Open Infrastructure Security Research and Engineering

Shawn C

#whois

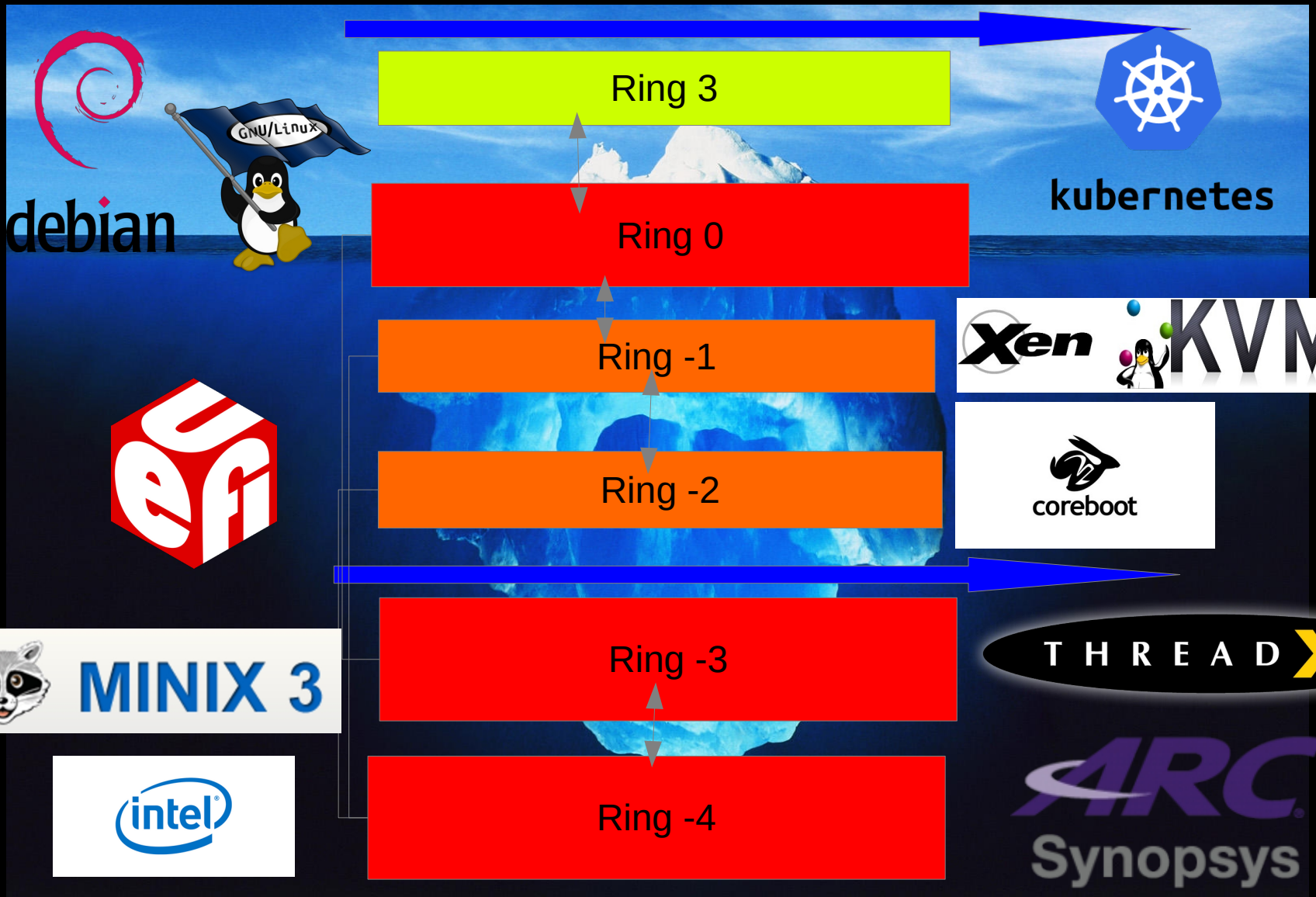
* Shawn C[a.k.a "citypw"]
* Day job at Hardened Vault
    * Open source security consulting
* Just another random security researcher
* Free/libre SW/FW/HW enthusiasts
* Member of EFF/FSF/FSFE/RISC-V foundation
* L0rd c0mmander of HardenedLinux
* “Do you know the way to | Vault | San Jose?”

