Shaping the Glitch: Optimizing Voltage Fault Injection Attacks

Conference on Cryptographic Hardware and Embedded Systems 2019

Claudio Bozzato
Riccardo Focardi
Francesco Palmarini

1Ca’ Foscari University of Venice, 2Cryptosense, 3Yarix, 4Talos

August 28, 2019
Atlanta, USA
Fault what?

- Exploits **hardware vulnerabilities** to “create” new bugs
- Influence (inject) a system with internal / **external stimuli**
- **Alter** the intended **execution flow** / behavior
- **Skip instructions**, influence branch decisions, corrupt memory locations, etc.
- Bypass security checks, leak data or crypto material, create side-channels, etc.
- **Non-invasive to invasive** techniques: clock, voltage, EM, FIB, laser, heat, flash, etc.
The most widespread Voltage Fault Injection setup [OC14]

Very easy to setup and low-cost

- Low control over glitch parameters
- Unpredictable: the glitch characteristics depends on circuit properties, MOSFET, etc.
Voltage Fault Injection... The MOSFET Way

- The most widespread Voltage Fault Injection setup [OC14]
- Very easy to setup and low-cost
- Low control over glitch parameters
- Unpredictable: the glitch characteristics depends on circuit properties, MOSFET, etc.
Voltage Fault Injection... The MOSFET Way

- The most widespread Voltage Fault Injection setup [OC14]
- Very easy to setup and low-cost
- Low control over glitch parameters
- Unpredictable: the glitch characteristics depends on circuit properties, MOSFET, etc.
Voltage Fault Injection... The MOSFET Way

- The most **widespread** Voltage Fault Injection setup [OC14]
- Very **easy** to setup and low-cost
- **Low control** over glitch parameters
- **Unpredictable:** the glitch characteristics depends on circuit properties, MOSFET, etc.
Our Idea: Arbitrary Glitch Waveforms

DESIDERATA

✓ Stable and repeatable results
✓ High degree of freedom in glitch generation
✓ Software managed attack parameters
✓ Low-cost and easy to build setup
Our Idea: Arbitrary Glitch Waveforms

DAC-based glitch generator

- Arbitrary Waveform Generator
- Analogue Waveform
- Trigger
- Digital Waveform

- Glitch Amplifier
- Glitch injection

- STM32F407
- I/O & Timing Controller
- I/O
- USB API

Python Framework

- Attack Logic
- Bootloader Protocol
- Genetic Algorithm
Our Idea: Arbitrary Glitch Waveforms

✓ Rising and falling edges affect V-Fl performance [ZDCR14]

? What if different devices / attacks need different glitch waveforms?

? How do we identify the best match?

DAC-based glitch generator
AGW: with big power comes lots of parameters

- Power **supply voltage** with < 10mV resolution
- Glitch **shape** and voltage in 2048 points
- Injection **timing** with ~20ns accuracy
- Glitch frequency / **duration**

→ Need for automatic parameter search and optimization!
AGW: with big power comes lots of parameters

- Power **supply voltage** with < 10mV resolution
- Glitch **shape** and voltage in 2048 points
- Injection **timing** with ~20ns accuracy
- Glitch frequency / duration

→ **Genetic Algorithm** (Selection, Crossover, Mutation, Replacement)
AGW: with big power comes lots of parameters

- Power **supply voltage** with < 10mV resolution
- Glitch **shape** and voltage in 2048 points
- Injection **timing** with ~20ns accuracy
- Glitch frequency / **duration**

→ **Cubic interpolation**
AGW: with big power comes lots of parameters

- Power **supply voltage** with < 10mV resolution
- Glitch **shape** and voltage in 2048 points
- Injection **timing** with ~20ns accuracy
- Glitch frequency / **duration**

→ Digital-to-Analog conversion
AGW: with big power comes lots of parameters

- Power supply voltage with < 10mV resolution
- Glitch shape and voltage in 2048 points
- Injection timing with ~20ns accuracy
- Glitch frequency / duration

→ Precise glitch triggering
Case Study: Renesas 78K Firmware Extraction

- Widely used by the **automotive** industry
- 32 to 256KB **integrated flash memory** for firmware / data
- **Internal bootloader** for flash programming via PC
- No knowledge on the firmware / bootloader code → **Blackbox**
- Bootloader protocol exposes a set of **API** via serial interface
  - **Program**
  - **Erase**
  - **Checksum**
  - **Verify**
- **Built-in security mechanisms:**
  - Commands operate on **256 bytes aligned memory** blocks
  - All programming and erasing **commands can be disabled**
  - **Voltage Supervisor** / BOR
Step I: Finding Vulnerabilities

- No `read` command... **Fail 😞**
- Use `FI` to verify just one byte... **Fail 😞**
- Use `FI` to calculate the checksum of one byte... **Fail 😞**
- Use `FI` to calculate the **checksum of 4 bytes** (aligned)...  
- Use `FI` to **verify 4 bytes** (aligned)...

