“behave in the expected manner for the intended purpose”
Usually the Monkey Gets You

_____ Voting Machine

www._____com: public picture of its key
Why Hardware?

By nature, software *lives* in the Matrix but ...

... hardware *makes up* the Matrix.
The Myth of Crypto Performance

Baseline.
Pentium 4. 3.6GHz.
1GB RAM. 11000 MIPS. OpenSSL 0.9.7f

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES/CBC</td>
<td>70MB/sec</td>
</tr>
<tr>
<td>RC4</td>
<td>138MB/sec</td>
</tr>
<tr>
<td>MD5</td>
<td>18-615MB/sec</td>
</tr>
<tr>
<td>SHA1</td>
<td>18-340MB/sec</td>
</tr>
<tr>
<td>Modular MUL 1024</td>
<td>273000/sec</td>
</tr>
<tr>
<td>RSA1024 Sign</td>
<td>261/sec</td>
</tr>
<tr>
<td>RSA1024 Verify</td>
<td>5324/sec</td>
</tr>
<tr>
<td>3DES</td>
<td>26MB/sec</td>
</tr>
</tbody>
</table>
Now we have Physical Threats

**Invasive**
direct access to components
damaging vs. non-damaging

**Semi-Invasive**
no electrical contact

**Local Non-Invasive**
close observation of device’s operation
(consider also knowledge of attacker)

**Remote**
observation of device’s normal i/o
Also Software Threats

Usual software suspects
External I/O Interface Drivers
Internal OS
Application Bugs
Certification Standards

Hundreds
Common Criteria (ISO/IEC 15408)
Federal Information Protection standards (FIPS)
Trusted Computing Group (TCG)
Evaluation Assurance Levels (EAL)

EAL1: Functionally Tested
EAL1 is applicable where some confidence in correct operation is required, but the threats to security are not viewed as serious.

EAL2: Structurally Tested
Requires the cooperation of the developer in terms of the delivery of design information and test results.

EAL3: Methodically Tested and Checked
Maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices.

EAL4: Methodically Designed, Tested and Reviewed
Maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources (Suse ES 10, RedHat 5, costs $2+ mil.)

EAL5: Semi-formally Designed and Tested
Maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering (Smart cards, IBM z/OS).

EAL6: Semi-formally Verified Designed and Tested
Applicable to the development of security for application in high risk situations.

EAL7: Formally Verified Design and Tested
EAL7 is applicable to the development of security for application in extremely high risk situations. Practical application of EAL7 is currently limited to tightly focused security functionality that is amenable to extensive formal analysis. (a single device so far, smart cards?)
# FIPS 140-2 Security Levels

<table>
<thead>
<tr>
<th>Security Level 1</th>
<th>Security Level 2</th>
<th>Security Level 3</th>
<th>Security Level 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cryptographic Module Specification</strong></td>
<td>Specification of cryptographic module, cryptographic boundary, Approved algorithms, and Approved modes of operation. Description of cryptographic module, including all hardware, software, and firmware components. Statement of module security policy.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cryptographic Module Ports and Interfaces</strong></td>
<td>Required and optional interfaces. Specification of all interfaces and of all input and output data paths.</td>
<td>Data ports for unprotected critical security parameters logically separated from other data ports.</td>
<td></td>
</tr>
<tr>
<td><strong>Roles, Services, and Authentication</strong></td>
<td>Logical separation of required and optional roles and services. Role-based or identity-based operator authentication.</td>
<td>Identity-based operator authentication.</td>
<td></td>
</tr>
<tr>
<td><strong>Physical Security</strong></td>
<td>Production grade equipment. Locks or tamper evidence. Tamper detection and response for covers and doors. Tamper detection and response envelope. FPGA or EPP.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Operational Environment</strong></td>
<td>Single operator. Executable code. Approved integrity technique.</td>
<td>Referenced FPs evaluated at EAL3 with specified discretionary access control mechanisms and auditing.</td>
<td>Referenced FPs plus trusted path evaluated at EAL3 plus security policy modeling.</td>
</tr>
<tr>
<td><strong>Cryptographic Key Management</strong></td>
<td>Key management mechanisms: random number and key generation, key establishment, key distribution, key entry/output, key storage, and key revocation.</td>
<td>Secret and private keys established using manual methods may be entered or output in unencrypted form.</td>
<td>Secret and private keys established using manual methods shall be entered or output encrypted or with split knowledge procedures.</td>
</tr>
<tr>
<td><strong>Self-Tests</strong></td>
<td>Power-up tests: cryptographic algorithm tests, software/firmware integrity tests, critical functions tests. Conditional tests.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Mitigation of Other Attacks</strong></td>
<td>Specification of mitigation of attacks for which no testable requirements are currently available.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# FIPS 140-2 Physical Requirements