#cat /proc/agenda

* The current situation of real-world problem
* Technical solution for main quest
* Side quest matters as a whole
* Major players in the eco-system
* How CSPs adapt into the game

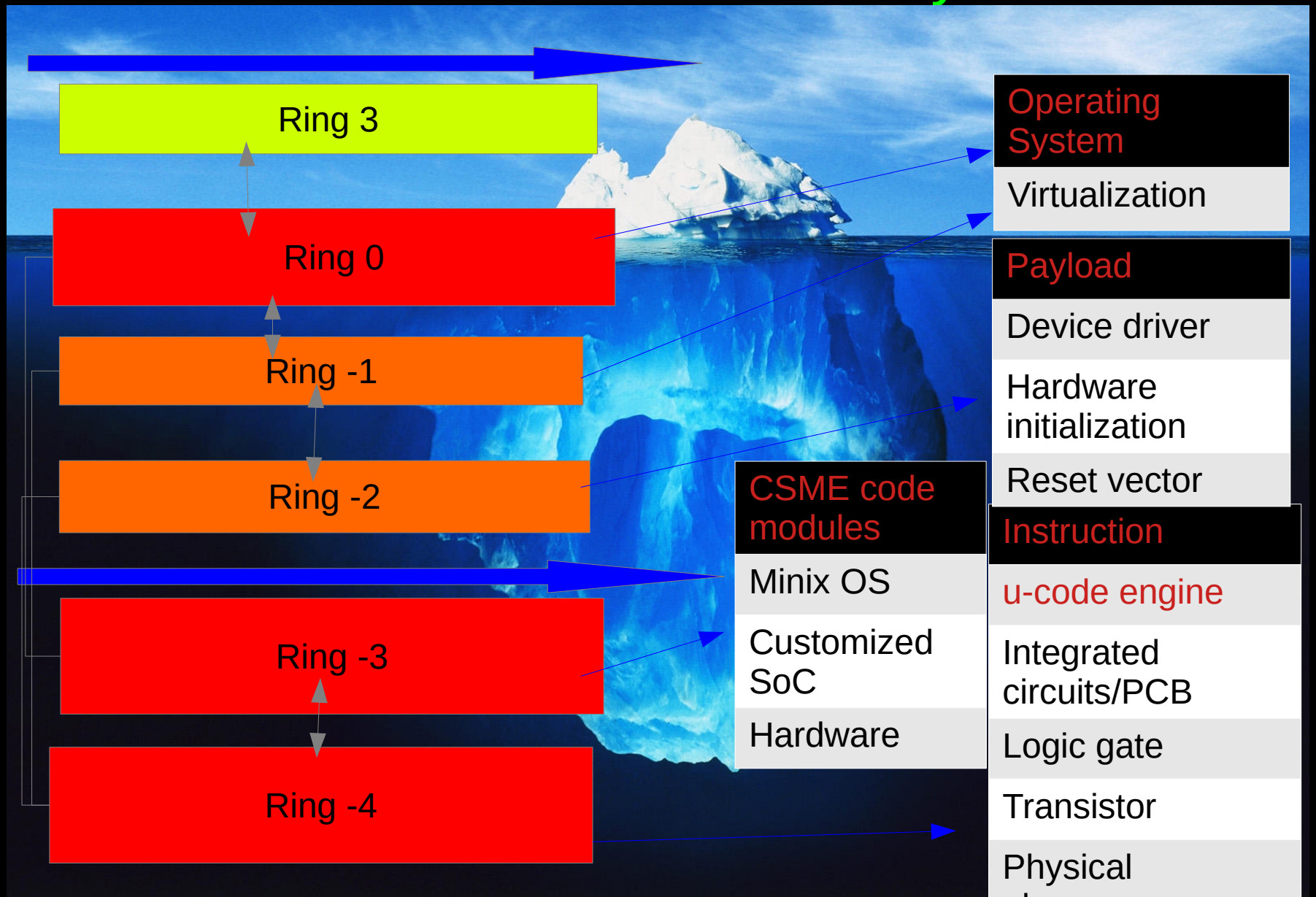
#Status

* What does a modern server/workstation look like?



#Deep “state”s

* RING 0 is **not** the CORE anymore



#Linux kernel vulnerablity Statistics – Jan 2016 – Jan 2021

Sources	CVEs	Public exploit works (except DoS)?
MITRE	1470	<10
Ubuntu security tracker	1100	<10

Most of the public exploits works on PaX/GRsecurity are from the variants of Meltdown/Spectre which is the transient execution issues in CPU.

#Linux kernel hardening solution

Tehnical Solution	Vendor support	Kernel version	Pros	Cons	Use-case for CSPs
PaX/GRsecurity	Open Source Security .Inc	4.14 to 5.4 for stable, >=5.8 for testing	* Cutting-edge defense techniques * More stable than LTS * More well-maintained backport	Expensive	For high-end customer or critical mission node
LKRG	N/A	4.14 to 5.x, tested on Debian mostly	* Free * Hardcoded policy targeted the popular exploits * Post-exploitation detection/prevention	* The design meant to be bypassed * Lack of regression tests and QA	Can be integrated with the current VM compliance

#Virtualization security

* Share the mitigation with kernel
* More importantly, userspace/QEMU:
    ** Attack surfaces reduction
    ** Sandboxing

