Who do you trust?
What if the *user* is the attacker?
Physical Attacks

Direct access to a chip is possible: signals can be injected, modified, or measured
- “SECRET ROM” STORED IN MCPX

- FLASH DECRYPTION KEY (RC4) STORED IN “SECRET ROM”
**Active**

- Inject new signals
- Modify existing signals in new ways

**Passive**

- No modification of signals
- Only observe regular operation
Fig. 7.3 Decapsulated chips
Fig. 7.6 Laser scan of unpowered and powered-up SRAM in PIC16F84 microcontroller

Fig. 7.7 Layout of SRAM cell and SRAM area in PIC16F84 microcontroller
Some Common Terms

**JTAG**
Joint Test Action Group: A debug interface for testing devices. If this is left enabled on a product, you can do fun things. See: J-Link, Jtagulator.

**SWD**
Single Wire Debug: The ARM debugger protocol built on JTAG. Think GDB but for embedded systems. May be disabled by security bit (this can be glitched).

**Flash**
(Sometimes Built-in) storage that the microcontroller uses to store firmware, code, data, etc. If external, can be dumped. May be encrypted.

**Security Bit**
Setting that disables code readout on an MCU. Allows manufacturers to leave debug ports (like SWD) on the PCB without worrying about us dumping their code. Can be glitched.

**Boot ROM**
Read-only code that initiates CPU bringup. This is fixed in silicon and cannot be modified. Bugs here are nearly always catastrophic for system security.
4 Attacks

Today
Fault Injection

Power Analysis

Coldboot

Timing Analysis
Fault Injection
Coldboot

Active

Passive

Power Analysis
Timing Analysis
Fault Injection
BLAST CHIPS WITH THIS BBQ LIGHTER FAULT INJECTION TOOL

by: Dan Maloney

January 29, 2022

Looking to get into fault injection for your reverse engineering projects, but don’t have the cash to lay out for the necessary hardware? Fear not, for the tools to glitch a chip may be as close as the nearest barbecue grill.
Notable Examples

AirTag
Lose your knack for losing things.
Voltage Glitching

Cut the power at the exact right time to make something go wrong
Voltage Glitching

Challenge

Need to deal with capacitors, which filter out our attack.
Clock Glitching

Oscillator

Spring

Spring

Ground
Crystal Oscillator
Crystal Oscillator

Inject Fault Here
void main () {
    int iter = 0;
    while(true) {
        int chksum = compute_checksum();
        print("Locked! %d %d", chksum, iter);
        iter++;
    }
    print("MIT{flag}");
}

Timing Analysis
Spot the Bug

bool memcmp (char *buf1, char *buf2, size_t len) {
    for (int i = 0; i < len; i++) {
        if (buf1[i] != buf2[i]) {
            return false;
        }
    }
    return true;
}
Spot the Bug

bool memcmp (char *buf1, char *buf2, size_t len) {
    for (int i = 0; i < len; i++) {
        if (buf1[i] != buf2[i]) {
            return false;
        }
    }
    return true;
}
No Demo:
You will do this in recitation next week!
Coldboot
MOS Capacitor

- Metal
- Oxide (SiO₂)
- Doped Semiconductor (n or p type Si)
MOS Capacitor

- Metal
- Oxide (SiO$_2$)
- Doped Semiconductor (n or p type Si)
Demo
Power Analysis
How can you measure current on an oscilloscope?
Apply Ohm's Law

Voltage (V) = Current (I) * Resistance (R)

Or in other words,

I = V / R
Shunt (100Ω)
Pay attention to probe grounding!
Simple Power Analysis

RSA on Atmel XMEGA

Differential Power Analysis
Paul Kocher, Joshua Jaffe, Benjamin Jun

- Statistical analysis of power traces
- Leak the contents of internal device bus
RSA Modular Exponentiation

```c
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while (e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
        e >>= 1;
    }
    return product;
}
```
RSA Modular Exponentiation

```c
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while ( e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
        e >>= 1;
    }
    return product;
}
```
RSA Modular Exponentiation

```c
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while (e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
        e >>= 1;
    }
    return product;
}
```
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while (e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
        e >>= 1;
    }
    return product;
}
RSA Modular Exponentiation

```c
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while (e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
        e >>= 1;
    }
    return product;
}
```
RSA Modular Exponentiation

```c
int rsa_modExp(int b, int e, int m) {
    int product = 1;
    b = b % m;
    while (e > 0) {
        if (e & 1) {
            product = modmult(product, b, m);
        }
        b = modmult(b, b, m);
    }
    e >>= 1;
    return product;
}
```

e = 0xf0
Back to the Xbox...
Have your thoughts changed?
- “SECRET ROM” STORED IN MCPX
- FLASH DECRYPTION KEY (RC4) STORED IN “SECRET ROM”
- DATA BUS “SNIFFER” BUILT BETWEEN FLASH AND MCPX

DECRIPTION KEY DISCOVERED, FLASH KERNEL (BIOS) DECRYPTED AND PATCHED FOR UNSIGNED CODE
What attackers are willing to do (Kamikaze Hack)
Xbox 360 Reset Glitch Hack (RGH)

XBOX360 Slim Reset Glitch Modification

Tutorial: RGH ANY PHAT XBOX 360

Image: Microsoft
Xbox One Security Architecture

Confidentiality

Plaintext of games and secret keys **never** leave the CPU die

Integrity

Attest software is not compromised before connecting to Xbox Live
Xbox One Security Architecture

Attacker == user

- Any bus / external device considered compromised (flash, HDD, DRAM)
- PCIe, SATA, USB, DRAM bus, motherboard fabric can be intercepted
- Can only trust CPU Si itself
Use a custom chip!

- Encrypt all busses and DRAM contents
- Custom on-die crypto registers hold keys
- Build a shared key between CPU and optical disk drive
- Reduce trusted computing base (TCB) by moving security critical code to a trusted minimal hypervisor
- Bringup uses secure Boot ROM to sign future stages
Takeaways

- Physical attacks pose a new threat model (customer may also be the attacker!)
- Cannot trust anything off-chip
- We can classify attacks based on costs and invasiveness
  - Some attacks are quite cheap...
- Defense in depth (no single point of failure), tradeoff between security and performance
  - Just need to ensure physical attacks aren't easy enough to be worth an attacker's time