OSS Japan 2018

Common attacks on IoT devices

Christina Quast

Agenda

- What is IoT? And why is security important?
- Hardware attacks
- Software attacks
- Example attack stories
- Take-aways

What is IoT?

- embedded device connected to the internet
- often power constrained, small, connected over some kind of wireless technology
- often memory-constrained
- e.g. PLC, SSD-Controller, Temperature-control
- Often easy to hack

- Can become part of botnet

[by geralt on pixabay/ CC0]

Approach

- Analysis: Inspect components, datasheets, firmware update process, contents of flash
- Code execution: Tamper with firmware update process, rewrite persistent memory content, gain access over debug channels/JTAG
- Communication channel: Get feedback from device over JTAG, serial console, etc
- Firmware exploitation:
  - Get firmware
  - Analyse it
  - Mount file system, analyze content (services provided, users configured)
  - Emulate firmware (dynamic/runtime analysis)

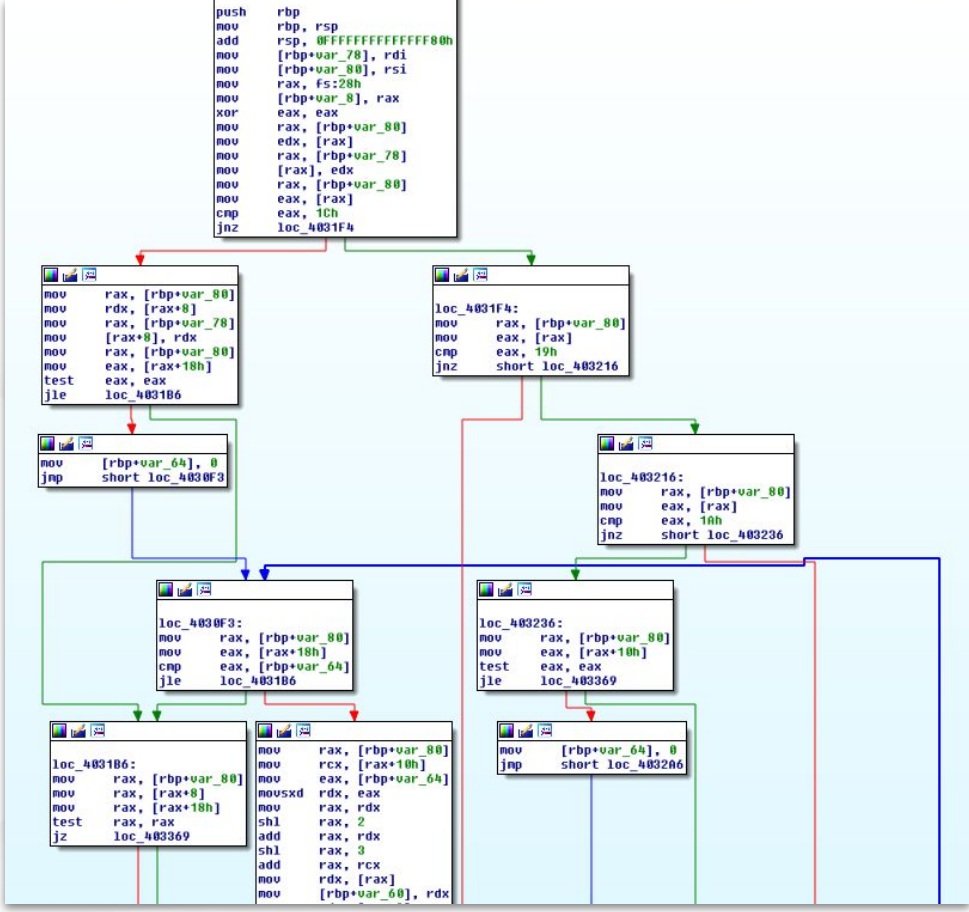
Software

Where to get the firmware

- Dump from device memory
- Download from manufacturer FTP server/search on ftp index sites
- Get from CD/DVD
- Wireshark traces of firmware updates

