Hardware Security

With an Emphasis on
Supply Chain Attacks & Verifiability

bunnie (@bunniestudios / twitter)

MIT 6.858 May 2022
Topics

• Breaking Hardware Security
  • Direct physical tampering
  • Indirect supply chain tampering

• Mitigations
  • vs. supply chain attacks: User-verifiable hardware
  • vs. direct attacks: plausible deniability
  • vs. direct attacks: (Not in covered, mentioned for completeness) tamper-evident / tamper-resistant and anti-cloning techniques
Protecting Secrets within a "Vault": Hardware Security Modules (HSMs)
Direct Attacks on Hardware: Overview

- Passive – little to no modification of target system
  - Direct observation
    - Optical
    - SEM
  - Side-channel (emissions)
    - Power
    - RF
    - Optical

- Active – no holds barred
  - Fault induction
    - Glitching (clock/VDD)
    - Coupling (e.g. row hammer)
    - Photonic
  - FIB edit
Passive: Direct Observation
Passive: Direct Measurement
Passive: Optical Emissions

Passive: Power Side-Channels

Fig. 1. A sample power trace of Spartan-6 (with 20MHz low-pass filter) during loading an encrypted bitstream

Moradi, A and Schneider, T. "Improved Side-Channel Analysis Attacks on Xilinx Bitstream Encryption of 5, 6, and 7 Series"
Passive: RF Side-Channels

**Fig. 5.** EM probes and different FPGAs, (a) XC5VLX50-1FFG324, (b) XC6SLX75-2CSG484C, (c) XC7K160T-1FBGC, (d) XC7A35T-1CPG236C

Moradi, A and Schneider, T. “Improved Side-Channel Analysis Attacks on Xilinx Bitstream Encryption of 5, 6, and 7 Series”
Active: FIB

From http://www.electronicdesign.com/eda/fib-circuit-edit-becomes-increasingly-valuable-advanced-node-design
Active: Fault Injection (Glitching, Optical)

Also available in remote attacks: See https://plundervolt.com/doc/plundervolt.pdf
Fault Injection: The General Idea

1. Compute Credentials
2. Valid?
   - Y: Do Secure Thing
   - N: Abort
Glitching To Run an Alternate Code Base

1. Compute Credentials
2. Validate Credentials
   - Y: Do Secure Thing
   - N: Something Else
3. Something Else
4. Abort
Glitching to Change a Branch

1. Compute Credentials
2. Valid?
   - Y: Do Secure Thing
   - N: Abort
3. Something Else
Glitching To Cause Cipher Faults That Leak Private Data

* Some ciphers (e.g. RSA) leak secrets if the computation is glitched
Fault Injection

- Can be surprisingly trivial to execute ("twiizer" attack") ---->
Active: Coupling (e.g. Rowhammering)

From https://www.raith.com/products/chipscanner.html
Active: Microarchitectural Side Channels

- Leverage timing differences in latency hiding features to leak secrets

DOI: 10.1109/ACCESS.2020.2988370
Can't Access the Hardware?

Attacks Prior to Installation: Supply Chain Tampering
"State of the Practice" for Trusting Chips: Reading the Label on the Box
Not Just Chips: Whole Assemblies Are Swapped Without Detection

NSA: Implanting beacons in CISCO routers


JTAG implants Dell PowerEdge servers
Andy Müller-Maguhn – listening device in cryptophone

An Ontology of Supply Chain Attacks
Degrees of Detection Difficulty

Detection

Visual/JTAG

X-Ray

SEM

Execution

Easy

Hard

IC mod

Hard IP edit

Substitute IC in package

Mask edit

Substitute component

component

device

PCB

C in package
Degrees of Execution Difficulty

If custom ICs are involved

$1\text{mm+}, \text{months}$

$10, \text{weeks}$

$<1, \text{seconds}$
"Substitute Component"

- Relies on the fact that many components look alike
"Add A Component"

- Easily detectable -> higher awareness
"Add IC in Package"

