Security and Trust

Kaarthik Sivakumar, Principal Engineer

Dan Backman, Technical Marketing Engineer

Rahul Mavila, Product Manager

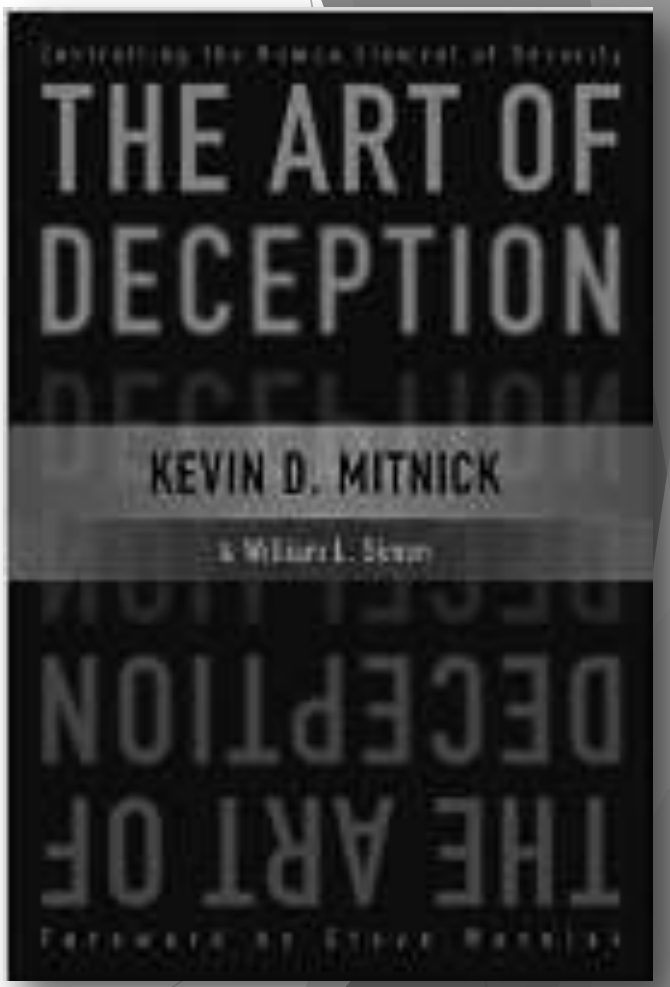
Cisco Systems

Agenda

- Attacks and More (Attacks)
- Securing Systems
- Trust and Integrity

Attacks and More (Attacks)

"This was a company that, like many, had what I refer to as candy security, after a description first used by two Bell Labs researchers, Steve Bellovin and Steven Cheswick. They described such security as **"a hard crunchy shell with a soft chewy center"**— like an M&M candy. The outer shell, the firewall, Bellovin and Cheswick argued, is not sufficient protection, because once an intruder is able to circumvent it, the internal computer systems have soft, chewy security. **Most of the time, they are inadequately protected**"




The New York Times
https://www.nytimes.com/2016/02/18/technology/apple-timothy-cook-fbi-san-bernardino.html?rref=collection%2Fnewseventcollection%2Fapple-fbi-
Apple Fights Order to Unlock San Bernardino Gunman's iPhone
Putin Ordered 'Influence Campaign' Aimed at U.S. Election, Report Says
.nytimes.com/2017/01/06/us/politics/russia-hack-report.html?_r=0
http://money.cnn.com/2017/04/14/technology/windows-exploits-shadow-brokers/
NSA's powerful Windows hacking tools leaked online
The Athens Affair
http://spectrum.ieee.org/telecom/security/the-athens-affair
SPECTRUM
/www.wired.com/2013/09/nsa-
WIRED
www.telegraph.co.uk/news/worldnews/europe/germany/10407282/Barack-Obama-approved-tapping-Angela-Merkels-phone-3-years-ago.html
US CERT
"Organizations can no longer rely on perimeter devices to protect the network from cyber intrusions… There has never been a greater need to improve network infrastructure security"
Alert TA16-251A, September 2016
Edward Snowden: Leaks that exposed US spy programme
http://www.bbc.com/
d-us-canada-23123
www.forbes.com/sites/thomasbrewster/2017/03
leaks-cia-apple-mac-iphone-
5a337bd71e3b
Forbes
http://www.networkworld.com/article/2984327/security/synful-knock-router-exploit-isn-t-going-away-soon.html
SYNful Knock router exploit isn't going away soon
NETWORK WORLD
http://www.newsweek.com/macron-emails-hacked-le-pen-france-election-trump-russia-clinton-putin-595808
WORLD
EMMANUEL MACRON'S EMAIL HACKED DAYS BEFORE FRENCH PRESIDENTIAL ELECTION
N