<table>
<thead>
<tr>
<th>Security Level</th>
<th>General Requirements for all Embodiments</th>
<th>Single-Chip Cryptographic Modules</th>
<th>Multiple-Chip Embedded Cryptographic Modules</th>
<th>Multiple-Chip Standalone Cryptographic Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Level 1</td>
<td>Production-grade components (with standard passivation)</td>
<td>No additional requirements.</td>
<td>If applicable, production-grade enclosure or removable cover.</td>
<td>Production-grade enclosure.</td>
</tr>
<tr>
<td>Security Level 2</td>
<td>Evidence of tampering (e.g., cover, enclosure, or seal)</td>
<td>Opaque tamper-evident coating on chip or enclosure.</td>
<td>Opaque tamper-evident encapsulating material or enclosure with tamper-evident seals or pick-resistant locks for doors or removable covers.</td>
<td>Opaque enclosure with tamper-evident seals or pick-resistant locks for doors or removable covers.</td>
</tr>
<tr>
<td>Security Level 3</td>
<td>Automatic zeroization when accessing the maintenance access interface. Tamper response and zeroization circuitry. Protected vents.</td>
<td>Hard opaque tamper-evident coating on chip or strong removal-resistant and penetration resistant enclosure.</td>
<td>Hard opaque potting material encapsulation of multiple chip circuitry embodiment or applicable Multiple-Chip Standalone Security Level 3 requirements.</td>
<td>Hard opaque potting material encapsulation of multiple chip circuitry embodiment or strong enclosure with removal penetration attempts causing serious damage.</td>
</tr>
</tbody>
</table>
“The cryptographic module components shall be covered by potting material or contained within an enclosure encapsulated by a tamper detection envelope (e.g., a flexible mylar printed circuit with a serpentine geometric pattern of conductors or a wire-wound package or a non-flexible, brittle circuit or a strong enclosure) that shall detect tampering by means such as cutting, drilling, milling, grinding, or dissolving of the potting material or enclosure to an extent sufficient for accessing plaintext secret and private keys cryptographic keys ...”
Instances

• Encryption disks
• USB tokens
• RSA SecurID
• TPMs
• Smart Cards
• Secure Co-processors
• CPU-level techniques
• PUFs
• misc others
Full Disk Encryption

- **Key Management:** internal
- **Authentication:** mostly external (BIOS, or app)
  - Pre-boot authentication
  - “hashed passwords” on drive
  - emergency password recovery file outside
  - multiple users
- **Encryption**
  - On-board AES – <3% overhead / traditional drive
  - “disk erase” = change encryption keys
- **On Chipset:** Intel vPro chipsets might add encryption in the south bridge (PCI/IDE/..., not until 2010)
USB Storage

Carry secrets on USB token, often un-locked with a password. Allows for 2-factor authentication.
RSA SecurID

1. Alice generates SID(time) using her SecurID device.
2. Alice sends SID(time) and user, pass to Bob.
3. Bob checks SID(time) is time synchronized.
4. Bob approves the transaction.

Mallory tries to intercept the SID(time) communication.

Eve tries to impersonate Alice using SecurID.
Trusted Platform Module (TPM)

Microcontroller that stores keys, passwords and digital certificates.