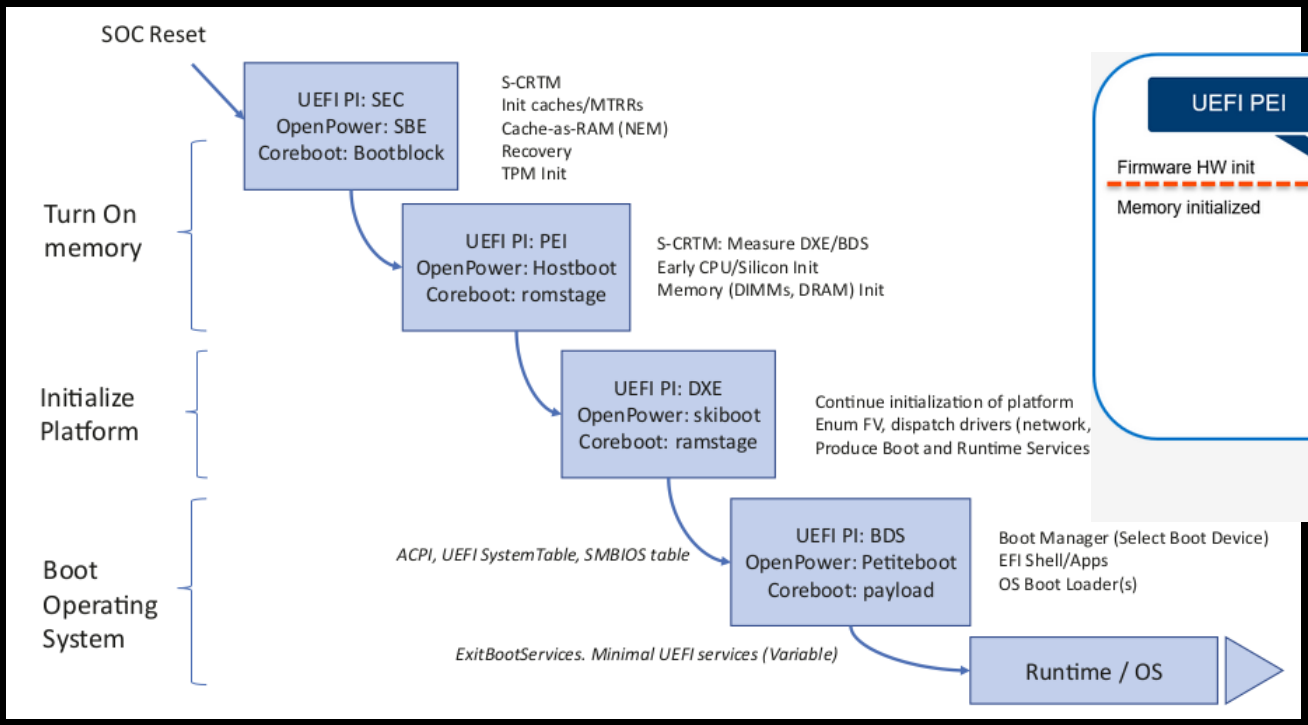
An example of real-life exploit is easy to mitigate by distro-level hardening:

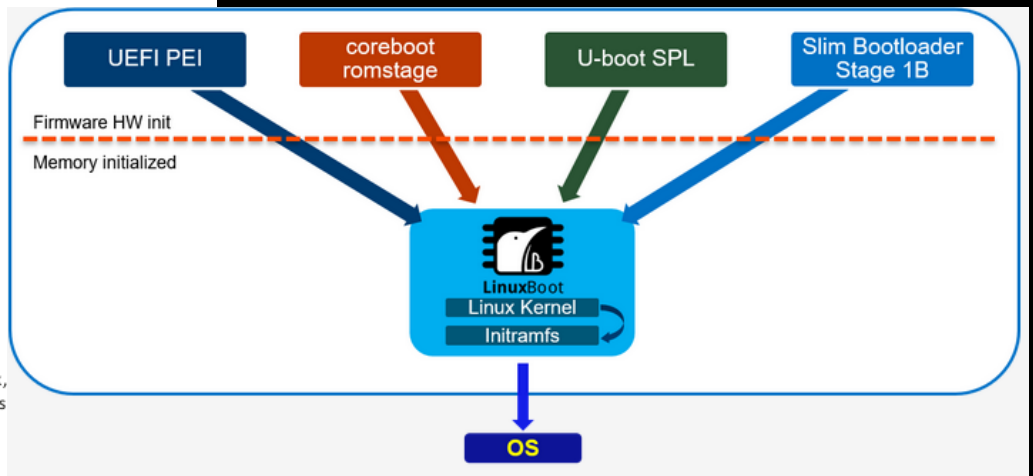
```
QEMU CVE-2020-14364 vulnerability analysis (including POC demonstration) Aug 2020

Qi Anxin Code Security Lab researchers discovered six vulnerabilities for Red Hat ( CVE-2020-14364 , CVE-2020-10756 , CVE-2020-12829 , CVE-2020-14415 , CVE-2020-15863 and CVE-2020-16092 ), where CVE-2020-14364 is assessed as having "significant impact". The researcher immediately reported to Red Hat and assisted it in fixing the vulnerability. This article mainly analyzes CVE-2020-14364, hoping to give you some inspiration.
```