Types of Attacks

- Attack the End System – IOT Devices, Windows, Linux, Mac OS
- Attack the Intermediates – Routers, Switches
- Attack the Traffic – DDoS, BGP and more
- Attack the Supply Chain
- Attack the Person – Phishing, Social Engineering

Motivations

- Ransomware
- Destroy critical infrastructure
- Competitive Advantages
- Geo-political Advantages

MIRAI

Devices infected by Mirai continuously scan the Internet for the IP addresses of Internet of things (IoT) devices. Mirai includes a table of IP address ranges that it will not infect, including private networks and addresses allocated to the United States Postal Service and Department of Defense.[14]

on. After a reboot, unless the login password is changed immediately, the device will be reinfected within minutes.[14] Upon infection Mirai will identify "competing" malware and remove them from memory and block remote administration ports.[17]

If a IoT device responds to the probe, the attack then enters into a brute-force login phase. During this phase, the attacker tries to establish a Telnet connection using predetermined username and password pairs from a list of credentials. Most of these logins are default usernames and passwords from the IoT vendor. If the IoT device allows the Telnet access, the victim's IP, along with the successfully used credential is sent to a collection server.

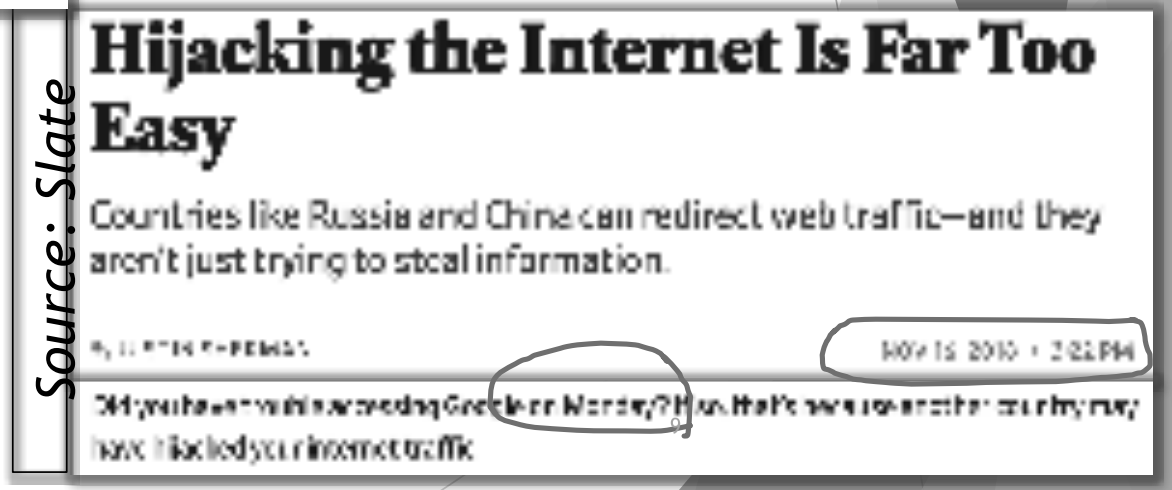
Mirai then identifies vulnerable IoT devices using a table of more than 60 common factory default usernames and passwords, and logs into them to infect them with the Mirai malware.[14][15][16]

Victim IoT devices are identified by "first entering a rapid scanning phase [16] where it asynchronously and "statelessly" sent TCP SYN probes to pseudo-random IPv4 addresses, excluding those in a hard-coded IP blacklist, on Telnet TCP ports 23 and 2323".[16]

Source: Wikipedia

BGP hijacking (sometimes referred to as **prefix hijacking**, **route hijacking** or **IP hijacking**) is the illegitimate takeover of groups of IP addresses by corrupting Internet routing tables maintained using the Border Gateway Protocol

How Pakistan knocked YouTube offline (and how to make sure it never happens again)



Source: Slate

Hijacking the Internet Is Far Too Easy

Countries like Russia and China can redirect web traffic—and they aren't just trying to steal information.