- Non-Volatile Storage
- Platform Configuration Register (PCR)
- Attestation Identity Key (AIK)
- Program Code
- Communications
- Random Number Generator
- SHA-1 Engine
- Key Generation
- RSA Engine
- Opt-In
- Exec Engine
Can the Trusted Platform Module control what software runs?
No. [...] it can only act as a 'slave' to higher level services and applications by storing and reporting pre-runtime configuration information. [...] At no time can the TCG building blocks 'control' the system or report the status of [running] applications.
... but it can do “attestation”

Idea: authenticate next link in chain before passing control.  
e.g., BIOS to OS, VMM to VM to Guest OS

“measure” = authenticate identity
Measurement

“RTM” = root of trust measurement (e.g., BIOS)
“SML” = stored measurement log (external)
“PCR” = hash(prev,extend), cannot be forced
Verification

SML → Event Structure
1. Extend Value
2. Extend Data
3. TPM
4. PCR
5. Key

Verifier

Q: Why trust?
A: TPM signature key
Breaking Key Correlation: AIK CA

“AIK” = attestation identity key
(2048 bit RSA generated by TPM, unlimited number of them)
“AIK CA” = external certificate authority for AIKs
Dynamic vs. static PCRs

Static PCRs: 0-16
Reset by reboot only

Dynamic PCRs: 17-23
Can be reset to 0 without reboot
Reboot sets them to 1 (can remotely distinguish reboot from dynamic reset)

Special PCR 17
Only hardware CPU command can reset it.
SKINIT instruction can trigger that.
Software cannot reset PCR 17
Attacking the TPM

TPM Reset Attack
Sean Smith et al.,
www.cs.dartmouth.edu/~pkilab/sparks/

also
Programming the TPM

Trusted Software Stack (TSS) Libraries
Use Windows TSS dll
Linux TSS SDK

Developer Support and Software
http://www.trustedcomputinggroup.org/developers/
eXecute Only Memory (XOM)

Smart Cards/Chips

Contact smart card

RFID smart card

Functionality
DES, RSA(?), MD5, SHA-1, 4-16kb ROM/RAM, soon 1MB (!), 16bit 10-30MHz CPU, 10-80kbps (source: Sharp)
Architecture

Philips Smart MX
Power Analysis

Cryptographic device
(e.g., smart card and reader)

Control, Cyphertexts

Control, Waveform data

Oscilloscope

Computer
US Passport

Made by Smartrac (Netherlands) and shipped to the US from Europe via Thailand. In 2007 China allegedly stole the RFID chip.
Heat and Acids
Polishers and Microscopes

Figure 1: (a) Source image of layer 2 after edge detection; (b) after automated template detection.


[Garcia, van Rossum, Verdult, Schreur, Wirelessly Pickpocketing a Mifare Classic Card, ]
Weak: LFSR Cipher, RNG

Figure 2: Crypto-1 stream cipher and initialization.
Smart Card: Windows login

Steps 1 - 3

WINLOGON

MSGSINA
collects
user’s
PIN

LSA

Active Directory

User Object:
Name
Group membership
Public certificate

Step 5

Contains: User’s certificate
signed with private key

Step 6

KDC

Step 7

Contains: TGT
Contains: Logon
Session Key
Encrypted with
Public Key

Step 8: Kerberos Client
Decrpts the Session Key
With its Private Key

Uses public key from AD certificate
to decrypt certificate. If
decryption works, creates TGT

Smart Card

PIN

Private

User’s
Public
Certificate

Stony Brook Network Security and Applied Cryptography Laboratory
Cell Broadband Engine

Apps: PS3, Xbox 360, IBM BladeCenter, HPC, Video cards etc.
Cell BE: Secure Processing Vault

Idea: isolate application.

Isolated SPE
- Disengaged from the bus
- SPE LS contains app code + data
- PPE-SPE control mechanisms are disabled
- Only external action possible is cancel: all information in the LS and SPE is erased before external access is re-enabled.
- All LS reads and writes from units on the bus (PPE, SPEs, I/O) have no effect on the locked-up region of the LS.
- Dedicated area of the LS is left open to data transfers.
- Any number of SPEs can be in isolation mode at any given time.
Cell BE: Runtime Secure Boot


1. Isolation mode is initiated.
2. Previous application is stopped and cancelled.
3. Application is fetched in and checked by the hardware authenticator
   - based on a hardware key and cryptographic algorithm
4. Integrity check fails; execution stopped
   - Application was tampered
5. Check succeeds; will kick-start the application's execution in isolation mode.
ARM TrustZone: “allows the system to be more easily partitioned for security while maintaining hardware-backed protection for the security sub-system.”