- Hide an additional chip inside a package
- Multiple chips in package is a mature technology
Solution: X-Ray All the Things?
Obvious

Less obvious
Problem #1

- Silicon (Z=14) is relatively transparent to X-rays
  - Copper traces, solder tend to mask the presence of silicon
- Mitigations
  - CT (Computerized Tomography) scanners
  - X-ray diffraction, spectroscopy
Problem #2: X-Rays Don't Trivially Detect Multiple ICs

Top view: looks like straight wires

Side view: visible, but requires unobstructed line of sight

IC Modifications

Detection

Execution

Easy

PCB

Add component

Substitute component

Hard

Add IC in package

IC mod

Hard IP edit

Mask edit

Netlist edit

Substitute IC in package

Hard in package

Easy
IC Fab: Attack Surfaces

- Netlist Tampering
  - RTL = Verilog, VHDL, Python
- Hard IP Tampering
- Mask Tampering
Netlist Tampering: ASIC vs COT

- **ASIC** – “Application Specific Integrated Circuit”
  - Customer does RTL + floorplan
  - Foundry does detail place/route, IP integration, pad ring
  - Popular for e.g. cheap support chips:
    - Server BMC (Baseboard Management Controller)
    - Disk controllers
    - Mid-to-low end I/O controllers

- **COT** – “Customer Owned Tooling”
  - Customer does full flow, down to a nominal GDS-II mask
  - Several extra headcount + $millions for back-end tooling software
  - Necessary for high-performance / flagship products (CPU/GPU/router)
ASIC Design Flow Example: SOCIONEXT

- One of many billion-dollar ASIC companies you've never heard of
So I'm Safe with COT, Right?
COT Weaknesses: "Hard IP"

- COT designers still leave large "holes" in the layout for hard IP
  - Foundry merges proprietary blocks with agreed upon connection points
COT Weaknesses: "Hard IP"

- COT designers still leave large “holes” in the layout for hard IP
  - Foundry merges proprietary blocks with agreed upon connection points

https://cornell-ece5745.github.io/ece5745-tut8-sram/
Hard IP: Who Cares?

- RF/analog
  - PLL, ADC, DAC, bandgap
- RAM
- ROM
- eFuse
- Pad rings

- Basically, all the points you need to backdoor an IC
Mask Tampering: Post-Design Processing

- Sub-wavelength features requires substantial mask post-processing

Mask Editing

- All masks go through an editing ("checking") step
What Can you Do with Mask Editing?

- Example: Dopant Tampering
  - No morphological change
  - Circuit-level behavioral change
- Spare cell rewiring
- Signal bypass

My Personal Fear: TSV + WLCSP Implants

Unmodified

With TSV implant
Concept: WLCSP

- Sold as "almost naked silicon"
- Direct chip-to-board solderballs
- Sold as "Ready to Hack"

Wafer
Level
Chip
Scale
Package
Concept: Through-Silicon Via "Mature" Tech (Used in HBM RAM)

https://www.youtube.com/watch?v=20t4FCH3K60
TSV + WLCSP = Nearly Undetectable Implant
Threat & Mitigation

- Scalable
- Targets off-the-shelf chips
- No decap / debond
- Hard to detect
  - Many WLCSP already have a small seam
  - No X-ray footprint
- Mitigation:
  - TSV templates are "expensive" ($100k's)
  - But Pegasus is even more expensive ($1mm+)...
Execution of Supply Chain Attacks: The Attack Surface
you
We're Not Going to Talk about "Evil Maids" (But They are Also Real)
(TS//SI//NF) Left: Intercepted packages are opened carefully; Right: A “load station” implants a beacon

Everyday Hacks for Everyday People Targets

- DIY supply chain attack:
  - Buy item online
  - Hack it
  - Return it to warehouse
  - ???
  - Profit!
It's a Big Attack Surface
What Can We Do About It?
Can Open Source Save Us?
Problem: Place of Check Too far from Place of Use
Open Factory Test (Trustable Factory) Is Only a Marginal Improvement
What, Then Is the Role of Open Source in Trustable Hardware?