ThreatList: Attacks on Industrial Control Systems on the Rise

Source:
http://threatpost.com

Stuxnet targets SCADA systems and is believed to be responsible for causing substantial damage to Iran's nuclear program.

Attacks on SCADA (Supervisory Control and Data Acquisition) and PLC Systems
Targeted at specific types of devices, countries and locations.
Typically Nation-State Attackers

So Many More Types

- Social Engineering leading to other attacks
- Ransomware
- Government Monitoring
- Supply Chain
- DDoS
- …..

Securing Systems is Hard – Very Very Hard

- Complex Hardware, Software and People Interactions
- Too Many Moving Parts
- Just one Hole is Sufficient
- There are many many unknown holes that are not patched since they are … unknown!

An Attack Is Practically Guaranteed and Most Likely Will Be Successful

> Your device, your network and your data either have already been breached or will be soon!

So Do We Just Hand Over Our Keys?

Source:
https://xkcd.com/538/

Make it Expensive for the Attacker to Exploit You

Securing Systems

Secure Development Lifecycle for Better Defense

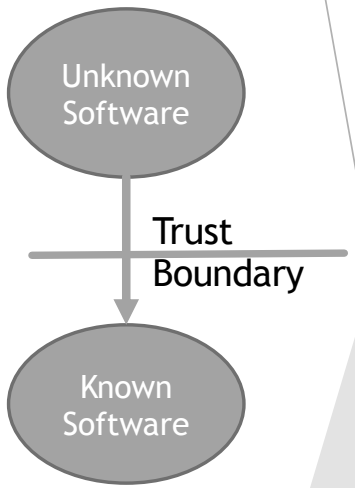
Secure Development Lifecycle: **Requirements**

Clearly Define Requirements!

- Logging / Auditing
- Authentication / Authorization
- Boot Integrity
- Runtime Integrity
- Operational Processes
- Threat Surface Reduction

Secure Development Lifecycle: **Design**

- Threat Modeling
  - Identify Security Threats
  - Address or Mitigate Threats
  - Trust Boundary between known and unknown software
- **Define clear APIs** with clearly documented Inputs and Outputs
- **Use the same APIs** internally as well as externally
- **Create Secure Enclaves** for isolated execution of sensitive code
- **Use Role Based Access Control (RBAC),** Network Namespaces and Containers to isolate execution and authorization space.
- Choose Strong, **Standard Cryptographic** Algorithms.
- **Regularly time out** passwords and password attempts



Secure Development Lifecycle: **Coding**

W^X : Write ^ Execute. Pages can be either Write or Execute but not both. W^X, DEP, NX

Address Space Layout Randomization: Randomize Locations of Stack, Heap, Text and Data Segments. Prevents attackers from targeting memory corruption vulnerabilities

SafeC Libraries: Set of Libraries that replace dangerous string functions – memcpy, strcpy, etc – with functions that perform addition checks.

Object Size Checking: Set of Compiler provided APIs to check buffer sizes at compile time or run time to determine possible overflow

Input Validation: *Most Important Check* to perform on any data taken in from the *"outside"*. All data must be checked before they are acted upon. Most common vulnerabilities and attacks are due to bad input

Common APIs: Maintain common set of APIs for common functions. DO NOT let each developer write their own set of APIs for common functions – they are likely to create subtle differences with disastrous consequences.

Sign Your Code: Sign your binaries and verify signatures prior to use.

Secure Development Lifecycle: **Coding**

DO NOT WRITE YOUR OWN CRYPTOGRAPHIC FUNCTIONS

Secure Development Lifecycle: **Testing**

Fuzz Testing, Penetration Testing and Vulnerability Testing:
Go Beyond Positive Feature Tests

Security Testing Requires a Different Mindset – ***Think Like An Attacker***

- Break the system
- Find undocumented APIs and behavior (example: backdoors)
- Learn to read undocumented code
- Learn to disassemble code

Cross boundary conditions

- If documentation says ”*input is an unsigned integer*”, pass a negative number
- Look for input validation checks and type casts and attempt to break them
- Pass huge amounts of data, and look for buffer overflows and Denial Of Service

Read the documentation and then disregard it.

