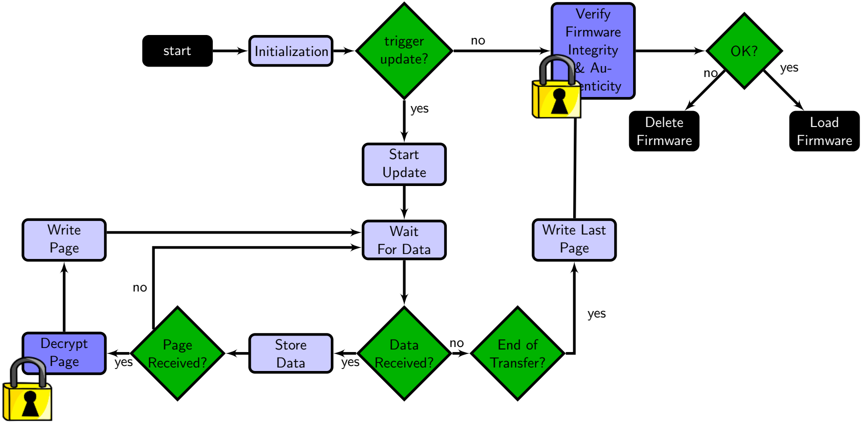
In particular Fault attacks?

Many fault models available:

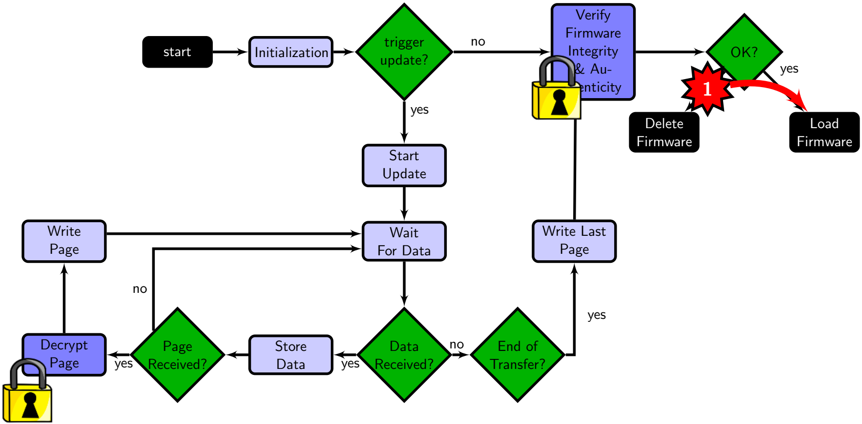
- Instruction skip
- Control flow
- Variable modifications (registers, memory)
- Attacks on encryption function

$\Rightarrow$ What consequences on the BL-FU process?