TrustZone adds a “parallel world” to allow trusted programs and data to be safely separated from the operating system and applications.
TrustZone: Partitioning

**Impact layers:** Bus (DMA controllers), RAM (Memory Wrapper, Address Space Controller, Cache controllers), I/O (secure bridges for peripherals).

**Chips:** ARM SecureCore Chips (SC100/200/300)
TrustZone: Writing Apps

![Diagram showing the comparison between usual and TrustZone implementation of applications.]

- **Key storage in main memory**
- **Usual Implementation**
  - API Implementation
  - Secure key storage

- **TrustZone implementation**
  - TrustZone access driver
  - TZ Monitor
  - Secure Kernel
  - API Implementation

- SW provided by ARM
Secure State Machine “guarantees policies while entering / executing / exiting secure environment”, automatic secured DMA transfers (bus-level encryption ?), secure chip interconnect, hardware crypto, ARM TrustZone
SKINIT (AMD)/SENTER (Intel)

kernel says
“SKINIT <address of SLB>”

CPU
disables DMA to SLB,
interrupts, debugging
resets PCR 17-23
transfers SLB to TPM
enters flat 32-bit addressing
jumps to entry point

TPM
measures SLB into PCR 17
Flicker: using SKINIT (AMD)

Left: a traditional computer with an application that executes sensitive code (S). Right: Flicker protects the execution of the sensitive code. Shaded portions represent components that must be trusted; applications are included on the left because many run with super-user privileges.

[McCune et al., "Flicker: An Execution Infrastructure for TCB Minimization", EuroSys 2008]
SLB core extends a well known value (function of input/output values of PAL + random nonce from remote party) in PCR 17: allow remote party to distinguish between values generated by the PAL (trusted) and those produced by the resumed OS (untrusted)
Intel SGX

Untrusted Part of App

Create Enclave

Call Trusted Func

Trusted Part of App

Execute

Return

Privileged System Code
OS, VMM, BIOS, SMM...
# Intel SGX

## Introduction to Trusted Hardware

### Intel SGX Instructions

<table>
<thead>
<tr>
<th>Super.</th>
<th>Description</th>
<th>User</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EADD</td>
<td>Add a page</td>
<td>EENTER</td>
<td>Enter an enclave</td>
</tr>
<tr>
<td>EBLOCK</td>
<td>Block an EPC page</td>
<td>EEXIT</td>
<td>Exit an enclave</td>
</tr>
<tr>
<td>ECREATE</td>
<td>Create an enclave</td>
<td>EGETKEY</td>
<td>Create a cryptographic key</td>
</tr>
<tr>
<td>EDBGRD</td>
<td>Read data by debugger</td>
<td>EREPORT</td>
<td>Create a cryptographic report</td>
</tr>
<tr>
<td>EBDGWR</td>
<td>Write data by debugger</td>
<td>ERRESUME</td>
<td>Re-enter an enclave</td>
</tr>
<tr>
<td>EINIT</td>
<td>Initialize an enclave</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELDB</td>
<td>Load an EPC page as blocked</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELDU</td>
<td>Load an EPC page as unblocked</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPA</td>
<td>Add a version array</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EREMOV E</td>
<td>Remove a page from EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETRACE</td>
<td>Activate EBLOCK checks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EWB</td>
<td>Write back/invalidate an EPC page</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Intel SGX: Remote Attestation
Intel SGX: Local Attestation

The diagram illustrates the interaction between two enclaves, Enclave A and Enclave B, with applications Application A and Application B. The process involves the following steps:

1. Enclave A initiates the communication with Enclave B.
2. Application A sends a message to Enclave B.
3. Enclave B processes the message and sends a response to Application B.