Analyse firmware

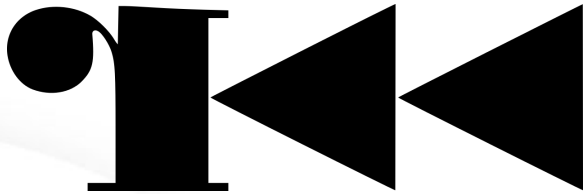
- Understand file format from firmware update routine
- Search for code/string on code.google.com, sourceforge.net, ..
- Decompile, compile, tweak, fuzz
- If not stripped and human readable strings, it's easier to reverse

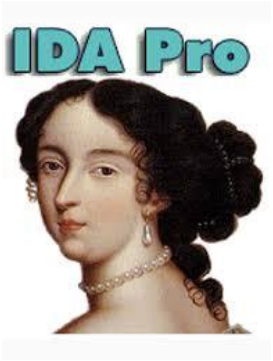


IDA view

Attacker Tools

- **Software:**
  - **Binary reversing:**
    - IDA Pro
    - radare2
    - binaryninja
  - **Bug finder**:
    - Flawfinder
    - Metasploit Framework
  - **Firmware analysis:**
    - firmwalker (with binwalk, cpu_rec)
    - firmware-analysis-toolkit
    - FACT (firmware analysis and comparison tool)
  - **Web testing:**
    - ZAP, sqlmap, sslyze, Gobuster (see OWASP)
  - **Debugging:**
    - GDB & OpenOCD

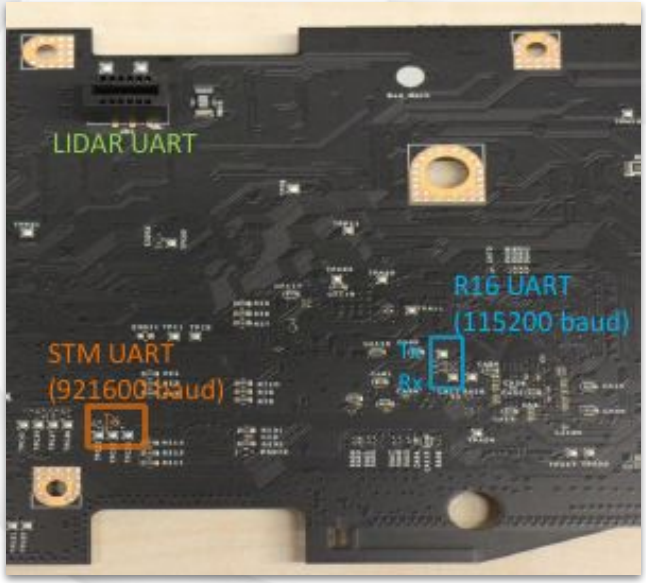




metasploit®

OWASP
Open Web Application Security Project

Hardware



[Backside layout mainboard XIaomi Vacuum Cleaner robot *by Dennis Giese and Daniel Wegemer*]

Non-invasive attacks

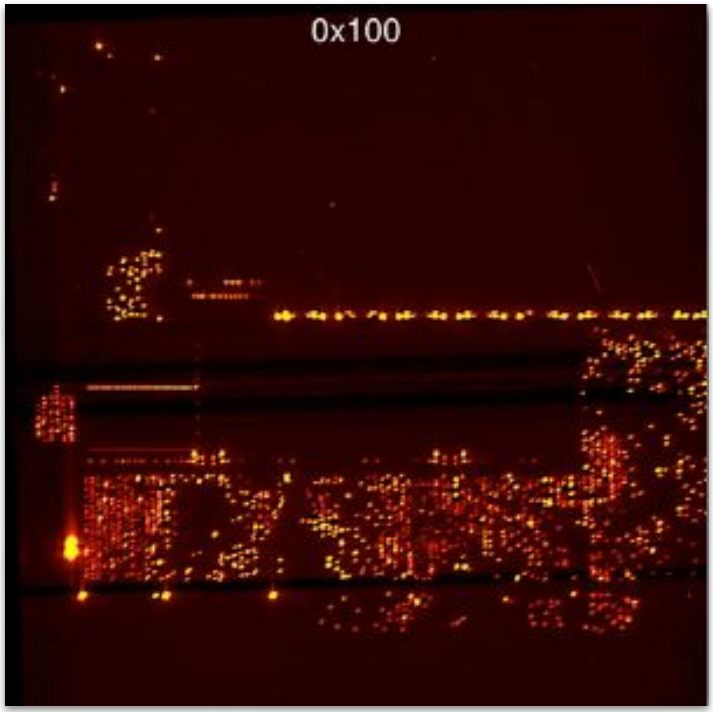
- Search for UART, JTAG, etc
- Write protection security fuses not enabled => Patch bootloader
- Hardware Fuzzing (automatically send random data and monitor whether device crashes)
- **Side channel attacks**
  - **Timing attacks**
    - Computation time depends on value of secret data
    - Cache miss and cache hit have huge timing difference => find access pattern in dependence of timing difference

