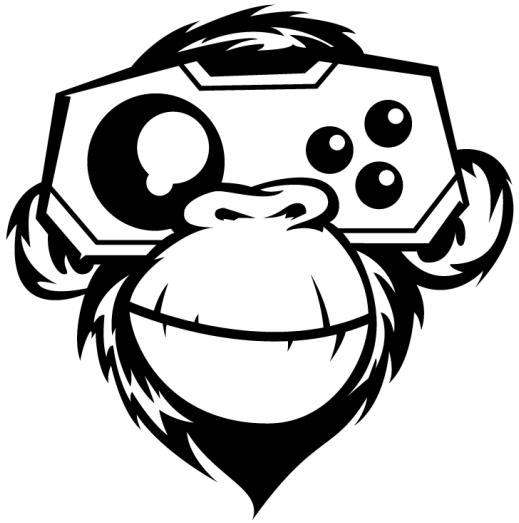


RHESUS
ENGINEERING

Watch your Software

Why Trustzone™ is great, but it’s not the end of the story

About: Rhesus Engineering

- Small engineering bureau located in Zürich, Switzerland

- Strong focus on:
  - Embedded Linux / Android Platform Porting
  - Linux Driver Development
  - Embedded Software / ECos, FreeRTOS, etc.
  - Bare Metal Software

- Background in SOC technologies (Intel, Xilinx, NXP)

- More info: info@rhesus.ch or https://rhesus.ch

Trustzone! Secure-Boot! Encrypted Bitstream! What could go wrong!

- Trustzone is pretty secure (combined with signed loaders, kernels and binaries)

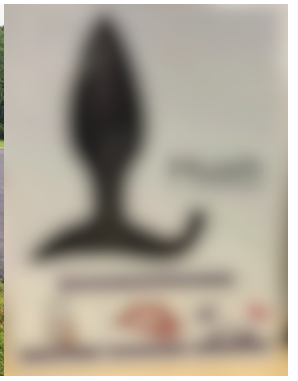
- Until a simple bug – five layers above – destroys it all

What can go wrong? Theft, DDOS, Proxies, Crypto-Coins

- IP-Theft
- Bandwidth
- Disguise
- Calculation-Power
- Piracy

Breaking: Gazillions of IOT-Blenders hacked!!!!!

- Not so “fake” headline as you would think at first
  - Refrigerators sending spam
  - People turning on and off lights in random homes
  - Access to CAN bus via GSM network
  - Taking over adult toys via Bluetooth
  - List goes on…



Example 1: High level fails

- Some large home-router company allowed admin access without a password
- All the browser had to do was to send a special user-agent string
- That backdoor was intentionally placed there

- Same company also included their private signing keys in a firmware update binary at some other point in time

Example 2: The Sony PS3 hack

- Strong Hypervisor for separating stuff and allowing Guest-OSs
- All guest memory accesses managed by the MMU via HTAB Entries
- Someone glitched the Address-Bus.

- MMU tricked to not release access to a memory location and hypervisor tricked to place a new HTAB entry into it
- HTAB entry changed by guest OS: full access to physical memory
- Some memory Read/Write primitives installed

- In the end: all level0 firmware encryptions keys where recovered (due to weak crypto) and leaked

Example 3: The Nintendo Switch hack

- Nintendo did good on HW level (full program: encryption, signing, etc.)
- But: they added a web-rendering engine for registering to WIFI networks (webkit)
- Attackers spoofed DNS replies and sent an evil “registration page” to exploit webkit.
- From there the attackers started to explore the system

- Today there is an exploit on the Tegra X1 **bootrom** level (USB rescue mode to load unsigned binaries)

Example 4: The Sony PS4 Hack

- Sony uses signing and encryption
- In the beginning there was a web browser

- Over time the attackers:
  - Created a file browser in the web browser
  - Dumped on older version kernel via PCIe (and found symmetrical keys & ELF information)
  - Found out that crash dumps are encrypted with the symmetrical keys
  - Found FreeBSD Kernel vulnerabilities
  - Etc. Etc.

- The PS4 runs Linux now

Fail 1: Web Browser / Server

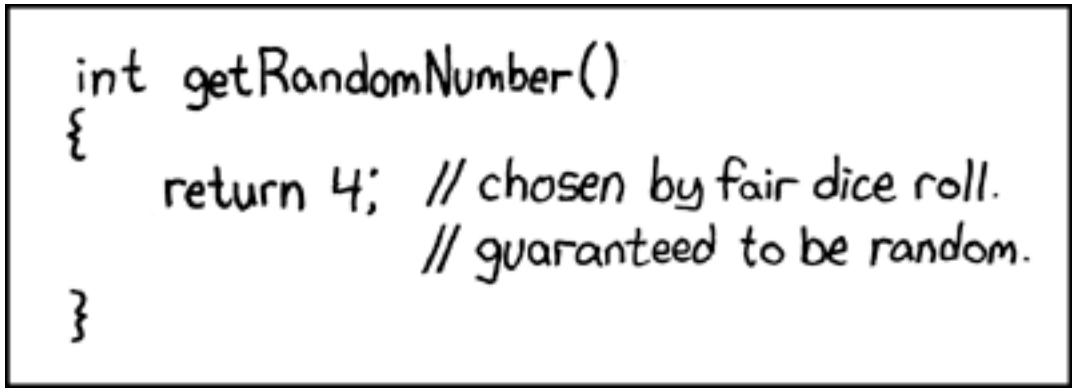
- Do not include a web browser into your product unless you absolutely have to!
  - If you have to: get professional help.
  - Javascript is pure evil (sorry… all you full-stack developers)

- Webservers are a target as well
  - XSS – sometimes even on “but we are only reading values” servers

Fail 2: Standard Passwords

- No development backdoors!
  - They will make it into the release build at some point
  - If interactive access is really needed: Add strong per-device passwords and transport encryption
    - No: the Mac Address is not a good choice for generating the passwords

Fail 3: You should: Get your crypto right



- Don’t roll your own
  - Crypto is hard
  - If you don’t understand it fully: you will do it wrong.

Fail 4: You should: Keep your keys secure

- Do not use the same key for every product
- Manage your keys

- Just keep them in a safe. They are called “keys” for a reason.

Conclusions

- Don’t under estimate the software layer
- Don’t feel safe just because you have a strong hardware security concept
- Adding security on hardware level is complex

- Deploy a defined and clear release process (over all involved departments)

Or simply put:

- Trustzone is a very strong tool to protect your hardware and software
- Most fails happen in software

- You should have a development process from the PCB up to the very last software-update

Questions?