Potential fault Attacks on BL-FU

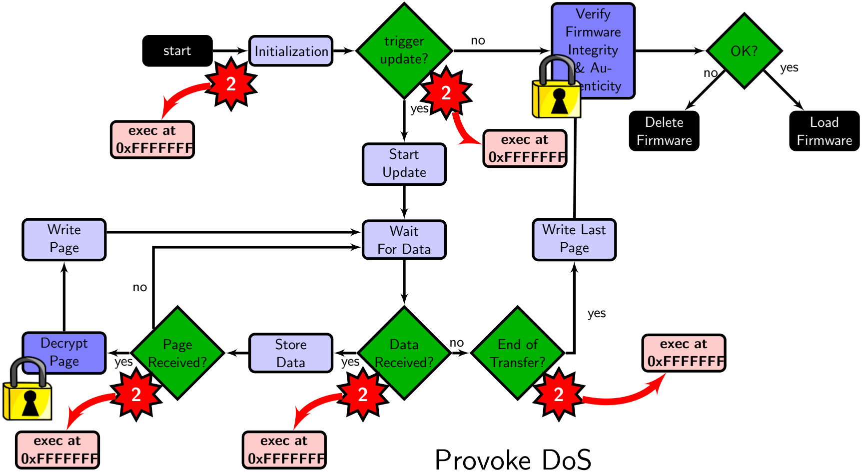


Potential fault Attacks on BL-FU



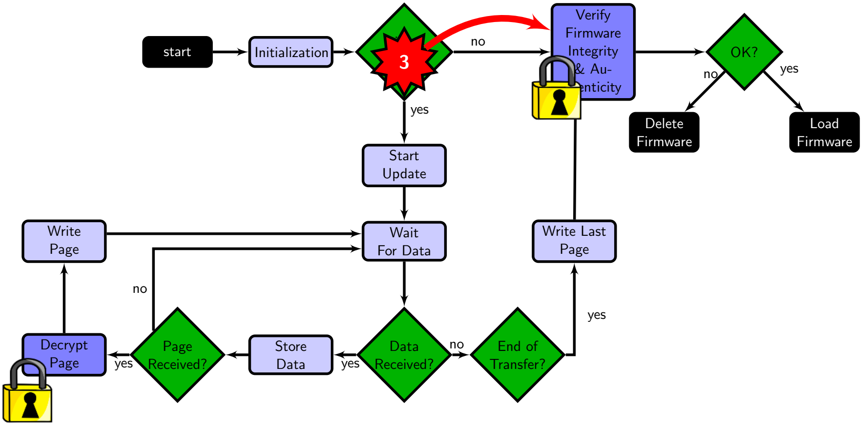
Load a code that is not authenticated

Potential fault Attacks on BL-FU



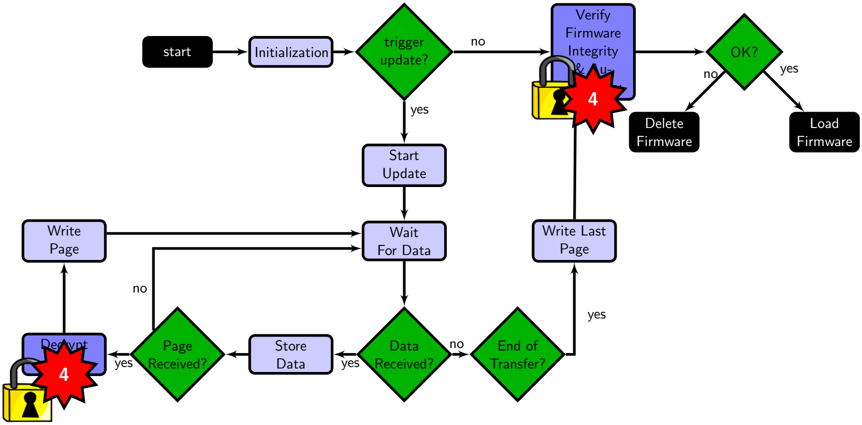
Provoke DoS

Potential fault Attacks on BL-FU



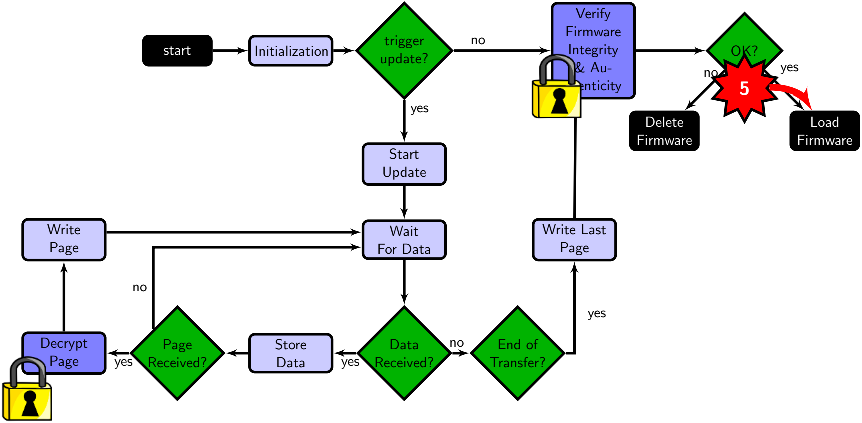
Prevent further FUs

Potential fault Attacks on BL-FU



Build attacks on crypto functions

Potential fault Attacks on BL-FU



Prevent firmware's suicide

Summary of Fault Models on BL-FU

- Load a code that is not authenticated
- Provoke DoS
- Prevent further FUs
- Build attacks on crypto function
- Prevent Firmware's suicide

Goals of the IRT-NanoElec CLAPS project

- **on-going** Develop a BL/FU, starting from simplistic version
- **on-going** Define fault models
- Analyse at source-level with Lazart
- Apply compiler-level counter-measures
- Build mechanisms for attack-detection, based on:
  - machine-learning models
  - runtime verification / monitoring
- Evaluate counter-measures with laser testbeds

Thanks!
Questions?

leti
Centre de Grenoble
17 rue des Martyrs
38054 Grenoble Cedex

list
Centre de Saclay
Nano-Innov PC 172
91191 Gif sur Yvette Cedex

- Atmel.
  At02333: Safe and secure bootloader implementation for sam3/4.
  Application Note, 2013.