Inter-CESTI:
Methodological and Technical Feedback on Hardware Devices Evaluations

Symposium sur la Sécurité des Technologies de l’Information et des Communications

ANSSI, Amossys, EDSI, LETI, Lexfo, Oppida, Quarkslab, SERMA, Synacktiv, Thales, Trusted Labs

5 June 2020
Introduction
Focus on the WooKey platform
Project start-up
Attacks
Conclusion
Introduction: *certification*
Goal: Provide assurance that the product is secured enough

- Verify that the product does what is intended
- Pentest the product to assess the robustness of security functions
- Use evaluation criteria and methodology
**Actors involved in product certification**

- Developer/Sponsor
- ITSEF
- Certifier
Who am I?

Developer/Sponsor -> ITSEF

ANSSI CCN
Who are they?

- Developer/Sponsor
- ITSEF

ITSEF

ANSSI CCN
ITSEFs: WHO ARE THEY?
Hardware ITSEFs

Products

Tools
Software ITSEFs

Inter-CESTI Feedback - June 2020 - Rennes
SOFTWARE ITSEFs

Products

- VPN
- Secure Connection
- AMOSSYS
- LEXFO
- OPPIDA
- Quarkslab
- SYNACKTIV
- Trusted Labs

Tools

- Inter-CESTI: the origins
- Inter-CESTI Feedback - June 2020 - Rennes
Software ITSEFs

**Products**
- VPN
- Secure Connection

**Tools**
- Wireshark
- Metasploit

**Security Certification**
- ITSEF

**Methodology**
- Inter-CESTI: the origins

**Attacks**
- SSL Secure Connection

**Conclusion**
- Products
- Tools
Introducing the WooKey platform:

**Methodology**
- ITSEFs
- Inter-CESTI: the origins

**Attacks**
- Security certification

**Conclusion**

***ITSEFs: what about these?***

Hardware devices:
- Inter-CESTI
  - common target
  - use cheap material
  - hardware + software attacks
Introducing the Inter-CESTI

“Hardware devices”

Inter-CESTI:
- **common** target
- use **cheap** material
- hardware + **software** attacks
WooKey: the test vehicle
Why WooKey?

- WooKey platform (presented at SSTIC 2018) fitted perfectly:
  - Open source software and hardware
  - A lot of security features
  - Numerous external interfaces
  - Knowledge of the product
---

# Hardware Architecture

- 2 MB of flash, 192 kB of SRAM
- Internal firmware
- **MPU**: Memory Protection Unit
- Hardware AES
- Deactivation of *debug* interfaces

![MCU = Cortex-M4 STM32F439](image)
**Hardware Architecture**

Token extractable

- Cortex-M4 STM32F439
- 2 MB of flash
- 192 kB of SRAM
- Internal firmware
- MPU: Memory Protection Unit
- Hardware AES
- Deactivation of debug interfaces
- PIN CODE:
  - 1
  - 6
  - 0
  - 8
  - 3
  - 5
  - 4
  - 2
  - 7
  - 9
  - ✗
Introduction

Focus on the WooKey platform

Methodology

Attacks

Conclusion

Overview

Hardware architecture

Software architecture

Module Token

Module PIN

Module Crypto

Module USB

Module SDIO

Master Key

Two factors

authentication

WooKey

Bootloader

check

and boot

Platform Secrets

EwoK Microkernel
**Introduction**

Focus on the WooKey platform

**Methodology**

**Attacks**

**Conclusion**

**Overview**

Hardware architecture

Software architecture

**Modules and services of WooKey**

- **Master Key**
- **Two factors authentication**
- **Module PIN**
- **Module Token**
- **Module Crypto**
- **Module SDIO**
- **Module USB**

**WooKey Bootloader**

- **Platform Secrets**
- **EwoK Microkernel**

Two factors authentication

Master Key

check and boot
Methodology details
In the context of the ITSEF security evaluation, we focus on the WooKey platform. The methodology for the evaluation includes testing the security target, evaluating cryptographic supplies, and analyzing WooKey platforms samples. The tests are conducted over a period of 20 days, with plans for feedback and further tests.

- Security target
- Cryptographic supplies
- WooKey platforms samples
**Inter-CESTI timeline**

- **T0**: Security target
  - Cryptographic supplies
  - WooKey platforms samples