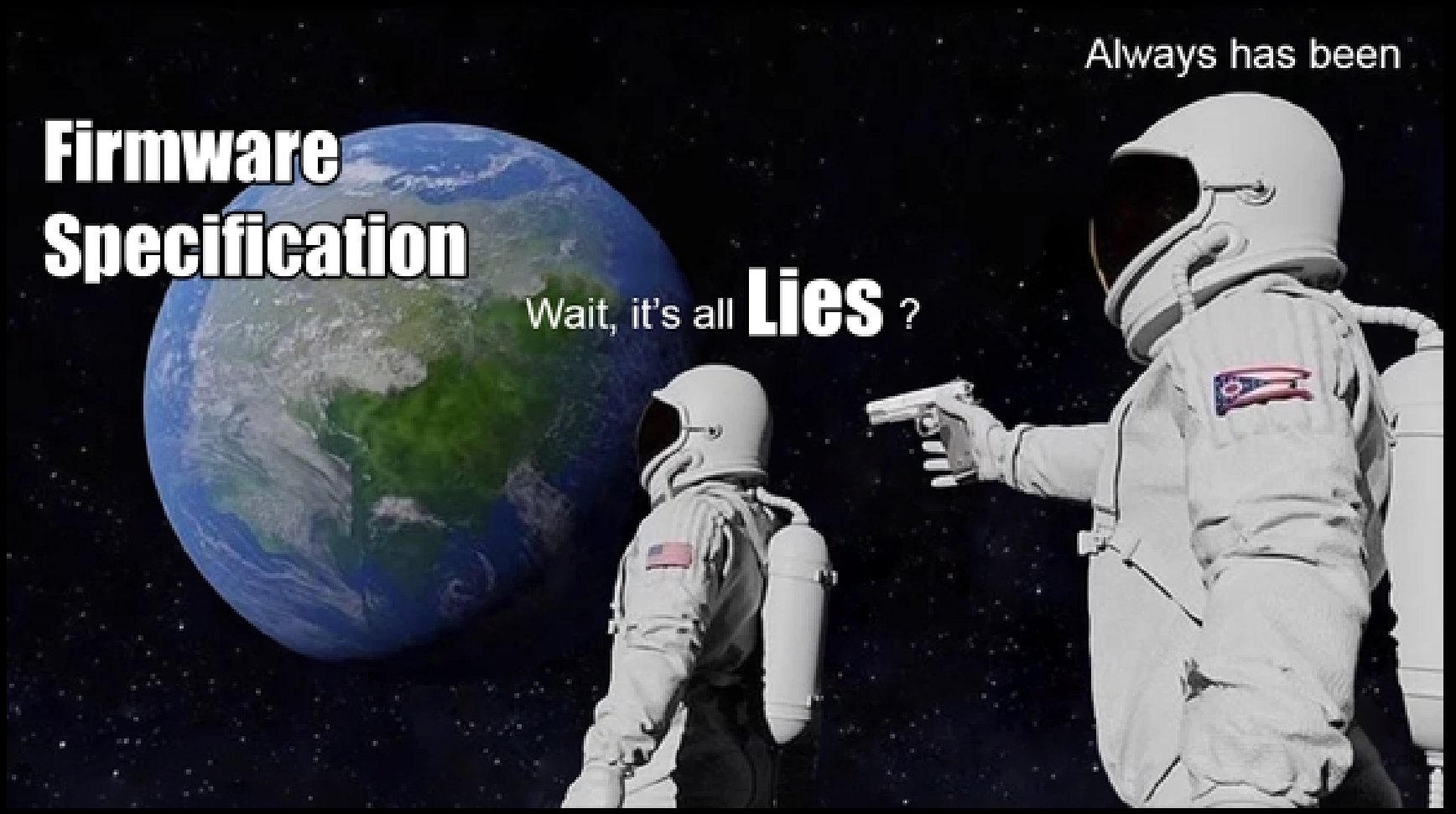
#Firmware security

* Heart of trusted computing
  ** Verifiedboot
  ** Measuredboot





#UEFI is not the only way



Trustworthy? Trusted? Treacherous?