* See Talk “Hardware Hacking - Extracting Information From Chips” by *Dmitry Nedospasov (@nedos)*

Hardware

Non-invasive attacks

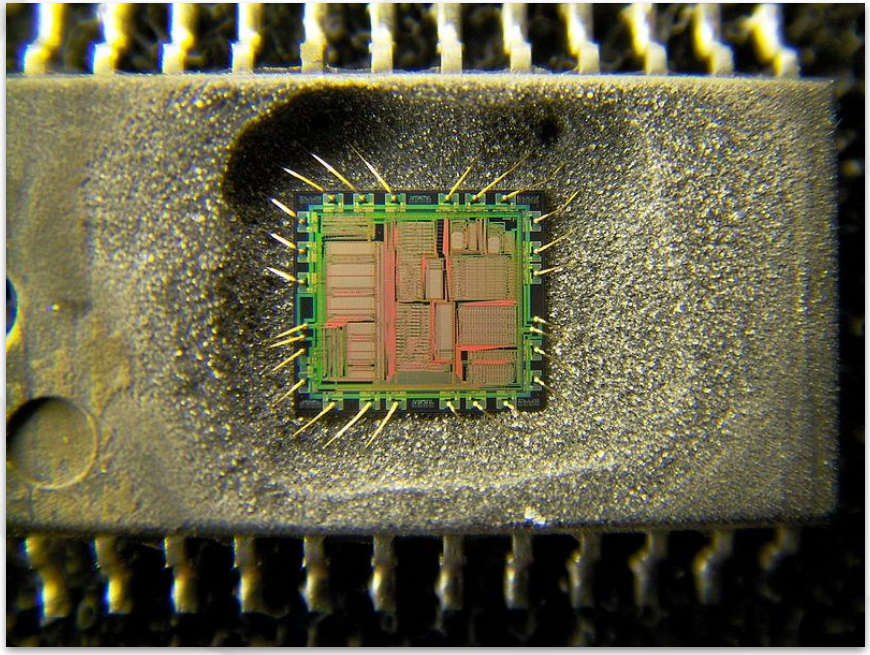
- **Side Channel Attacks (2)**
  - **Hardware Glitching**
    - very high/low voltage
    - alter clock period during execution
  - **Power Analysis**
    - Power consumption of a chip depends on the secret data that is computed on the chip):
      - SPA (Simple power analysis)
      - DPA (Differential power analysis)
    - EM Radiation channel
    - Acoustic channel
    - Photonic emission side channel



[Visible and infrared light emitted by switching transistors/ by *Dmitry Nedospasov*]

* See Talk “Hardware Hacking - Extracting Information From Chips” by *Dmitry Nedospasov (@nedos)*

Hardware



[Yamaha audio IC decapsulated by Olli Niemitalo/ CC0 1.0]

Semi-invasive attacks

- Decapping package
- Infrared light/photon emission analysis of backside to find location for attack
- Then use laser to flip bits and break crypto