- Design Correctness
  - Peer review can find bugs
  - SPECTRE hardening
    - Microarchitectural state modeling in compilers
    - Potential for provably correct compiler mitigations
The Big Problem: You Can't "Hash" Hardware

- There is no convenient, easy-to-use method to confirm the correctness of hardware immediately before its use.

- Hardware is one big "Time of Check versus Time of Use" (TOCTOU) problem!
But You Once Said: "There's Always a Bigger Microscope..."

- "Ptychographic X-Ray Imaging" to the rescue?
  - Non-destructive
  - 3D imaging of complex chips
  - Great for reverse engineering and design verification

https://www.nature.com/articles/nature21698
Problem #1: A Building-Sized Microscope

https://www.psi.ch/en/sls/about-sls
Problem #2: Verifying One Chip Verifies Only One Chip

- Just because 99.9% of your hardware is OK...
  - Doesn't mean you are safe
  - One compromised server out of thousands is all it takes
- Random sampling is not effective
  - Would you "random sample" signature checks on downloaded software?
Can We Build an Evidence-Based Case To Trust Our Computers?
Three Principles For Evidence-Based Trust in Hardware

1) Complexity is the enemy of verification

2) Verify entire systems, not just components

3) Empower end-users to verify and seal their hardware
Problem: Complexity is Complicated

- Absent a robust "hashing" function, verification falls back to bit-by-bit...or "atom-by-atom"
- More complexity ->
  - More difficult to verify
  - More places to hide things
  - Verification might be destructive

via iFixit
Point of Use Verification Tradeoff: Ease of Verification vs. Features & Usability

Features & Usability

Ease of verification

>10^7 transistors

1 transistor

Maurizio Pesce CC BY 2.0
Three Principles For Evidence-Based Trust in Hardware

1) Complexity is the enemy of verification

2) Verify entire systems, not just components

3) Empower end-users to verify and seal their hardware
Why a Device, and Not a Chip?

- Private keys are not your private matters
- Screens can be scraped, keyboards can be logged
The "IME Problem"

Your Phone

Security Enclave

Encryption

Plaintext

CPU

Plaintext

This input method may be able to collect all the text that you type, including personal data like passwords and credit card numbers. It comes from the app Keepass2Android. Use this input method?

CANCEL  OK
The "IME Problem"

Your Phone

Note

This input method may be able to collect all the text that you type, including personal data like passwords and credit card numbers. It comes from the app Keepass2Android. Use this input method?

CANCEL  OK
Three Principles For Evidence-Based Trust in Hardware

1) Complexity is the enemy of verification

2) Verify entire systems, not just components

3) Empower end-users to verify and seal their hardware
Empower Verification At Multiple Levels
Precursor: A Case Study in Verifiable Hardware

- Designed to facilitate evidence-based trust
  - Simple in construction
  - Open in design
  - Sufficient in function
Getting HCI Right is A Major Issue in Security

- HCI = Human Computer Interface
- Humans are increasingly the "weakest leak"

- Simple, inflexible interface
  - Minimal attack surface
- “Just enough and no more”
  - Securable attack surface
- Featureful, flexible interface
  - Intractable attack surface
Precursor: What Functions?

- Designed for mostly single-app deployments of:
  - Secure text messaging
  - Voice chat
  - Multi-lingual capability
  - Password management
  - Crypto wallet
- Not designed for
  - Web browsing
  - Games
  - Photos and videos
- Specs:
  - 100MHz RV32IMAC + MMU + AES extensions
  - Curve25519 + SHA2 accel
  - 16MiB RAM
  - 536x336 "memory" LCD
  - USB + Wifi connectivity
  - Audio only via jack
  - Full-custom OS "Xous"
    - QNX-like microkernel, written in Rust
Precursor: Simple in Construction
Simple to Inspect
Physical Keyboard

