A Quick Intro to Physical Attacks

Embedded Physical Attacks 101 @ HardwearIO

Lejla Batina
Kostas Papagiannopoulos

Radboud University, The Netherlands
NXP Semiconductors, Germany

Outline

How did we get here?


ROCK

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed
- We moved away from ‘closed’ approaches in favor of an open community of cryptanalysis

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed
- We moved away from ‘closed’ approaches in favor of an open community of cryptanalysis
- Work ow:

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed
- We moved away from 'closed' approaches in favor of an open community of cryptanalysis
- **Work ow:**
  1. Construct a cipher

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed
- We moved away from 'closed' approaches in favor of an open community of cryptanalysis
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks

Crypto in the 70s and 80s

- The birth of modern cryptography (and rock music)
- DES (1975), RSA (1977) and many other early designs surfaced
- Pure mathematical structures were proposed and analyzed
- We moved away from 'closed' approaches in favor of an open community of cryptanalysis
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Put it on the mainframe


pngtree.com

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secure and computationally e cient

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secureand computationally e cient
- Work ow:

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secureand computationally e cient
- Work ow:
  1. Construct a cipher

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secure and computationally e cient
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secure and computationally e cient
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is fast

Crypto in the 90s and 00s

- The advent of personal computing, the Internet (and hiphop music)
- Everything is connected on the network
- Every computer was vulnerable unless strong encryption was used
- Consensus was reached: every device should be well protected, so lets build AES (1998)
- AES should be mathematically secure and computationally e cient
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is fast
  4. Put it on your PC

Crypto in the 10s and 20s

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:
  1. Construct a cipher

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is very e cient

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is very e cient
  4. Put it on every imaginable device

Crypto in the 10s and 20s

- Smartphone revolution, the Internet of Things (and pop music)
- An always-online army of embedded devices
- Smartphones, smartcards, USB dongles, RFID transport cards
- Every single one of them containing critical data
- Every single one of them is given to a potential attacker
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is very e cient
  4. Put it on every imaginable device
  5. Give such devices to everyone

Crypto in the 10s and 20s

- This was the breaking point

Crypto in the 10s and 20s

- This was the breaking point
- Hands-on device access was easy

Crypto in the 10s and 20s

- This was the breaking point
- Hands-on device access was easy
- Everyone could try hardware hacking!

Crypto in the 10s and 20s

- This was the breaking point
- Hands-on device access was easy
- Everyone could try hardware hacking!
- Many products were found insecure!

Crypto in the 10s and 20s

- This was the breaking point
- Hands-on device access was easy
- Everyone could try hardware hacking!
- Many products were found insecure!

What are Physical Attacks?

Input: 4-digit PIN code, Output: PIN verified or rejected

Process CheckPIN (pin[4])

```
int pin_ok=0;
if (pin[0]==5)
    if (pin[1]==9)
        if (pin[2]==0)
            if (pin[3]==2)
                pin_ok=1;
            end
        end
    end
end
return pin_ok;
```

Side-Channel Attacks

Input: 4-digit PIN code, Output: PIN verified or rejected

Process CheckPIN (pin[4])

```
int pin_ok=0;
if (pin[0]==5)
    if (pin[1]==9)
        if (pin[2]==0)
            if (pin[3]==2)
                pin_ok=1;
            end
        end
    end
end
return pin_ok;
```

- What are the execution times of the process for PIN inputs [0,1,2,3], [5,3,0,2], [5,9,0,0]

Side-Channel Attacks

Input: 4-digit PIN code, Output: PIN verified or rejected

```
Process CheckPIN (pin[4])

int pin_ok=0;
if (pin[0]==5)
    if (pin[1]==9)
        if (pin[2]==0)
            if (pin[3]==2)
                pin_ok=1;
            end
        end
    end
end
return pin_ok;
```

- What are the execution times of the process for PIN inputs [0,1,2,3], [5,3,0,2], [5,9,0,0]
- The execution time increases as we get closer to [5,9,0,2]

Side-Channel Attacks

- A Side-Channel Attack observesdata from the physical device, then uses it to bypass security

Side-Channel Attacks

- A Side-Channel Attack observesdata from the physical device, then uses it to bypass security
- Work ow:

Side-Channel Attacks

- A Side-Channel Attack observesdata from the physical device, then uses it to bypass security
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is that is resists side-channel attacks
  4. Give such devices to everyone

Side-Channel Attacks

- A Side-Channel Attack observesdata from the physical device, then uses it to bypass security
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is that is resists side-channel attacks
  4. Give such devices to everyone
- Several side-channels exist: timing, power consumption, electromagnetic emission of the device

Side-Channel Attacks

```
Input: 128-bit plaintext, Output: 128-bit ciphertext

Process AES (plaintext)

for round = 1 until 10
  AddRoundKey;
  Sbox;
  ShiftRows;
  MixColumns;
end
return ciphertext;
```

Fault Injection Attacks


```
Input: 128-bit plaintext, Output: 128-bit ciphertext

Process AES (plaintext)

for round = 1 until 10
  AddRoundKey;
  Sbox;
  ShiftRows;
  MixColumns;
end
return ciphertext;
```

- What if we glitch the voltage just enough to zero the `round' variable?

Fault Injection Attacks


```
Input: 128-bit plaintext, Output: 128-bit ciphertext

Process AES (plaintext)

for round = 1 until 10
  AddRoundKey;
  Sbox;
  ShiftRows;
  MixColumns;
end
return ciphertext;
```

- What if we glitch the voltage just enough to zero the `round' variable?
- Then we essentially stop the AES encryption

Fault Injection Attacks

- A Fault Injection Attack alters data in the physical device, then uses this to bypass security

Fault Injection Attacks

- A Fault Injection Attack alters data in the physical device, then uses this to bypass security
- Work ow:

Fault Injection Attacks

- A Fault Injection Attack alters data in the physical device, then uses this to bypass security
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is that is resists side-channel and fault injection attacks
  4. Give such devices to everyone

Fault Injection Attacks

- A Fault Injection Attack alters data in the physical device, then uses this to bypass security
- Work ow:
  1. Construct a cipher
  2. Make sure it resists mathematical attacks
  3. Make sure it is that is resists side-channel and fault injection attacks
  4. Give such devices to everyone
- Several fault injection techniques exist: power glitching and electromagnetic glitching, laser injections

Fault Injection Attacks

What do we cover in this course?

Crypto in this course

- Crypto reminders : we will discuss standard algorithms such as AES and RSA
- No need to implement them, we assume they are already in place, inside embedded devices
- We need to remember their basic structure: so try to freshen them up in Wikipedia ,

Side-Channels in this course

- Data analysis: We will provide you with side-channel measurements for several devices, then we will teach you how to code standard side-channel attacks
- We will use the developed code to analyse measurements and extract the secret crypto key
- We will cover most standard side-channel techniques such as simple power analysis, di erential power analysis and template attacks
- Data capturing: We also show how to perform side-channel measurements on the ChipWisperer device
- We recommend coding in Matlab or Octave: So do try to freshen your basic coding skills,

Fault Injection in this course

- Using the ChipWhisperer setup we will produce voltage-glitching e ects
- We will use them to recover the secret key of RSA
- We recommend coding in Python: So do try to freshen your basic coding skills of standard Python and NumPy,