Securing Systems

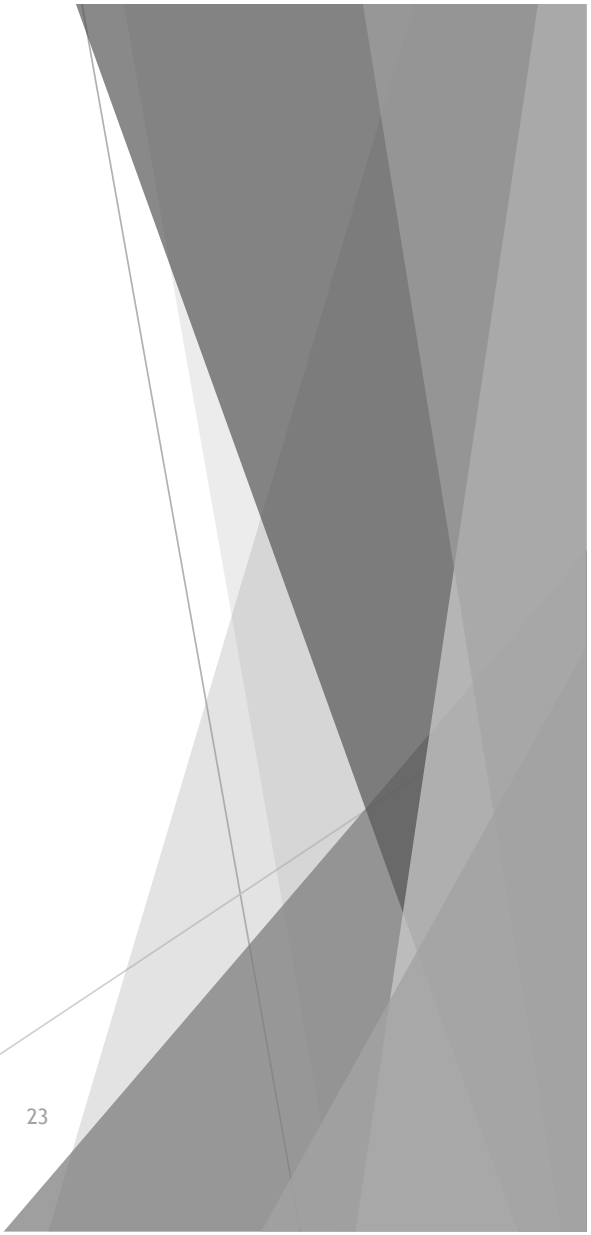
- Secure Development Lifecycles
- Isolated Environments
- Secure Enclaves
- Security Focused Design, Development and Testing Methods

- Slow down an Attacker
- Increase the cost of attack

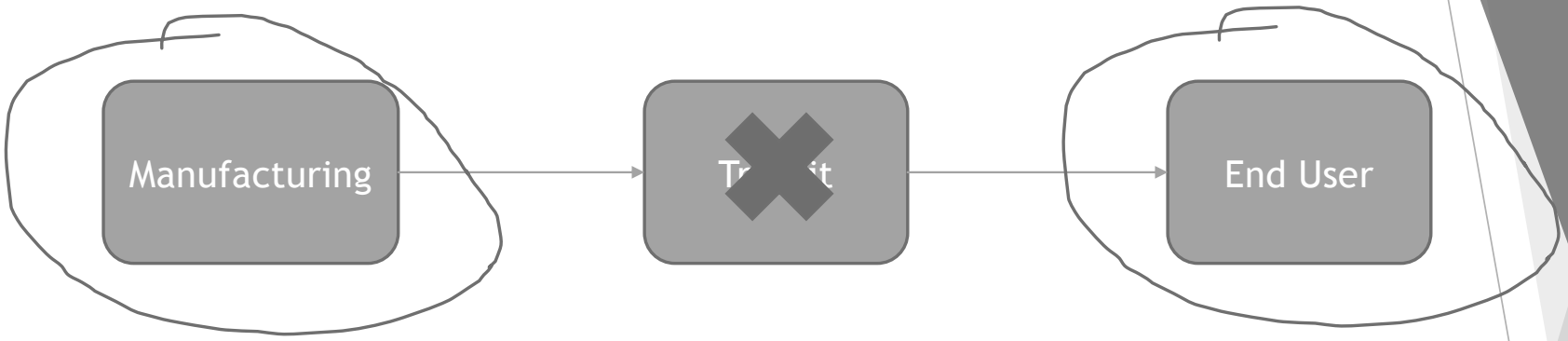
Trust

Security is a goal, a perfect end state… a paradise. Everybody wants it, but you will never actually get there. *You can only try to get as close as you can*.

Trust, however, is something you can measure.