RA protocol

PCR17

Decentralized clou

Intel

* CBnT

SINT

PCONF

MLE

ACM

CRTM

coreboot/OEM

OS

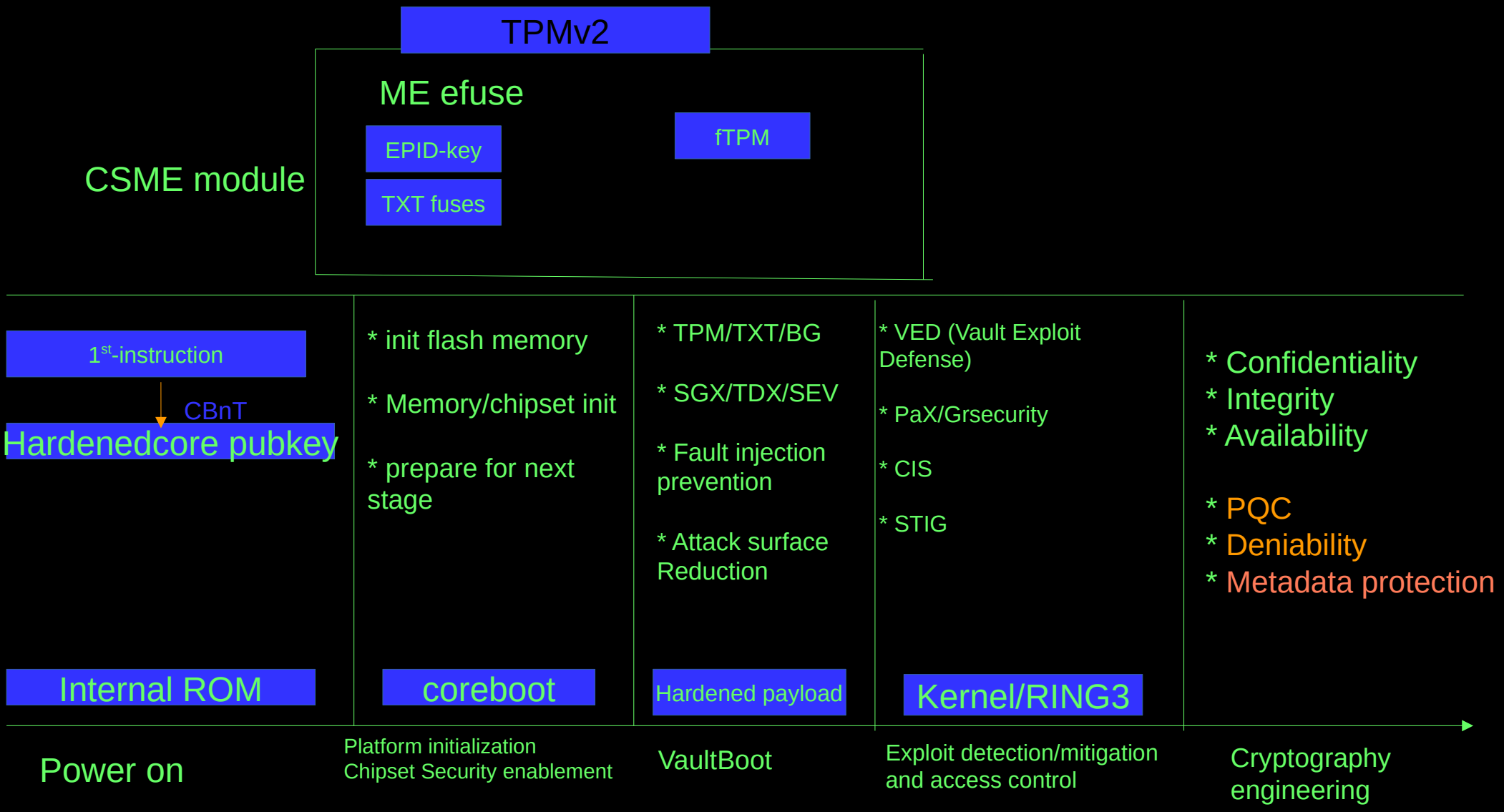
OEM/Manufacture

* e-fuses
* CBnT: BACMs + SINT for servers
* ME/SPS modules
* RoT

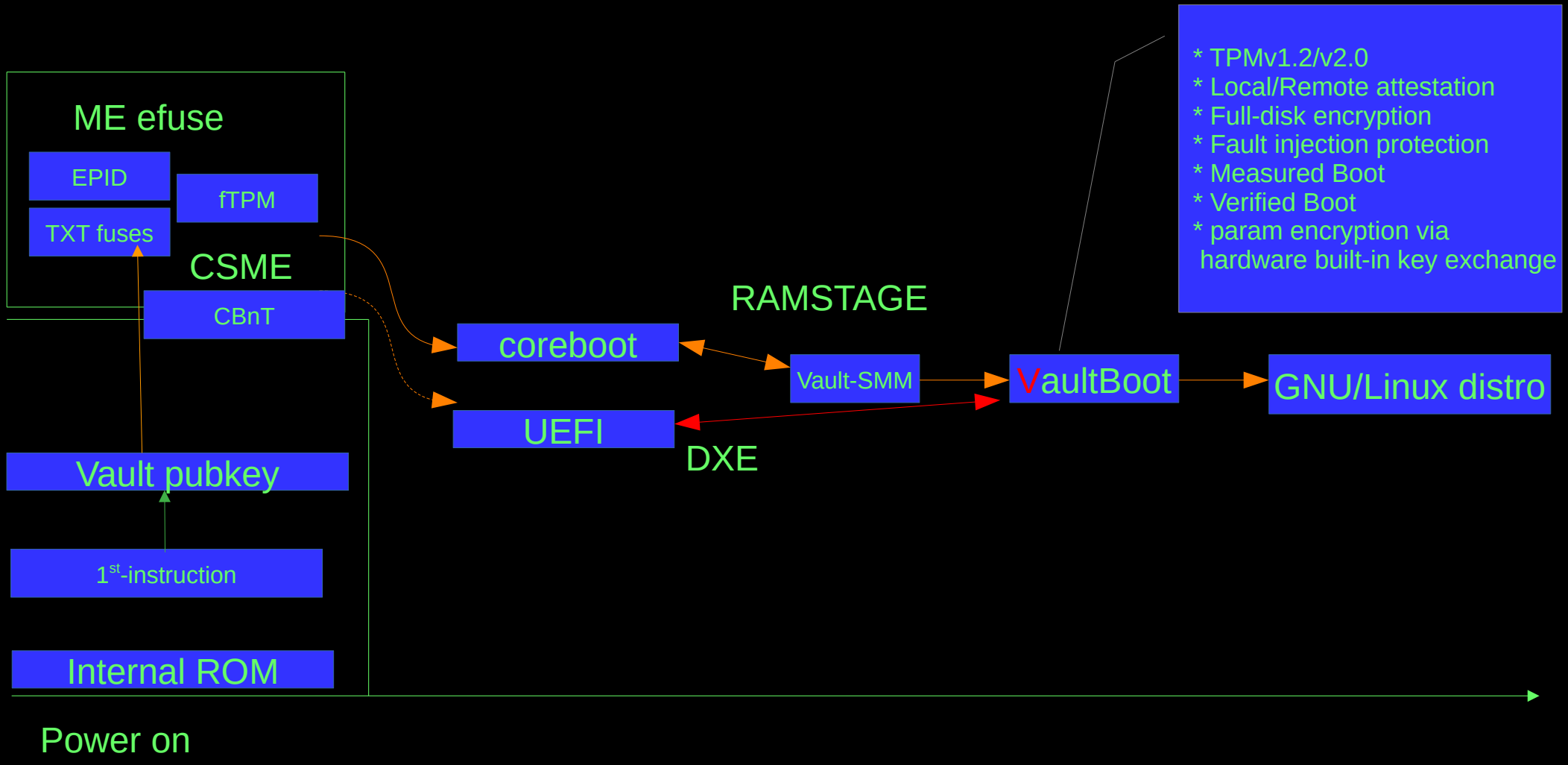
* u-code verify the signature of BACM
* TXT/tboot: SINT
* TPMv2 policy
* CoT

* Hardening based on PaX/Grsecurity
* PCR17

Core of the * cloud?



Core of the * cloud?



HARDENED VAULT

#Invincible devil

* Intel CSME
* A lot of CSME "apps" based on it
* Changed a lot since v11
* Can’t be disabled until HAP/altmedisable bit disclosured that we know how BIGBRO does about Intel CSME for their own defense

#Ops Hunting the Shadow

Intel CSME	Before	After
Neutralization	Boot every 30 mins	Minimizing it by unsigned code removal
Kill switch	N/A	Found HAP/Altmedisable bit
Internal fabric	No access without NDA	More knowledge of PCH/IOSF/VISA
CSME security	RTOS without basic mitigation	Modern mitigation and bug hunting process
Official material	No public release	Intel released security white paper for CSME v14/v15

#The dark side of x86

NDA ALERT !