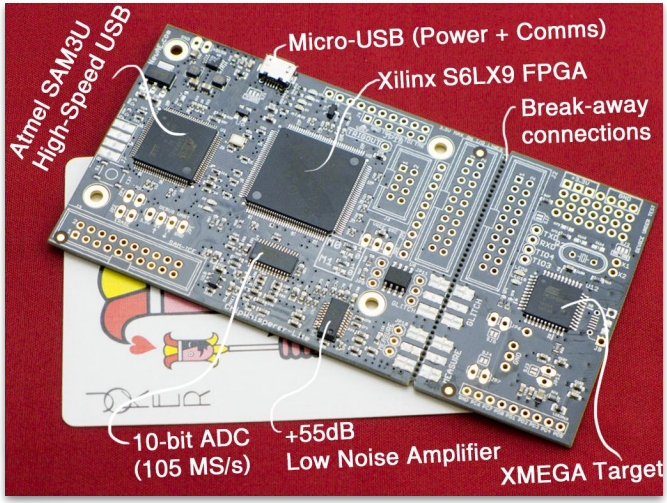
Fully-invasive attacks

- Much effort, but 100% success rate
- Modify chip with FIB (Focused Ion Beam)
- Microprobing
- Linear code extraction (LCE)

* See Talk “Hardware Hacking - Extracting Information From Chips” by *Dmitry Nedospasov (@nedos)*

Attacker Tools

- **Hardware**:
  - Oscilloscope
  - Logic Analyzer (e.g. Salae)
  - JTAG:
    - GoodFET, BusBlaster, BusPirate, JTAGulator, JTAGenum, Black Magic Probe
  - Side Channel Attacks:
    - ChipWhisperer (power analysis, glitching attacks)
  - USB:
    - Facedancer
  - SDR:
    - HackRF
  - Datenkrake (FPGA-base



