Secure Systems: Hardware is the Answer

Secure, Trustworthy, Assured and Resilient Semiconductors and Systems (STARSS) Workshop
May 22, 2014

Benjamin Jun
VP & Chief Technology Officer
Cryptography Research, Inc. a division of Rambus
Network Working Group
Request for Comments: 1945
Category: Informational

T. Berners-Lee
MIT/LCS
R. Fielding
UC Irvine
H. Frystyk
MIT/LCS

May 1996

Hypertext Transfer Protocol -- HTTP/1.0

Status of This Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

IESG Note:

The IESG has concerns about this protocol, and expects this document to be replaced relatively soon by a standards-track document.

RFC 1945: HTTP 1.0
(May 1996)

Al Gore
Complexity and Security
Life with Moore’s law

Millions of Transistors in Intel CPUs  
1
10
100
CPU data courtesy Intel Corp.
Millions of Lines of Code in Microsoft Operating Systems

Human Intelligence

CPU data courtesy Intel Corp.
What happens as complexity increases?

What if the number of elements doubles?

- 2X more opportunities for bugs
  - Odds of having zero catastrophic weaknesses squared (e.g., 20% -> 4%)

- Number of interactions increases as the square of complexity
  - Odds of having zero catastrophic weaknesses raised 4th power (e.g., 20% -> 0.016%)
About software...

“... the tape can be moved back and forth through the machine, this being one of the elementary operations of the machine. Any symbol on the tape may therefore eventually have an innings.”

-- Alan Turing, “Intelligent Memory”, 1948

Even in the original Turing machine, any piece of code could wreck security of the entire system.
Coping strategies can help...

• Various strategies:
  ◦ Security training
  ◦ Design/code reviews
  ◦ Safer languages
  ◦ Code scanning tools
  ◦ Better components/libraries
  ◦ Additional layers of abstraction
  ◦ Anomaly detection
  ◦ Assertions
  ◦ Monitoring/canaries

• But complexity overwhelms everything

4. Heartbeat Request and Response Messages

The Heartbeat protocol messages consist of their type and an arbitrary payload and padding.

```c
struct {
    HeartbeatMessageType type;
    uint16 payload_length;
    opaque payload[HeartbeatMessage.payload_length];
    opaque padding[padding_length];
} HeartbeatMessage;
```

The total length of a HeartbeatMessage MUST NOT exceed $2^{14}$ or max_fragment_length when negotiated as defined in [RFC6066].

type: The message type, either heartbeat_request or heartbeat_response.

RFC6520 ➔ Heartbleed
About hardware...

• The first rule about hardware: try to make silicon shippable even when there are bugs

• The art of using “free” transistors
  ◦ Multi-core
  ◦ Complex memory hierarchies
  ◦ Half-baked stuff + chicken bits
  ◦ Functionality gaskets and ways to fix in-field
  ◦ Explosion of fuse maps

• RTL looks like code, but is concurrent + subtle
  ◦ …just read published CPU errata!

• Hardware tools are built for 1/1000th the number of users
  ◦ Tool problems created a cottage industry: Formal equivalence checking
Platform Trust
Apps require a secure, reliable foundation

- What gets to run on the platform?
  - Boot / code authentication
  - Secure debug lock

- Am I in the real world or the matrix?
  - Environment attestation
  - Peripheral authentication

- Do my secrets remain opaque?
  - Application partitioning
  - Hardware-based secure key storage
Example: EM analysis of an RSA implementation

- Android app with RSA implementation on modern 4G phone
- Magnetic field pickup coil
- Measurements collected during computation of \( M^d \mod N \)

Commercial standards requiring side-channel resistance

- PCI
- Movie Labs
- FIPS 140-3
- Common Criteria
Trust from the top down

- Device enrollment
- System auditing & risk management
- Online revocation
- Remote management & updates
Lifecycle considerations for “Internet Things”

“Direct to field”

- Early provisioning of dev. credentials
  - Inject keys, certificates
  - Enroll device
  - May be done before OS load
  - Often an outsourced (faraway) manufacturing site

Limited UI for administration steps

- Device administration secured by base credentials
  - In-field challenge/response authentication
  - Add/update user credentials
  - Send signed updates
Trust meets in the middle

Identity + key provisioning
Authentication service
Secure session management
Security updates

Identity + key management
Sandboxed secrets
Partitioning of critical state
Reliability & integrity
Trust Boundaries
Security for command & control

• Egyptian signet ring
  ◦ Used by pharaohs & officials
  ◦ ~500BC

• Mark of the fisherman
  ◦ Individualized for each pope
  ◦ On death, Cardinal Carmerlengo to locate ring & destroy seal
  ◦ Earliest note in 1265
Security for command & control

US nuclear “football”

Starfleet auto-destruct procedure

…but most security problems are not this clean or well funded!
Things got more complicated

US NIKE missile site
Angel Island, San Francisco

Honeywell 6180 (1973)
Multics w/Hardware ring 0

American EP-3E returns from Hainan, China (7/2001)
Real life control and responsibility: Mobile

“SENSITIVE STUFF”
Application
Application marketplace
User
Device owner
Carrier / System operator
OS vendor
OEM device manufacture
OEM code
Outsourced chip assembly and test
Semiconductor fab
SoC designer

Can we really aggregate trust to “Hypervisor”?

“The Supply Chain” really?

Platform “Owners” with different control, responsibility domains
Privacy?

It takes a village!
Let’s get to work!

Benjamin Jun
ben@cryptography.com