Establishing Trust



(TS/SI/NF) Left: Intercepted packages are opened carefully; Right: A "load station" implants a beacon

Two Primary Concepts

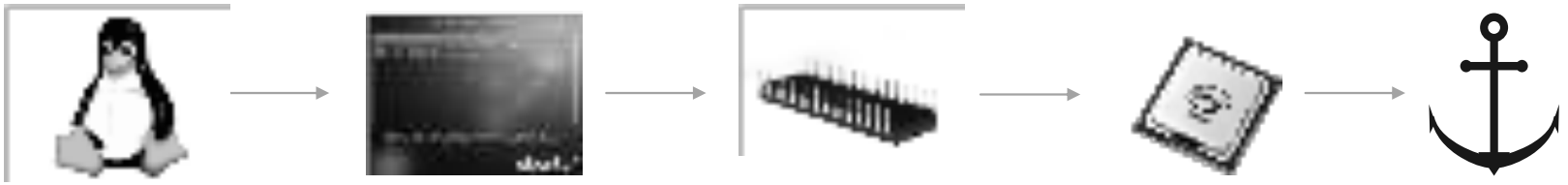
Security Controls

Tamper Evidence

Two Questions

- *How do I know my device is running only the software I think it is?*
- *How do I know my device hasn't been physically altered since it was built?*

Chain of Trust



Linux Kernel

GRUB Bootloader

BIOS (Firmware)

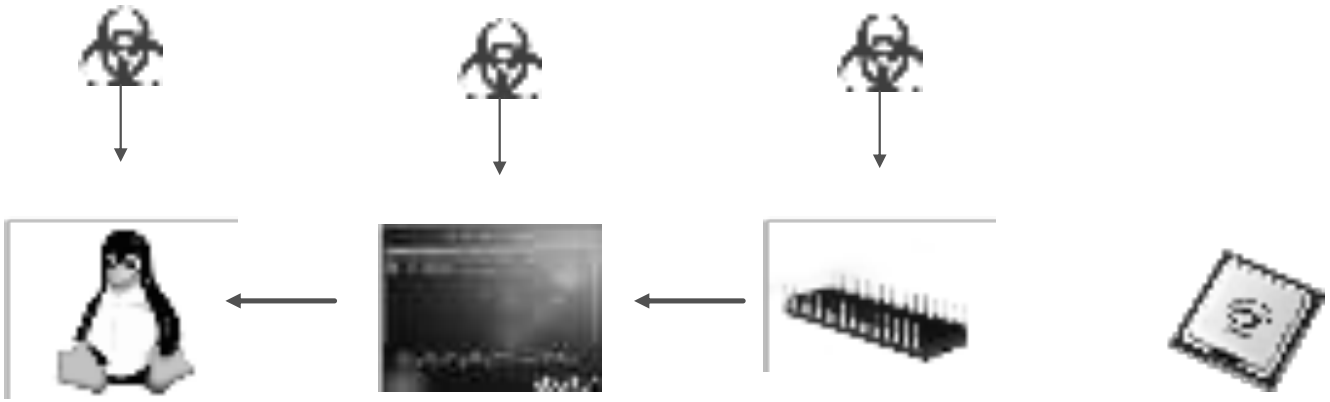
Processor (Intel?)

Root-of-Trust (Trust Anchor)

A small non-tamperable component that must always behave a certain way and can be trusted by a compute engine. Must be in Hardware for complete trust. Public Examples:

- Google Titan
- Microsoft Cerberus (Part of OpenCompute Project)
- Apple T2

Attacking System Integrity



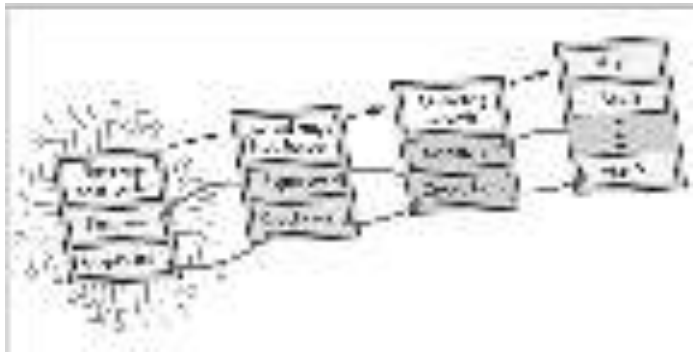
Linux Kernel

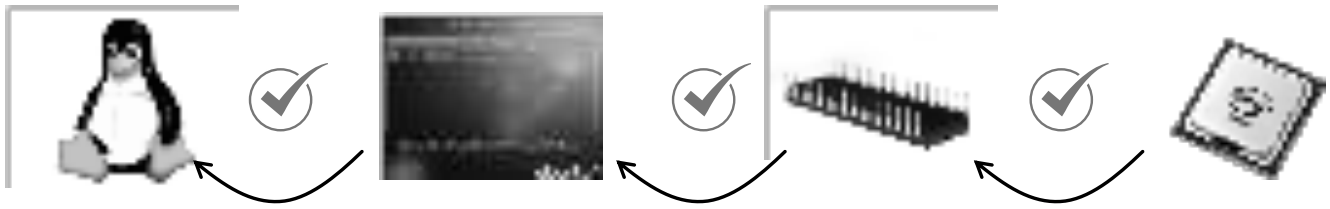
GRUB Bootloader

BIOS (Firmware)

Processor (Intel?)

Secure Boot





Linux Kernel

GRUB Bootloader

BIOS (Firmware)

Processor (Intel?)

Why Secure Boot Matters:

https://blog.f-secure.com/cold-boot-attacks/

https://www.welivesecurity.com/2018/09/27/lojax-first-uefi-rootkit-found-wild-courtesy-sednit-group/

Secure Boot is a Control

Once its complete how do you know it actually happened?


“Relax. Everything is fine.”
Integrity Visibility

Would you trust a device to tell you it’s trusted?

Trusted Computing Group (TCG) Trusted Platform Module 2.0

https://www.trustedcomputinggroup.org

TPM Platform Control Register

- A measurement is a hash stored in a register on a hardware TPM device
- Measurements are held in a secure location that cannot be spoofed
- Measurements are performed using cryptographic hashes – SHA1 and SHA256 in TPM 2.0

SHA functions are one way functions that cannot be reversed

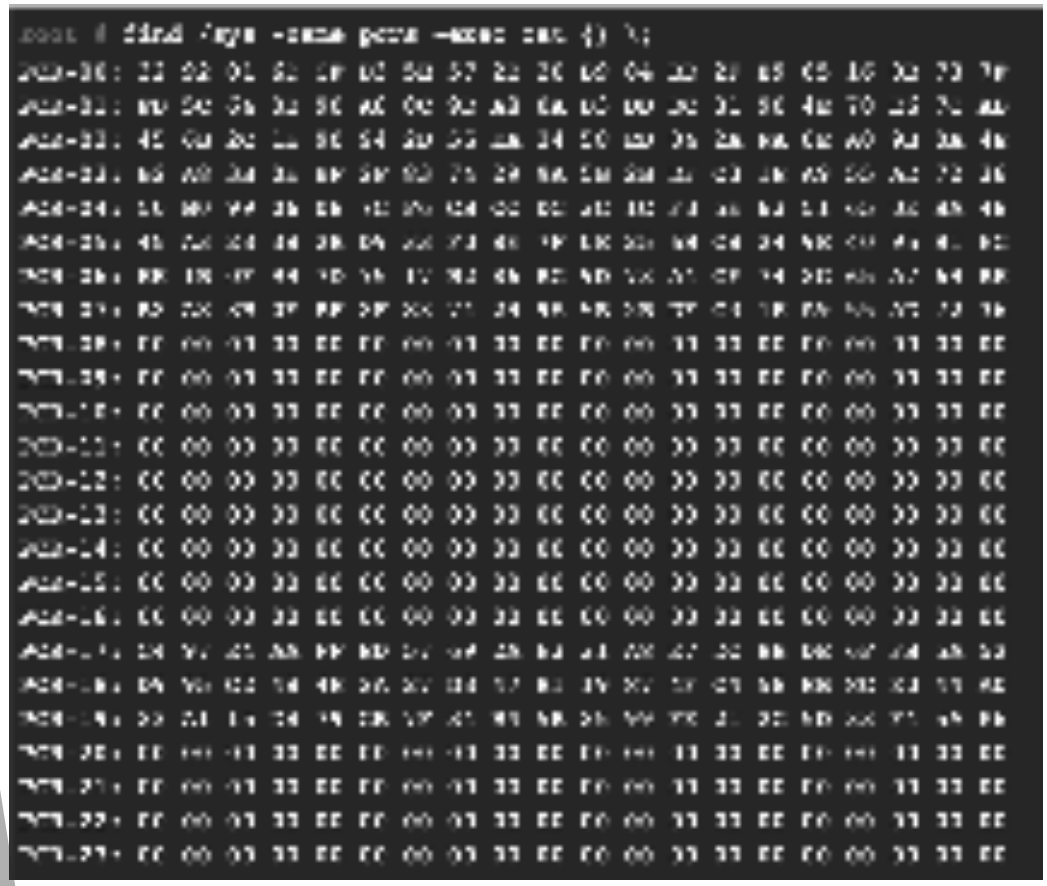
- Measurements can be of code, data structures, configuration, or anything loaded in memory
- Measurements must be done in a sequence and a PCR is extended with subsequent measurements