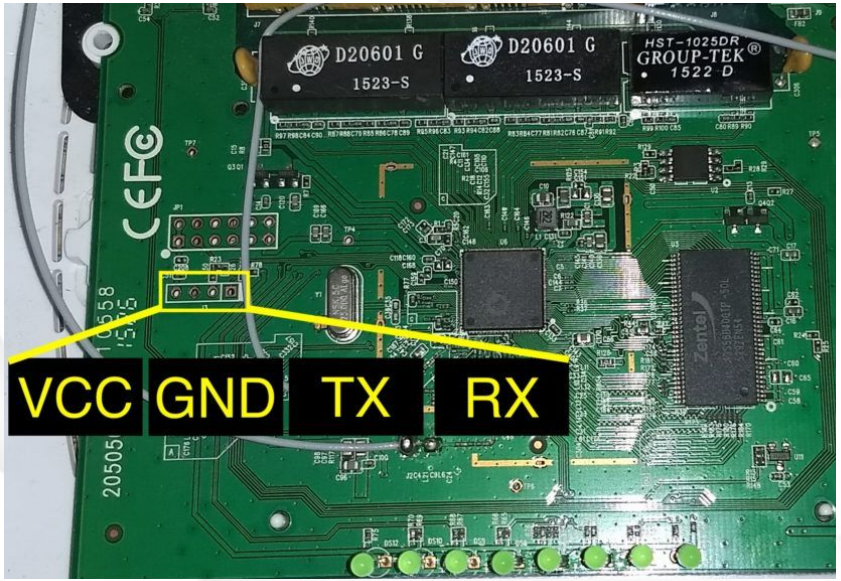
ChipWhisperer

BusPirate

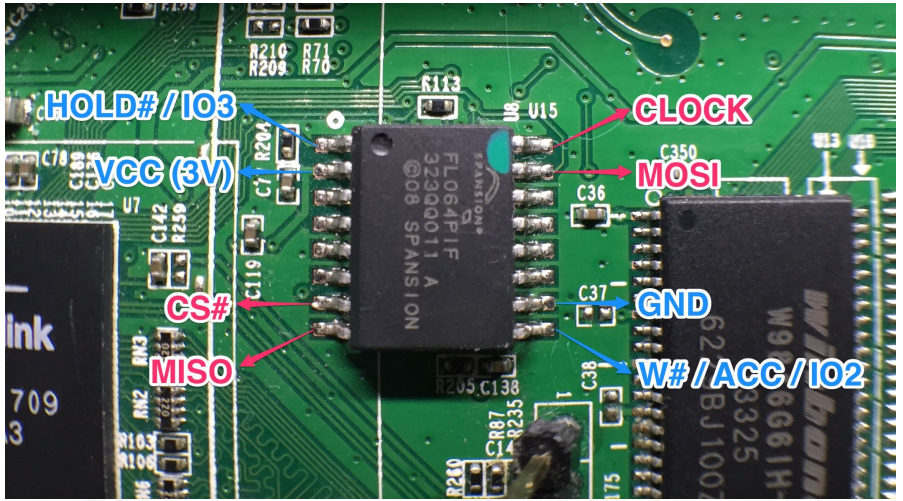
HackRF

Real attack stories*

- **UART** (populated or not): Usually device boots into special console/root console



[From 5-Min Tutorial: Gaining Root via UART *by @konukoli*]

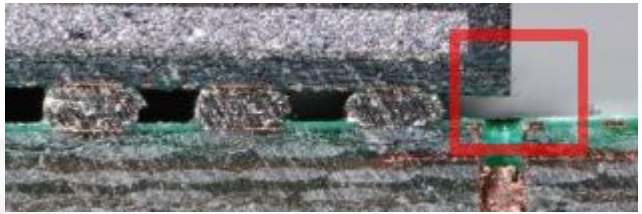


[From Hack The World *by Juan Carlos Jiménez*]

* See Talk “Hack All The Things: 20 Devices in 45 Minutes” by gtvhacker

Real attack stories*

- **Root with U-Boot**:
  - Access bootloader shell, add **init=/bin/sh** into kernel cmdline
  - Will execute **preconfigured script name** ‘xyz’ => replace script with own script
  - **Short pins** on NAND, power on => boot into corrupted U-Boot environment
- **Hardcoded/base64 encoded username** and **password** in binary
- **Bruteforce** easy password



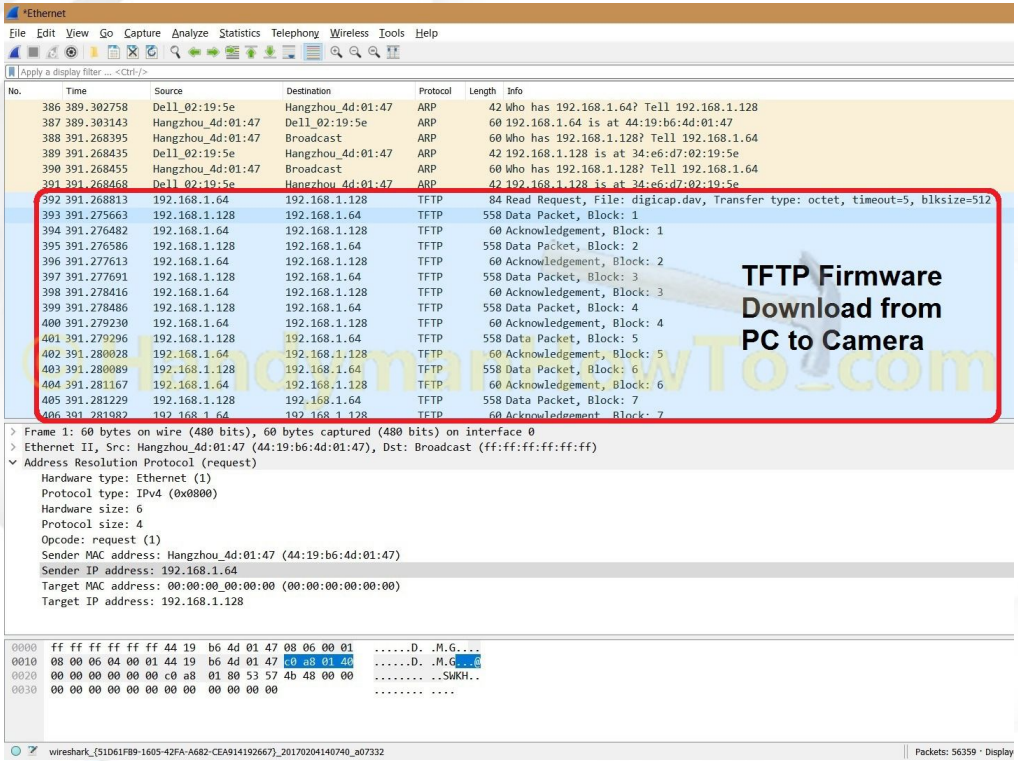
[BGA by *Smial / GFDL-1.2]*

```
la      $t9, _strcpy
lui     $s1, 0x5A
move    $a0, $s0
jalr    $t9 ; _strcpy
addiu   $a1, $s1, (aAdmin - 0x5A0000)  # "admin"
lw      $gp, 0x60+var_50($sp)
addiu   $a1, $s1, (aAdmin - 0x5A0000)  # "admin"
la      $t9, _strcat
nop
jalr    $t9 ; _strcat
move    $a0, $s0
lw      $gp, 0x60+var_50($sp)
nop
la      $t9, _strlen
nop
jalr    $t9 ; _strlen
move    $a0, $s0
move    $a2, $v0
addiu   $a0, $sp, 0x60+var_48
jal     md5checksum
move    $a1, $s0
lw      $v0, 0x60+var_44($sp)
lui     $a0, 0x5E
addiu   $v1, $a0, (dword_5E3E00 - 0x5E0000)
sw      $v0, (byte_5E3E04 - 0x5E3E00)($v1)
li      $v0, 1
sw      $v0, (dword_60D9C8 - 0x610000)($s2)
lw      $v0, 0x60+var_48($sp)
lw      $gp, 0x60+var_50($sp)
sw      $v0, dword_5E3E00
```

