Hardware Security Modules: The Ultimate Black Boxes

Ryan Lackey
<ryan@venona.com>

28 January 2019

USENIX Enigma 2019, Burlingame, CA

What is a Hardware Security Module (HSM)?

Physically secure processing module designed for key management and processing

Server (Direct-attach and networked)

Client (mobile) and embedded

Cloud (virtualized)

Processor/platform technologies

HSMs are critical system components but also hard to inspect, use, and trust

HSMs are becoming more important and relevant again after a period of stasis

HSMs have new uses, with new technical, architectural, and business requirements

Important concepts in the HSM world

- Trusted Computing Base (TCB)
- Bootloader security (multi-stage)
- Remote attestation
- Key management roles
- Backup/export functionality
- Certification

HSMs are critical system components but also hard to inspect, use, and trust

Migration to the Cloud meets an anchor

Hard to inspect a black box

Outdated, hard to use tools

Inherent tension between tamper-response and reliability

Cost and sales/licensing processes

Automatic signing/just a big smartcard problem

HSMs are becoming more important and relevant again after a period of stasis

History of the HSM in 60 seconds

Devices: shrinking from safes to chips

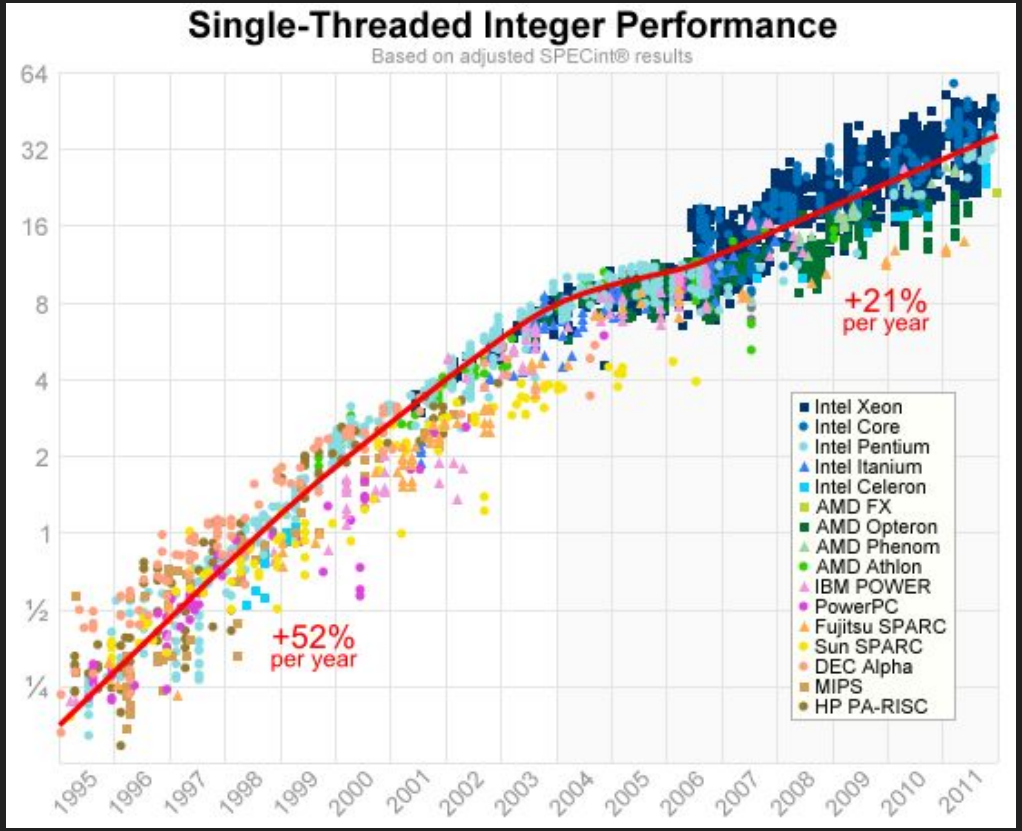
Applications: banking, infrastructure (CA, DNSSEC)

Vendors: major consolidation

Cost: Generally has gone up

Product cycles: Longer, legacy deployments

Vendor consolidation



Horizontal scaling and CPU cryptographic performance

DevOps world and orchestration

Seems like a declining market, but no!