This framework ensures secure communication and attestation between the enclaves and applications.
Acalis CPU 872 Secure Processor

- Secure boot
- Encrypt/decrypt
- Secure interconnect
- Hardware firewall
- Triggered zeroization signal
- Unique serial code

More? Ryan?
CPU872: “Secure Anchor”
CPU872: “Secure Mobile Comp.E.”
Secure Co-Processors

“A secure coprocessor is a general-purpose computing environment that withstands physical and logical attacks.

The device must run the programs that it is supposed to, unmolested. You must be able to (remotely) distinguish between the real device and application, and a clever impersonator.

The coprocessor must remain secure even if adversaries carry out destructive analysis of one or more devices.”
Dyad and Strongbox


SCPU: IBM 4764-001 PCI-X

266MHz PowerPC. 64MB RAM. 64KB battery-backed SRAM storage. Crypto hardware engines: AES256, DES, TDES, DSS, SHA-1, MD5, RSA. FIPS 140-2 Level 4 certified.
IBM 4764-001 Architecture

Physical Security (Sense and response)
- CFU
  - PPC 405GPr
SDRAM
- Battery-Backed RAM
Flash (RAW)
ROM (ROMmed Flash)
FPGA Flash

Custom comm. hardware
- Cryptoengines (IBM UltraCrypt 2)
  - TDES (DES)
  - AES
  - SHA-1
  - MD5
  - Pubkey (RSA)

Hardware lock (access control), EEPROM

External interface (PCI-X etc.)

Secure enclosure
- Batteries
- Misc. support circuitry
- PCI-X base board

PCI-X bridge
- Card PCI-X interface
  - RS-232 port
  - Ethernet port
# IBM 4764-001 Segments

<table>
<thead>
<tr>
<th>Segment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Basic code</td>
</tr>
<tr>
<td></td>
<td>The basic code manages coprocessor initialization and the hardware component interfaces. This code cannot be changed after the coprocessor leaves the factory.</td>
</tr>
<tr>
<td>1</td>
<td>Software administration and cryptographic routines</td>
</tr>
<tr>
<td></td>
<td>Software in this segment:</td>
</tr>
<tr>
<td></td>
<td>• Administers the replacement of software already loaded to Segment 1.</td>
</tr>
<tr>
<td></td>
<td>• Administers the loading of data and software to segments 2 and 3.</td>
</tr>
<tr>
<td></td>
<td>• Is loaded at the factory, but can be replaced using the CLU utility.</td>
</tr>
<tr>
<td>2</td>
<td>Embedded operating system</td>
</tr>
<tr>
<td></td>
<td>The coprocessor Support Program includes the operating system; the operating system supports applications loaded into Segment 3. Segment 2 is empty when the coprocessor is shipped from the factory.</td>
</tr>
<tr>
<td>3</td>
<td>Application software</td>
</tr>
<tr>
<td></td>
<td>The coprocessor Support Program includes a CCA application program that can be installed into Segment 3. The application functions according to the IBM CCA and performs access control, key management, and cryptographic operations. Segment 3 is empty when the coprocessor is shipped from the factory.</td>
</tr>
</tbody>
</table>
Introduction to Trusted Hardware

Performance

<table>
<thead>
<tr>
<th>Function</th>
<th>Context</th>
<th>IBM 4764</th>
<th>P4 @ 3.4Ghz</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA sig.</td>
<td>512 bits</td>
<td>4200/s (est.)</td>
<td>1315/s</td>
</tr>
<tr>
<td></td>
<td>1024 bits</td>
<td>848/s</td>
<td>261/s</td>
</tr>
<tr>
<td></td>
<td>2048 bits</td>
<td>316-470/s</td>
<td>43/s</td>
</tr>
<tr>
<td>RSA verif.</td>
<td>512 bits</td>
<td>6200/s (est.)</td>
<td>16000/s</td>
</tr>
<tr>
<td></td>
<td>1024 bits</td>
<td>1157-1242/s</td>
<td>5324/s</td>
</tr>
<tr>
<td></td>
<td>2048 bits</td>
<td>976-1087/s</td>
<td>1613/s</td>
</tr>
<tr>
<td>SHA-1</td>
<td>1KB blk.</td>
<td>1.42 MB/s</td>
<td>80 MB/s</td>
</tr>
<tr>
<td></td>
<td>64 KB blk.</td>
<td>18.6 MB/s</td>
<td>120+ MB/s</td>
</tr>
<tr>
<td></td>
<td>1 MB blk.</td>
<td>21-24 MB/s</td>
<td></td>
</tr>
<tr>
<td>DMA xfer</td>
<td>end-to-end</td>
<td>75-90 MB/s</td>
<td>1+ GB/s</td>
</tr>
<tr>
<td>CPU freq</td>
<td></td>
<td>266MHz</td>
<td>3400Mhz</td>
</tr>
<tr>
<td>RAM</td>
<td></td>
<td>16-32MB</td>
<td>2-4GB</td>
</tr>
</tbody>
</table>