[Reverse Engineering the TP-Link HS110 *by Lubomir Stroetmann, Consultant and Tobias Esser, Consultant/ © Softscheck]*

* See Talk “Hack All The Things: 20 Devices in 45 Minutes” by *gtvhacker*

Real attack stories*



[How to Fix a Bricked Hikvision IP Camera Firmware *by Bob Jackson*]

- Write su binary into eMMC fs
- **Command injection**
  - system("ls %s"): will reboot on user input "; reboot;")
  - often in WEP or Wifi password field of Configuration Web page, Network folder names, ..
  - In URL parameters (http://foobar/subpage?action=command&command=reboot)
- App installation /Firmware update over **unencrypted HTTP/FTP** => can be intercepted
- SMB share without restrictions, run su binary via adb

* See Talk "Hack All The Things: 20 Devices in 45 Minutes" by *gtvhacker*

Real attack stories: Xiaomi Vacuum Cleaning Robot*

Micro USB Port: was authentication protected X

Serial communication: Didn't find X

Port Scan: No suspicious open ports X

Sniff network traffic X

Recovery mode: Shorting BGA pins with aluminium foil

[CC by 4.0 34C3 media.ccc.de}

* See Talk "34C3 - Unleash your smart-home devices: Vacuum Cleaning Robot Hacking" by *Dennis Giese, Daniel Wegemer from TU Darmstadt*

Real attack stories: PLC*

- Downgrading to older firmware
- Physical mapping of JTAG not easy to find
- Injecting code into firmware update