- Wires visually inspectable
- 2-layer daughtercard:
  - Bright light may be employed to rule out buried traces
- *No silicon chips*
- User replaceable keyboard overlay for multi-lingual support
Verification Difficulty: Trivial
Touch Keyboard Verification: Very Hard

- Captouch screens require the use of a proprietary microcontroller with a firmware blob.
Verifiable LCD

- High-DPI black and white screen
  - 200 dpi
  - 336x536 pixels
Verifiable Screen

- All drive electronics on-glass
  - Inspectable with a cheap optical microscope (50x zoom shown)
- All circuits verifiable through non-destructive inspection
- No chips to verify
  - Less places to hide things => less need to check things
Why Not a Color LCD?

- Virtually all LCDs incorporate a driver IC
  - Contains a framebuffer and a command interface
The PCB: Designed Along Attack Surfaces
T-Domain Attack Surfaces Illustrated
The Hardest Problem: Evidence-Based Trust and the CPU (or SoC)

- Silicon inspection is typically destructive and hard
- Difficult to check and use a specific chip

Non-Destructive Silicon Verification???

- **Proposal:** use optical fault induction
  - **Pros:**
    - Non-destructive
    - Optical methods are relatively cheap
  - **Cons:**
    - Lower bound on trojan circuit complexity
      - RTL-level design methods can make small trojans difficult
    - Probably requires chip thinning for effective back-side illumination
      - Top metal scatters light too much
  - **Years to develop**

Laser spot size >> single transistor
Use sub-$\lambda$ scan overlap + BIST syndrome readout to correlate with expected silicon pattern
A Solution: The FPGA

- FPGAs are "Field Programmable Gate Arrays"
- Consist of large arrays of logic + wires that are user-configured to implement hardware designs
FPGA: Narrowing the TOCTOU Gap by Compiling Your Own SoC

- Anyone can compile their design from source
- Enables trust transfer via signatures "like software"!

- Subtlety: toolchain openness
  - Symbiflow is the F/OSS flow
    - Lattice ICE40 and ECP5 is 100% open flow
    - 7-Series FPGA is "coming soon" but currently requires closed vendor tools
FPGA Features "ASLR for Hardware": Pseudo-Random Mapping of Design to Device
FPGA's Biggest Potential Advantage: Moves Point-of-Check Towards the End User

- One can imagine a bitstream checker
  - Correlate design-to-bitstream

- Vision: a "one-click" tool to verify the FPGA bitstream!
  - Point of check = Point of use

“From Boot to Root in One Hour”
https://www.bunniestudios.com/blog/?p=6336
What About Direct Attacks Against Users?

- Strong security makes humans the weakest link
  - Lawful (and unlawful) coercion of secrets through search, seizure, subpoena
- Philosophical debate:
  - Should security prioritize the user's safety, or the secret's safety?
In Practice, Security Is a Function of Social Context

- Doors remain locked not because locks are effective, but because of social context
- Alternatively: police rarely have to pick locks
Lesson Learned Since 2016: Under Investigation? Plausible Deniability is Powerful!

Everything Attorney General Jeff Sessions Has Forgotten Under Oath

Over the course of four recent congressional hearings, Attorney General Jeff Sessions has somehow forgotten dozens of people, places, and events. Here's all of them in one place.

Week 66: Scott Pruitt's Selective Memory

Pruitt can't recall his misdeeds, science is out at the EPA, and Rick Perry wants to declare a national emergency to keep coal plants open.

April 27, 2018  Brian Palmer

Welcome to our weekly Trump v. Earth column, in which onEarth reviews the environment-related shenanigans of President Trump and his allies.