- **T1**: Test plan
  - Comments on the Security target

---

9/29

Inter-CESTI Feedback – June 2020 – Rennes
Introduction

Focus on the WooKey platform

Methodology

Attacks

Conclusion

**Inter-CESTI timeline**

- **T0**
- **T1**
- **T2**

20 days

- **Security Functions to evaluate**

- Security target
  - Cryptographic supplies
  - WooKey platforms samples

- Test plan

- Comments on the Security target

- Security Functions to evaluate

- Software
  - ITSEF

- Hardware
  - ITSEF

- Software
  - Attacks

- Hardware
  - Attacks

9/29

Inter-CESTI Feedback - June 2020 - Rennes
**Inter-CESTI timeline**

- **T0**
- **T1**
- **T2**

**Software ITSEF**

- **Software Attacks**

**Hardware ITSEF**

- **Hardware Attacks**

---

**Comments on the Security target**

- **T0**
- **T1**
- **T2**

**Security Functions to evaluate**

- **Software**
- **Hardware**

---

**Security target**

- **Cryptographic supplies**
- **WooKey platforms samples**

**Test plan**

- **20 days**

---

**Inter-CESTI Feedback - June 2020 - Rennes**
**Introduction**

Focus on the WooKey platform

**Methodology**

Attacks

**Conclusion**
Way *too many assets* and *security functions*
Way too many assets and security functions
Introduction

Focus on the WooKey platform

Methodology

Attacks

Conclusion

Inter-CESTI timeline

T0

T1

T2

T3

25 days

Report delivery

Feedbacks and restitution

Inter-CESTI Feedback - June 2020 - Rennes
Identified attacks scope

- ✗ Software attacks (pre and post-auth)
- ✗ Pre-auth hardware attacks
- ✗ Stealthy post-auth hardware attacks

On the platform and the AUTH and DFU tokens
## Selection of attack paths

<table>
<thead>
<tr>
<th>Software</th>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Static analysis and fuzzing of exposed code</td>
<td>- Side-channel attacks (SCA)</td>
</tr>
<tr>
<td>- Analysis of the Bootloader</td>
<td>- Fault injection attacks (FIA)</td>
</tr>
<tr>
<td>- MPU policies analysis</td>
<td>- Eavesdropping/injection on buses</td>
</tr>
<tr>
<td>- Javacard applets analysis</td>
<td>- TEMPEST</td>
</tr>
</tbody>
</table>
## Selection of attack paths

### Software
- Static analysis and fuzzing of exposed code
- Analysis of the Bootloader
- MPU policies analysis
- Javacard applets analysis

### Hardware
- Side-channel attacks
- Fault injection
- Eavesdropping/injection on buses
- TEMPEST

---

Use “cheap” material to fit in the CSPN constraints
Attacks details
A comprehensive list of attacks

<table>
<thead>
<tr>
<th>15 different attacks (see article)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All found and performed by ITSEFs</td>
</tr>
</tbody>
</table>

**Transparency initiative**

- Security target available
- Attacks details in the article:
  - Tools, settings and timings of attacks
  - Reproducible methodology
  - Mitigations

Publicly available
No direct attack path found

- Only partial attacks
- Seems like stealing once a WooKey will not allow much
- Multiple pilferage attacks needed
  - Practical attacks require physical access and cloning/trapping
  - Time required to perform cloning/trapping

Defense in depth seems useful!

0

- Steal

1

- Attack & clone/trap (takes time)
- Give back

2

- Capture Master key
- Steal again

3

- Recover Master key
**Attacks overview (from the article)**

<table>
<thead>
<tr>
<th></th>
<th>static code analysis/review</th>
<th>Software exploitation</th>
<th>Software fuzzing</th>
<th>Hardware fuzzing</th>
<th>MPU Analysis</th>
<th>Bus sniffing</th>
<th>Bus injection</th>
<th>Crypto attack</th>
<th>SCA</th>
<th>FIA</th>
<th>TEMPEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Javacard applet analysis</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>libiso7816 and libtoken fuzzing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>libiso7816 glitch attacks</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>EwoK privilege escalation</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>MPU configuration review</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>PetPIN bruteforce attack</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>Secure Channel review</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>ECDSA physical attacks</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>HMAC physical attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Bootloader RDP2 downgrade</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Bootloader EM Faults</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Bootloader Anti-rollback bypass</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SDIO bus analysis</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>SPI bus analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SPI TEMPEST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
# Attacks Overview (From the Article)

<table>
<thead>
<tr>
<th></th>
<th>static code analysis/review</th>
<th>Software exploitation</th>
<th>Software fuzzing</th>
<th>Hardware fuzzing</th>
<th>MPU Analysis</th>
<th>Bus sniffing</th>
<th>Bus injection</th>
<th>Crypto attack</th>
<th>SCA</th>
<th>FIA</th>
<th>TEMPEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Javacard applet analysis</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>libiso7816 and libtoken fuzzing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>libiso7816 glitch attacks</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>EwoK privilege escalation</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>MPU configuration review</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>PetPIN bruteforce attack</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>Secure Channel review</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>ECDSA physical attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>HMAC physical attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Bootloader RDP2 downgrade</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Bootloader EM Faults</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Bootloader Anti-rollback bypass</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SDIO bus analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>SPI bus analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SPI TEMPEST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Attacks with cloning and trapping**

**Attacks with stealthy spying and stealing**
**Goal:** Get the **Platform Secrets**, then **clone and trap** to get the Master Key
Software: code analysis and **fuzzing** didn’t reveal any vulnerability