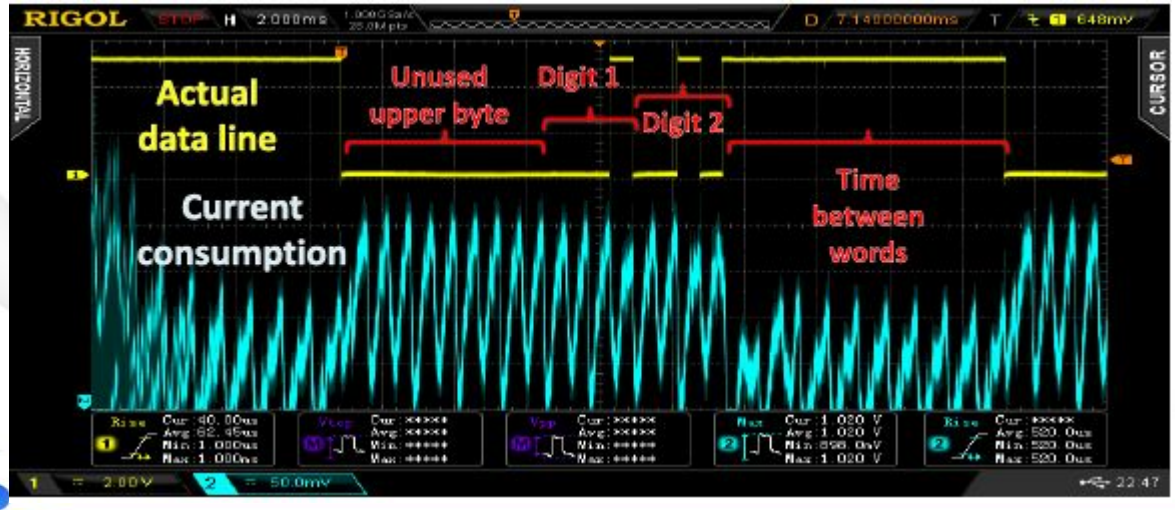
- Injecting code via flash reprogramming
  - rewrote bootloader after partly desoldering pins asserting write protection
  - MitM like setup for quick prototyping and testing of bootloader replacement code

Picture: https://www.astiautomation.ro/en/product/plc-canopen-training-panel-s7-1200-siemens/

* See Paper “Off-the-shelf Embedded Devices as Platforms for Security Research” by *L.Cojocar, K.Razavi, H.Bos* (see References)

Real attack stories: Electronic Safe Lock*

- Resistor in series to battery and lock
- Amplified current => Power analysis Side channel attack (high current consumption => 0 read from EEPROM, low current => 1 read from EEPROM
- *Mitigate*: Don't store secret in EEPROM



Sargent & Greenleaf 6120-332
[by *Plore]*

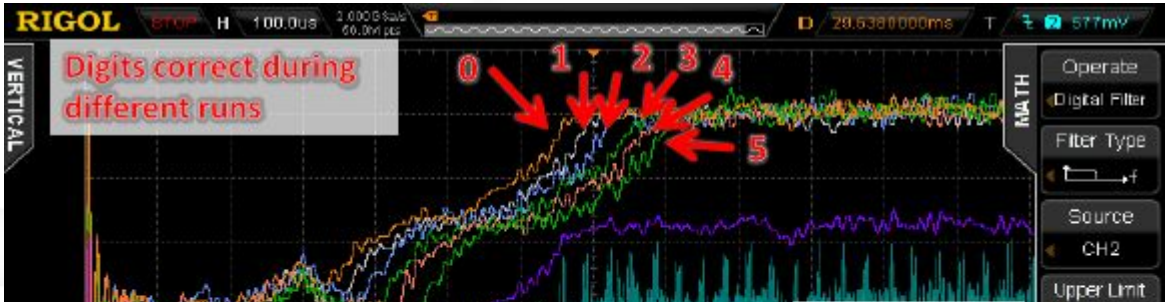
[by *Plore]*

* See Talk "DEF CON 24 - Plore - Side channel attacks on high security electronic safe locks" by *Plore*

Real attack stories: Electronic Safe Lock*

- Timing attack: The correct key will have a longer delay
- Problem: 5 tries, then locked out for 10 minutes
- Counter of tries stored in EEPROM
- Reset counter by turning off MCU shortly after write of counter started, where cell is erased but not written yet
- Mitigate: Constant time for comparison, hashed secrets

S&G Titan PivotBolt [by *Plore]*



[by *Plore]*

* See Talk “DEF CON 24 - Plore - Side channel attacks on high security electronic safe locks” by *Plore*