Cryptocurrency

Key management and authentication

More mature security models for applications

Better deployment models and tools

HSMs are key to solutions to many of the biggest problems in security today

Key management for increasingly high-value keys

Separation of roles and internal control

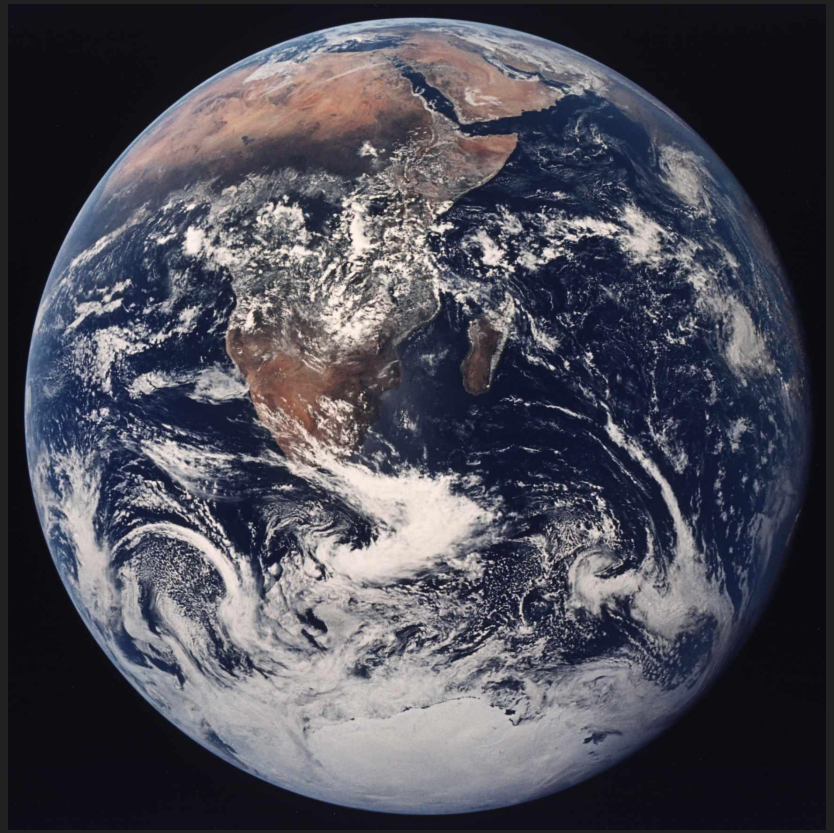
Someone else’s physically remote hardware with your critical secrets

Third-party application updates and trust

Limiting system impact of bugs and breaches

Lots of non-Internet applications use HSMs extensively (particularly finance)

HSMs (on client devices, ie mobile) are well on their way to world domination



So why haven’t HSMs taken over the world yet?

HSMs have new uses, with new technical, architectural, and business requirements

Conventional server-side HSMs still have painful tools, price points, etc.

Cloud-based HSM products are early stage (and lots of hybrid/legacy tech)

Custom application development inside the HSM is even more niche/difficult/slow

Processor/platform security is “free” but hard to develop for and has limitations

Certification process (NIST FIPS 140-2) delays, limitations (algorithms!)

The easiest path forward

Less-expensive, non-FIPS or FIPS-optional (e.g. Yubico YubiHSM 2)

Non-FIPS security platforms like USB Armory and continued embedded progress

Simplified development of on-HSM secure code (beyond PKCS11)

Clouds integrating HSMs internally (continuing past HSM-backed KMS)

Clouds offering optional non-FIPS HSMs for diverse algorithm needs

Permissionless, easy deployment using platform security with remote attestation

Hybrid HSM and platform security solutions

Successor to FIPS 140-2 certification for more agile environments

The ideal world

Gap between conventional HSMs and platform security

Dream HSM of 2020s

- Fundamentally open
- Designed for inspection and trust
- Range of price/performance levels
- Designed for virtualization/cloud

Why this can work?

- Mostly a software problem
- Strong early applications and tools exist
- Existing standards for backward compatibility
- Viable early hardware platforms

Roadblocks?

- Cloud provider adoption of hardware
- Incumbent vendors at high-end
- Pricing pressure from the platform security
- Limited deployment of HSM-required applications

Questions: email <ryan@venona.com>

ENIGMA®