<table>
<thead>
<tr>
<th>Filename</th>
<th>Function Coverage</th>
<th>Line Coverage</th>
<th>Region Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>fuzzing_javalang/libjpeg/src/mm/mm.c</td>
<td>8.00% (0/1)</td>
<td>0.00% (0/5)</td>
<td>0.00% (0/3)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/utils/utils.c</td>
<td>8.00% (0/1)</td>
<td>0.00% (0/6)</td>
<td>0.00% (0/1)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/soft/soft.c</td>
<td>85.71% (6/7)</td>
<td>46.26% (185/227)</td>
<td>34.01% (50/147)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/soft_unpooled.c</td>
<td>66.67% (8/12)</td>
<td>54.41% (192/261)</td>
<td>58.23% (46/79)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/fuzzing.c</td>
<td>100.00% (6/6)</td>
<td>100.00% (28/28)</td>
<td>100.00% (12/12)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/hmac.c</td>
<td>100.00% (4/4)</td>
<td>74.87% (100/135)</td>
<td>77.62% (46/62)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/libtoken.h</td>
<td>8.00% (0/2)</td>
<td>0.00% (0/19)</td>
<td>0.00% (0/2)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/platform_platform.c</td>
<td>66.67% (18/15)</td>
<td>60.42% (294/481)</td>
<td>66.67% (10/15)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/smartcard.c</td>
<td>50.00% (7/14)</td>
<td>34.35% (181/527)</td>
<td>49.91% (126/250)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/smartcard_iso8186.c</td>
<td>82.00% (41/50)</td>
<td>79.64% (1604/2014)</td>
<td>82.01% (539/665)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/token.c</td>
<td>88.95% (17/21)</td>
<td>75.00% (759/1012)</td>
<td>79.40% (468/589)</td>
</tr>
<tr>
<td>fuzzing_javalang/libjpeg/src/token_dfu.c</td>
<td>100.00% (2/2)</td>
<td>90.76% (39/43)</td>
<td>88.89% (16/18)</td>
</tr>
<tr>
<td><strong>Totals</strong></td>
<td><strong>74.81% (101/135)</strong></td>
<td><strong>69.28% (3017/4355)</strong></td>
<td><strong>72.03% (1715/2381)</strong></td>
</tr>
</tbody>
</table>

Hardware: ITSEF successfully exploited **power glitches**
**Vulnerability:**

- A **glitch** during a masking instruction allows a **buffer overflow**
- + Stack canaries misconfiguration

⇒ **Code execution** in the **SMART** task

```c
int SC_get_ATR (SC_ATR * atr) {
    /* Get the historical bytes */
    atr->h_num = atr->t0 & 0x0f;
    for (i = 0; i < atr->h_num ; i++) {
        if (SC_getc_timeout(&(atr->h[i]), WT_wait_time)) {
            goto err;
        }
        checksum ^= atr->h[i];
    }
    [...]
}
```

- Demonstration of a **hybrid attack**
Fuzzing syscalls revealed kernel privilege escalation

- Error in parsing the parameter of one syscall
  ➞ Deactivation of MPU

Coupled with libiso7816 glitch attack:

- attacker can modify the firmware in place
- trapping a closed platform is possible
**Goal:** Get the Platform Secrets, then clone and trap to get the Master Key
Vulnerabilities:

- One FIA on the STM32 for the **RDP level downgrade**
- One FIA on WooKey Bootloader to **bypass the RDP level verification**
**Goal:** Exploit **vulnerable firmware** using **version downgrade**
About: formal methods used for vulnerability analysis

- Software:
  - Frama-C used on Bootloader source code, but no vulnerability (RunTime Errors) found!

- Hardware:
  - Lazart, which simulates FIA found exploitable path in firmware version check
  - Exploited using a voltage glitch
**Goal:** Get user PIN using EM leaks, then steal the platform and token.
SPI bus between screen and PCB shows TEMPEST leaks
More a characterization than a full attack
Conclusion
## Conclusion

### Inter-CESTI feedback

- **Challenging** for all entities
- Attacks efficiently performed by all ITSEFs *(beyond their specialization)*
- Results encourage the creation of a **Hardware Device CSPN domain**

### Attacks feedback

- Cheap physical attacks quite easily achievable
- Hybrid attack paths and approaches are efficient
- Using accessible equipment with CSPN in mind

![](important.png) This equipment will **never scale** for more hardware secured products (HSM, banking cards, etc.)
## Conclusion

### Inter-CESTI feedback
- **Challenging** for all entities
- Attacks efficiently performed by all ITSEFs *(beyond their specialization)*
- Results encourage the creation of a Hardware Device CSPN domain

### Attacks feedback
- Cheap physical attacks quite easily achievable
- Hybrid attack paths and approaches are efficient
- Using accessible equipment with CSPN in mind

### WooKey project feedback
- Very interesting technical discussions
- New commits on WooKey’s github: [https://github.com/wookey-project](https://github.com/wookey-project)

This equipment will **never scale** for more hardware secured products (HSM, banking cards, etc.!)
Inter-CESTI: Questions?

ANSSI, Amossys, EDSI, LETI, Lexfo, Oppida, Quarkslab, SERMA, Synacktiv, Thales, Trusted Labs