$\text{Checksum(B1, B256)} = 0x10000 - B1 - B2 - B3 - \ldots - B255 - B256$
Step I: Finding Vulnerabilities

- No read command... **Fail ☹**
- Use FI to verify just one byte... **Fail ☹**
- Use FI to calculate the checksum of one byte... **Fail ☹**
- Use FI to calculate the **checksum of 4 bytes** (aligned)... **Success 😊**
- Use FI to verify 4 bytes (aligned)... **Success 😊**

\[
\begin{array}{c|c|c|c|c|c|c|c|c}
\end{array}
\]

\[0x10000 - \{B1...B4\} = 0xFF9A\]

\[\text{Verify(}0xAA...0xDD) = \text{True/False}\]
Step II: Leaking Flash Memory

- More leaks required → more faults
- **Side-channel** from the *checksum* computation?

```python
def checksum(start, end):
    if (end != start + 256):
        raise
    result = 0x10000
    for i in range(start, end + 1):
        result = result - flash[i]
    return result
```

\[ 0x10000 - \{B1...B4\} = 0xFF9A \]
Step II: Leaking Flash Memory Content

- More leaks required → more faults
- Side-channel from the checksum computation?

```python
def checksum(start, end):
    if (end != start + 256):
        raise
    result = 0x10000
    for i in range(start, end + 1):
        result = result - flash[i]
    return result
```

0x10000 - B1 - B3 - B4 = 0xFFAB
0xFF9A - 0xFFAB = 0x11
Step III: Deal With Timing Errors

- What is the extracted value for B3?
  - 0x22 with ~10% probability
  - 0x33 with ~4% probability
  - 0x11 with ~3% probability
  - 0x00 with <1% probability
  - 0x55 with <1% probability
  - Plus the false positives!

Glitch trigger

Just inject a fault for every byte, right? **Nope.**
Step IV: Mount the Full Attack

- Calculate the sum of $B_1 + B_2 + B_3 + B_4 = 0x66$
- For each extracted candidate byte $B_x$:
  - Find all the 4-bytes permutations with $B_x$
  - Discard permutations which do not sum to $0x66$
  - Glitch the verify command to test each new permutation
  - Stop when the verify is successful
- Iterate for {B5...B8} {B9...B12} ... until the flash is dumped! MANY hours later...

<table>
<thead>
<tr>
<th>Candidate #1</th>
<th>Candidate #2</th>
<th>Candidate #3</th>
<th>Candidate #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 33 00 22</td>
<td>00 00 22 78</td>
<td>01 32 00 33</td>
<td>06 11 22 33</td>
</tr>
</tbody>
</table>
Step V: Compensate for Temperature Errors

Bootloader runs from internal oscillator

The RC oscillator drifts with temperature

The rate is about 0.1% / °C

With +6 °C the trigger moved by > 4 us

Solved by software compensation

Let the attack go day and night, right? **Not that easy.**
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6d 19h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3d 16h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2d 12h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used ~2x the number of glitches and MOSFET ~5x
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Comparison of the Renesas attack performance for three major V-FI techniques.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Tested combinations</th>
<th># ShortVerify</th>
<th># ChecksumLeak</th>
<th># ShortChecksum</th>
<th>Total glitch count</th>
<th>Total dump time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mosfet</td>
<td>351 k</td>
<td>13.9 M</td>
<td>3.1 M</td>
<td>699 k</td>
<td>18.1 M</td>
<td>6 d 19 h</td>
</tr>
<tr>
<td>Pulse</td>
<td>142 k</td>
<td>3.8 M</td>
<td>2.6 M</td>
<td>582 k</td>
<td>7.1 M</td>
<td>3 d 16 h</td>
</tr>
<tr>
<td>AGW</td>
<td>105 k</td>
<td>1.5 M</td>
<td>1.5 M</td>
<td>351 k</td>
<td>3.3 M</td>
<td>2 d 12 h</td>
</tr>
</tbody>
</table>

- **Speed**: our technique is **32% faster** than PULSE and **63% faster** than MOSFET
- **Efficiency**: PULSE used \( \sim 2x \) the number of glitches and MOSFET \( \sim 5x \)
- **Reliability**: AGW produces **30% the number of false positives** than MOSFET
Different glitch waveforms provide the best performance for different vulnerabilities.

Evaluation and comparison
Comparison of the glitch waveforms / techniques for the Renesas attack.

Evaluation and comparison
Contributions

- Studied the effects of Arbitrary Glitch Waveforms on the performance of V-FI
- Investigated on the feasibility of automatic attack parameter selection and optimization using Genetic Algorithms
- Found unpublished vulnerabilities that enable firmware extraction attacks for six microcontrollers from three major silicon manufacturers:
  - STMicroelectronics - STM32F1 & STM32F3
  - Texas Instruments - MSP430 F5xx & MSP430 FRAM
  - Renesas Electronics - 78K0/Kx2 & 78K0R/Kx3-L
- In-depth analysis and evaluation of the attack performance compared to other V-FI techniques
THANK YOU!
References