Protection*

- **Buffer/Stack Overflow Protection, heap overflow protection**
  - Use **safe equivalent** functions (gets()->fgets())
  - Verify buffer bounds
  - Secure compiler flags (-fPIE, -fstack-protector-all, -Wl,-z,noecexstack, -Wl,-z,noexecheap,..)
  - See https://wiki.debian.org/Hardening#Using_Hardening_Options

- **Injection (SQL/command injection, XSS) protection for webservers**
  - Whitelist commands
  - No user data into OS system commands
  - Validate input & output

* See Talk “AppSec EU 2017 Don't Get Caught Em-bed” by Aaron Guzman

Protection*

- **Firmware Updates with cryptographic signatures, update over TLS**
  - Force updates for high critical bugs
  - Anti-rollback protection
  - Infrastructure with pub-priv key for verifying signed packages
  - **Don't Roll Your Own Crypto!**

- **Secure sensitive information**
  - **No hardcoded** secrets (usernames, passwords, tokens, priv keys,.).
  - Store secrets only in protected storage (NOT EEPROM, flash)
  - Use Trusted Execution Environment (TEE) or security element (SE), TrustZone (for ARM)

* See Talk “AppSec EU 2017 Don't Get Caught Em-bed” by Aaron Guzman

Protection*

- **Identity Management**
  - Separate accounts for internal/remote web management, internal/remote console access
  - No sessionIDs/Tokens/Cookies in URL (can be replayed)
  - Tokens should be randomized, and invalidated on logout
  - Secure and complex password for accessing UART, EEPROM, ssh
  - Each device: individual secret (one device's gets hacked, the others stay safe)

- **Hardened toolchains, libraries and frameworks**
  - **Remove unused** language/shell interpreters (/bin/dash, /bin/bash, /bin/ash, /bin/zsh, ..), dead (debugging) code (dead code which can be used for attacks), unused libs
  - **Disable ancient legacy** protocols (ftp, telnet, ..)
  - Remove debugging interfaces
  - Remove (or secure) ~~backdoors~~ management interfaces for consumer support/debugging purposes,...usually with root privilege
  - Check third party code and SDKs

* See Talk "AppSec EU 2017 Don't Get Caught Em-bed" by Aaron Guzman

Protection*

- **Keep kernel, frameworks & libraries up to date**
  - Use package managers opkg, ipkg
  - Check against vulnerabilities DBs
  - Load tools to check third party code and components (retirejs, libscanner, nsp, lynis, owasp zap, ..),

- **Threat modeling**

* See Talk “AppSec EU 2017 Don't Get Caught Em-bed” by Aaron Guzman

Take-aways

- Main attack vectors: web-interface, crypto, outdated/unpatched firmware, sniffing unencrypted communication and cleartext passwords..
- Don’t have your key or password fixed in your binary, store secrets in hardware protected place
- Integrate security tests into your CI/development cycles

- There is always a way to hack a system, just a matter of cost and time

Questions?

Resources

- https://www.owasp.org/index.php/OWASP_Embedded_Application_Security
- http://www.sharcs-project.eu/m/documents/papers/a01-cojocar.p df (Off-the-shelf Embedded Devices as Platforms for Security Research)
- https://www.handymanhowto.com/how-to-fix-a-bricked-hikvision-ip-camera-firmware/
- http://jcjc-dev.com/2016/06/08/reversing-huawei-4-dumping-flash/
- http://konukoii.com/blog/2018/02/16/5-min-tutorial-root-via-uart/

Recommended Talks

- “34C3 - Unleash your smart-home devices: Vacuum Cleaning Robot Hacking” by Dennis Giese, Daniel Wegemer from TU Darmstadt
- “Hardware Hacking - Extracting Information From Chips“ by Dmitry Nedospasov
- “Lockpicking in the IoT...or why adding BTLE to a device sometimes isn't smart at all“ by Ray
- “DEF CON 24 - Plore - Side channel attacks on high security electronic safe locks” by Plore
- Hack All The Things: 20 Devices in 45 Minutes
- “Black Hat 2013 - Exploiting Network Surveillance Cameras Like a Hollywood Hacker” by Craig Heffner