Hardware Reverse Engineering

Starbug & Karsten Nohl
University of Virginia
Critical hardware relies on proprietary security primitives
- These algorithms can easily be reverse-engineered
- Their security level is often low

When designing security, prepare for failure
- Goal should be low risk of large damage, but not perfect security
- Publicly reviewed algorithms and independent analysis yield best results
Lots of critical systems rely on secure hardware
- Smartcards for access control, payment tokens
- Also: satellite TV cards, car keys, printer cartridges, ...

Security often considered hard and expensive
- Hence, often excluded from initial design
  - Protection added after problems arise
  - Patchwork security is harder and more expensive!

Finding security bugs in hardware systems becomes ever easier, threat grows.
Security is a chain

- Its strength is determined by the weakest link
Example: Smart Cards

Cryptographic cipher

Challenge-response protocol

Cryptographic cipher
Example: Satellite TV

Encrypted data → Cipher → Decrypted data
Hardware security relies on
a) Key storage
b) Cryptographic cipher (encryption)

Many systems fail to acknowledge lack of secrecy in hardware

This talk discusses common weaknesses in secure key storage and proprietary encryption.
Reverse-engineering secret algorithms
1. Open chips
2. Find structures
3. Reconstruct circuit

Impact:
- Find proprietary encryption
- Open cryptographic key storage
Microchip Basics
Infineon SLE66, courtesy Flylogic
Infineon SLE66 address/data bus, courtesy Flylogic
Analyze chips using “last principles”

Principle #1: Chips are structured
  - Crucial for design partitioning and refactoring

Principle #2: Chips are designed to be read back
  - Enables prototyping and debugging

Complement analysis with “first principle”

Principle #3: Nothing can be hidden in silicon
  - Chips are self-contained; hence all data, programs, and algorithms are available on the chip
Protection Meshes

- Meshes can sometimes protect data, but not algorithms.

“Last resort”: Hide security in secret algorithms.
Reverse-Engineering Secret Algorithms
Chemically extract chips:
- Acetone
- Fuming nitric acid
Revealing Circuits

Polishing:
- Automated with machine
- Manually with sand paper

- Potential problem: tilt
- Solution: glue chip to block of plastic
Etching with HF (Hydrofluoric Acid)
Imaging Chips

- Simple optical microscope
  - 500x magnification
  - Camera 1 Mpixel
  - Costs < $1000, found in most labs

- Confocal microscope
  - Colors images by layer
  - Makes structures easy to spot
  - Expensive: > $10k
Deluxe Imaging: Confocal Microscope
Stitching Images

- Need to stitch 100x100μm images
- Tool of choice: hugin
- Borrowed from panorama photography

hugin:

![Hugin Panorama Tools Frontend](image)

Reference points

Starbug & Karsten Nohl – Hardware Reverse Engineering
Chip Layers

Cover layer (optional)

Interconnection layers

Logic layer

Transistor layer
Logic Gates – Inverter
Logic Gates – 2NOR

[Diagram showing logic gate symbols and connections labeled A, B, and Y]
The Silicon Zoo

- Collection of logic cells
- Free to everyone for study, comparison, and reverse-engineering of silicon chips
- Zoo wants to grow—send your chip images!

www.siliconzoo.org
Logic cells are picked from a library:
- Library contains fewer than 70 gate types
- Detection automated (template matching using MATLAB)
Automated Cell Detection
Logic Gates Interconnect

- Mifare: 1500 connections for Crypto-1
- DECT: 2000 connections for DSC
- Manually tracing connections
  - Tedious, time consuming
  - Error-prone (but errors easily spottable)
  - Tracing automated by now
Tracing Connections
Automated Tracing

Metal wire

Intra-layer via
Countermeasures

- Obfuscated placing and wiring of logic cells
  - May defeat human inspection, but not automated tools

- Dummy cells
  - Makes reversing harder, but not impossible

- Large chips
  - Huge effort, huge rewards?

- Self-destructive chips?
  - May protect secret keys, not secret algorithms

Source: flylogic.net
Mifare Classic Break

- Mifare cards use proprietary Crypto-1 algorithm
  - Never publicly reviewed for 20+ years
- We reverse-engineered the algorithm and announced insecurities at 24C3
- Feb/Mar: Reports find Crypto-1 to be strong enough for a “few more years”
  - We released more details about attacks
    - Final report recommends migration
- April: Dutch researchers publicly demonstrate attacks against Oyster
  - Law suit erupts, free speech prevails
  - Details published in October
Once strong cryptography is used, key storage becomes weakest link

- More ubiquitous systems typically have more copies of the secret keys in accessible places
Secret keys can be stored:

- **Online:**
  - Keys only stored on central server
  - Expensive setup, long response times

- **Semi-online:**
  - Devices receive keys at boot time
  - Keys often stored in DRAM at runtime; bad idea!

- **Offline:**
  - Devices “securely” store key copy
Secret keys should be
  ▪ Different for every user
    ▪ Requires many different keys
  ▪ Immediately accessible
    ▪ Requires small number of keys

Best practice: derive user keys from master key; store master key in „key vault“
Secret Key Storage

- **Hardware Security Modules (HSM)**
  - Used in ATMs (cash machine), few smart card readers
  - Use proprietary encryption
  - Hence, can be broken
    - Usually high effort (> $100,000)
- **Secure Access Modules (SAM)** are much easier to break
  - Credit card / smart card readers
Key Vault

Everything needed to disclose key is found on chip.

Finding secret algorithms might be costly.

Hardware Security Module (HSM)

HSM ID

Encrypted key

Proprietary Decryption

Master key

AES / 3DES

Card ID, sector, ...

Card key
SAM chips

- "Secure" Access Modules are standard microprocessors
  - Low effort to extract master keys
  - SIMs/SAMs are becoming cheaper and less secure!
  - (cell phones are not any better)

Source: Flylogic
Guidelines learned from past hacks include:

- Prepare for security breaks, no measure is perfect
  - Need: redundancy, “layering”
  - Need: migration plan
- Use standardized security
  - Never rely on your own security “inventions”
- Manage risks through threat modeling
  - Find acceptable balance between potential losses and cost of security
“There are no secrets in silicon.” —bunnie