Trump claims ignorance of ‘burner phones’. Here’s how they work

Disposable phones may appeal to anyone trying to hide their identity - whether a criminal or an activist.
Effective Plausible Deniability

- Requirement: An omniscient adversary cannot prove or disprove that a secret exists
  - With a full forensic image of a device:
    - Encrypted data is indifferentiable from empty space *(free space wipe)*
    - No metadata leakage *(veracrypt, truecrypt in certain modes)*
      - No mysterious partitions
      - No "missing" free space on device
    - No application leaks of pointers to encrypted data *(PDDDB, [1])*
      - No password-specific salts, usernames
      - No dangling file references
      - No record in browser history, application history

The Plausibly Deniable DataBase (PDDB)

- A **(key, value)** store
- **(k,v)** pairs stored in a **Dictionary**
- **Dictionaries** stored in a **Basis**
- **User View** of the database is the union of one or more **Bases**
Mitigating API Deniability Leakage

- Locked (unmounted) **Bases** are automatically hidden in the **User View**
- Minimal application guidelines for successful plausible deniability
  - Basically: don't cache state
Mitigating Forensic Disclosure

Both Basis A and Basis B Unlocked

Cipher Requirement: IND$-CPA \[1\]
("indistinguishable from uniform randomness by a chosen-plaintext attacker")

Only Basis A Unlocked

Details: Making It Run Fast

Virtual Memory-Mapped Storage Spaces (64-bit address space)

Page Table 128-bit Entries

Physical Storage Organized in 4k Pages

Page Table Entry Format

<table>
<thead>
<tr>
<th>Virtual Page Number</th>
<th>Flags</th>
<th>Nonce</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>56 bits</td>
<td>8 bits</td>
<td>32 bits</td>
<td>32 bits</td>
</tr>
</tbody>
</table>

Basis A

Dictionary: Contacts
Key | Value
---|---
Alice | 320 Memorial Dr
Bob | 3 Ames St

Dictionary: Passwords
Key | Value
---|---
Athena | 9hjq5QshA4a3W
Bank | LTn6K8B2r1Y8

Basis B

Dictionary: Contacts
Key | Value
---|---
Trent | 5B Manchester Rd

Structured map

Randomized permutation map

Linear map

AES-GCM-SIV | Key B
---|---
Noise
AES-GCM-SIV | Key A
---|---
Noise
AES-GCM-SIV | Key A'
---|---
Noise
AES-GCM-SIV | Key B'
Details: Free Space

• Locked Bases Are Indistinguishable from Free Space
  • Problem:
    - How to allocate a block without erasing locked data?
  • Solution:
    - Map all known Bases
    - Select a random subset of freespace equal to ~10% of disk -> cache it as "definitely free space"
    - Re-lock secret Bases
    - Allocate from "definitely free space" until exhausted
    - OOM -> go back to first step
PDDB General Properties

- Erasing a Basis is equivalent to forgetting the key
  - "I do not recall" === "The data never existed (or is erased)"

- Strong deniability versus a single forensic imaging event
  - Pros: Attacker cannot prove or deny that all Basis passwords have been disclosed
  - Cons: Attacker can force the deletion of undisclosed secret Bases by filling a known Basis with junk data
    - In some cases this is a desirable outcome

- Diminishing deniability versus repeated forensic imaging events
  - Small secret datasets are easier to deny
  - Disk can be re-encrypted/shuffled to restore deniability
PDDB Is Not a Panacea

- Deniability is fundamentally a **social tool**
  - Not all people can execute deniability to the same proficiency
  - Deniability is **optional**; it is not appropriate for all situations
  - However, **no** users can successfully deny anything without the option of strong plausible deniability
- PDDB is just one tool of many that are needed to help navigate upcoming legal challenges to privacy and security
Q&A
@bunniestudios

“From Boot to Root in One Hour”
https://www.bunniestudios.com/blog/?p=6336
https://precursor.dev
#betrusted:matrix.org

With thanks to:

Presentation CC-BY-SA 3.0
Some icons CC-BY 3.0 via flaticon.com:
smashicons, Appzgear, Gregor Cresnar, Freepik, Good Ware, Mavadee, Smallikeart, Rami McMin, photo3idea_studio