Table 3: Hardware Performance Overview. SCPPUs (e.g., IBM 4764-001 PCI-X) are orders of magnitude slower for general purpose computation than main CPUs (Pentium 4, 3.4Ghz, OpenSSL 0.9.7f). On the other hand, the crypto acceleration in the SCPU shows in direct speedup of crypto operations. Also optimized key setups might result in slightly different numbers for the main CPU.
Limitation: Heat

Dissipating heat while being tamper-proof.
Attacks

Possible Attacks
Probe Penetration
Power Sequencing (filter on power supply)
Radiation
Temperature Manipulation
Improper battery removal

Response (on tamper detection)
Zeroes its critical keys
Destroys its certificates
Is rendered inoperable
## 4764: Ranges

<table>
<thead>
<tr>
<th></th>
<th>Operating environment</th>
<th>Storage environment</th>
<th>Shipping environment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Temperature</strong></td>
<td>+10°C - +40°C (+50°F - +104°F)</td>
<td>+1°C - +60°C (+33.8°F - +140°F)</td>
<td>-15°C - +50°C (+5°F - +140°F)</td>
</tr>
<tr>
<td><strong>Relative humidity</strong></td>
<td>8 - 80%</td>
<td>5 - 80%</td>
<td>5 - 100%</td>
</tr>
<tr>
<td><strong>Wet bulb</strong></td>
<td>+27°C (+80.6°F)</td>
<td>+29°C (+84.2°F)</td>
<td>+29°C (+84.2°F)</td>
</tr>
<tr>
<td><strong>Pressure (minimum)</strong></td>
<td>768 mbar</td>
<td>700 mbar</td>
<td>550 mbar</td>
</tr>
</tbody>
</table>
relationship between “tamper-evident”, “tamper-resistant”, “tamper-proof” etc.
Miscellaneous “SCPU”s

**netHSM**
Networked shareable cryptographic resource for multiple servers. Just crypto, no tamperXXX CPU.

**nShield**
FIPS 140-2 level 2/3 TPM/SCPU

**miniHSM**
FIPS 140-2 level 3 mini SCPU
Physically Unclonable Function

Based on a physical system
Easy to evaluate (using the physical system)
Its output looks like a random function
Unpredictable even for an attacker with physical access

Silicon PUF: no two ICs are the same
PUFs as Unclonable Keys

Set of challenge/response pairs
PUFs: Applications

Anonymous Computation
Run computations remotely and ensure correct results. Return a certificate showing they were run correctly.

Software Licensing
Sell software which only runs on specific PUF-identified chip.
Areas

Finance
Online banking, ATMS

Commerce
Energy, Smart-grid, Healthcare

Government
Regulatory compliance

Military
Secure battle-field devices
Understand *your* adversary

e..g., physical, insider vs. software-only, remote

Understand defenses and cost of attack

$10^1$ of overcoming defenses should not protect

$10^6$
Thanks 😊
some rare references


“Practical server privacy with secure coprocessors”, IBM Systems Journal 2001, S. W. Smith, D. Safford


A. Iliev, S.W. Smith, “Prototyping an Armored Data Vault: Rights Management on Big Brother’s Computer.”, Privacy-Enhancing Technology 2002


Maheshwari, Vingralek, and Shapiro, How to build a trusted database system on untrusted storage, OSDI 2000