PCR new value = Digest of (PCR old value || data to extend)

- Not only must measurements match but measurements **must** be made in a particular order.
- On Linux the measurements can be pulled out from sysfs filesystem
- OS / Applications can *bind* operations to specific values of the PCRs thereby validating integrity of the system prior to operation

Example: Decrypt file system if the PCR X value matches a particular hash

TPM Platform Control Registers



/sys/class/tpm/tpm0/device/pcrs

PCR Number	Allocation
0	BIOS
1	BIOS configuration
2	Option ROMs
3	Option ROM configuration
4	MBR (master boot record)
5	MBR configuration
6	State transitions and wake events
7	Platform manufacturer specific measurements
8–15	Static operating system
16	Debug
23	Application support

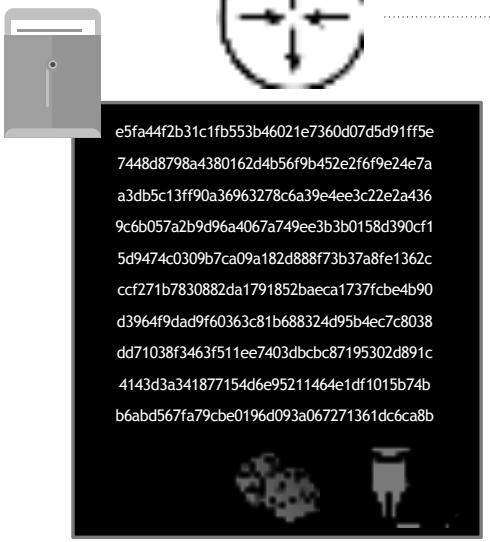
Ok, so we have PCRs. How do we use them to prove Integrity?

Attestation is a Cryptographic Proof of Software State

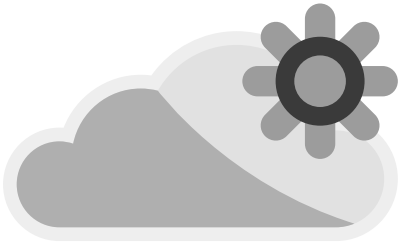
- PCRs cannot be rolled back, measurements cannot be undone
- TPM has an Attestation Integrity Key (AIK) private - public key pair
- Private Key is stored in non-accessible, non-tamperable hardware and Public Key is available for verification
- *Known Good Values* for the software and configuration must be available that provide proof of non-tampering.

Secure Quote Process

1) Service requests quote with unique nonce value



2) Device gathers measurements with nonce, creates attested quote



Integrity Verification Service

3) Service validates:

- PCRs came from the specific device and the PCR values are not tampered with
- Nonce Value
- PCR Measurements vs Known Good Values

Use Cases for PCR Based Attestation

- Integrity of a Device Prior to connecting to the Intranet
- Verification of state (such as non-reboot) prior to performing financial transactions

Free Under Open Access

ISBN:

9781430265849 (online)
9781430265832 (print)

https://link.springer.com/book/10.1007/978-1-4302-6584-9

Summary

- **Securing a System is Hard.**
  Lets Make it Expensive for an Attacker
- **Follow Standard** Secure Development Lifecycle Practices
- Trying to **Secure a System by Patching** Constantly
  is a Cat-and-Mouse game
- There are exploits we don't know about
  so how do we **deal with "unknown unknowns"**
- **Build *Trusted Systems***, with Trust starting at
  a Hardware Anchored Root of Trust
- Establish **Trust**, Validate and Measure **Integrity**
- **Build Auditable Systems** with non-tamperable logs
  – No attack must go undetected

Call To Action