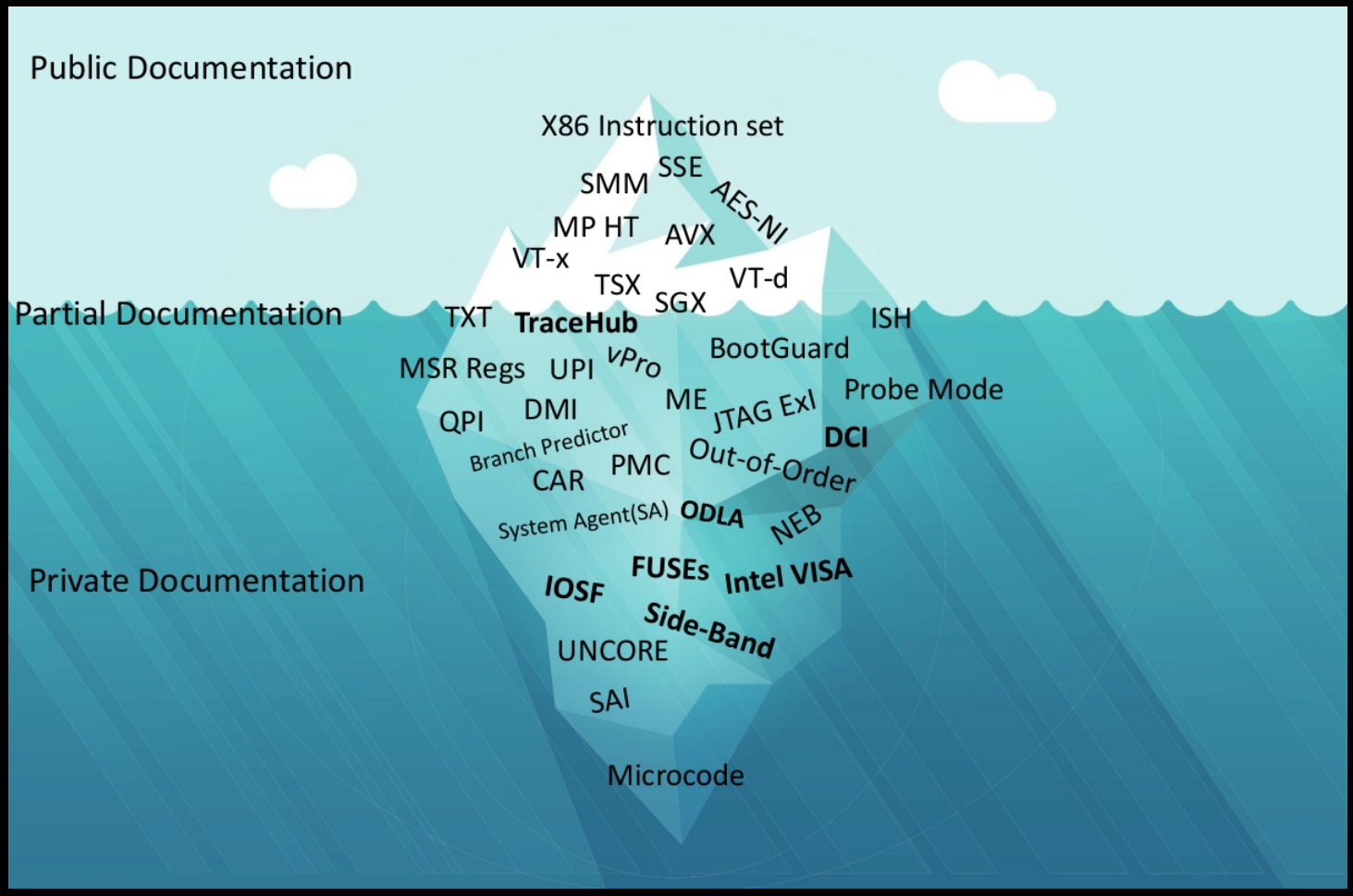
DAL

Debugger

TXT/PFR

FSP

Microcode



Hardening level and compliance

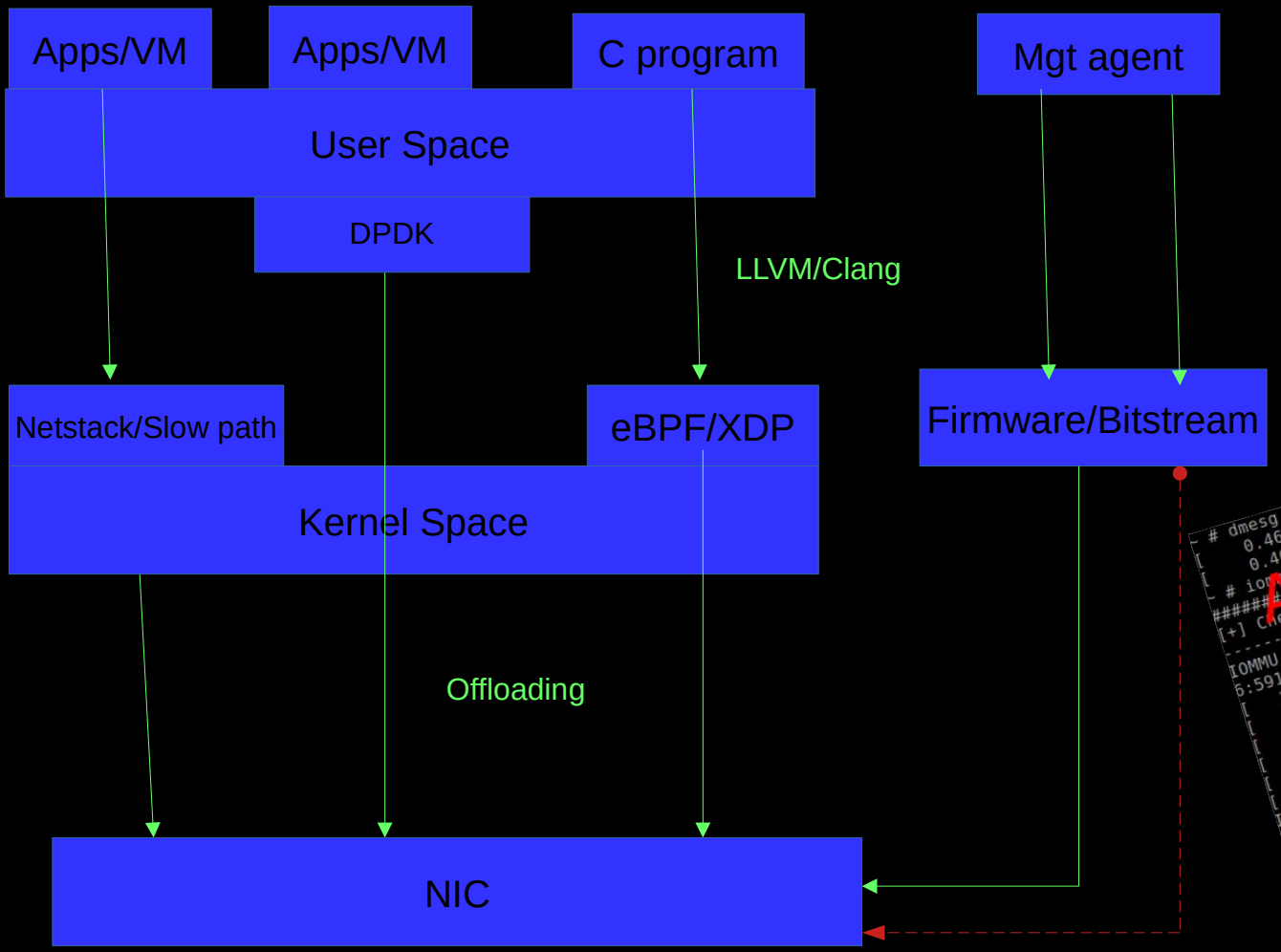
Level	Type	**Security Hardening Profile**
Critical	Physical machine	Situational hardening for Firmware/Kernel + Enclave + Remote attestation + Harbian audit Level 2
	Virtual machine(not recommended)	Situational hardening for kernel + Hypervisor-based Remote attestation + Harbian audit Level 2 + Optional(firmware mirror integrity verification)
Important	Physical machine	Standard hardening for Firmware + Situational hardening for kernel + Enclave + Remote attestation + Harbian audit Level 1
	Virtual machine	Situational hardening for kernel + Hypervisor-based Remote attestation + Harbian audit Level 1
Normal	Physical machine	Harbian audit Level 1 + Optional(Enclave + Remote attestation)
	Virtual machine	Harbian audit Level 1 + Optional(Enclave + Remote attestation)

#Extra notes;-)

* Neutralized ME affected remote attestation via SGX?
* Think harder about your situational hardening solution before ask OEM write your pk
* You don’t need Intel Bootguard if the data center could take care of physical security. Besides, open source provisioning tool is prefered.
* Big clouds(AWS/Google/Facebook/BAT3H/etc) should do the situational hardening for their core infrastructure. Otherwise it might become someone else’s computer;-)
* Masters of Pwn killed by PaX/Grsecurity

#Case study – SmartNIC for data center

* Peripheral device
* Signature



#Side quest: what constitutes an average case

* RISC-V
* Fuzzing as a Service
* Hardware manufacture and supply chain

#RISC-V security

* An open standard ISA(instruction set architecture)

* NDA or not it’s only the vendor’s decision

* The eco-system is currently good for microcontroller but not friendly(yet) to GNU/Linux

RISC-V is an open ISA can provide more options to build our free/libre firmware/hardware solution. RISC-V may get us rid of "too many blobs" issues, which is nearly impossible to be resolved in x86. RISC-V is in very early phase and there are tons of work haven't done yet. Priviledged ISA spec is still in "draft" stage(v1.10 for now) and security spec is not open for the public yet(maybe in 2018?). The software/firmware side of RISC-V community is growing so fast in past two years, e.g: glibc/binutils got merged already, upstreaming it into linux kernel might still need a couple of months, coreboot can support it partially. In some aspect, RISC-V may have better opportunity to build-security-in in the 1st place. Both x86 and ARM took years to bring those security mitigations into the hardware support. Plz note that those security mitigation also takes time to being utilized by GNU/Linux. Let's quick review the linux kernel defense as an exmaple:

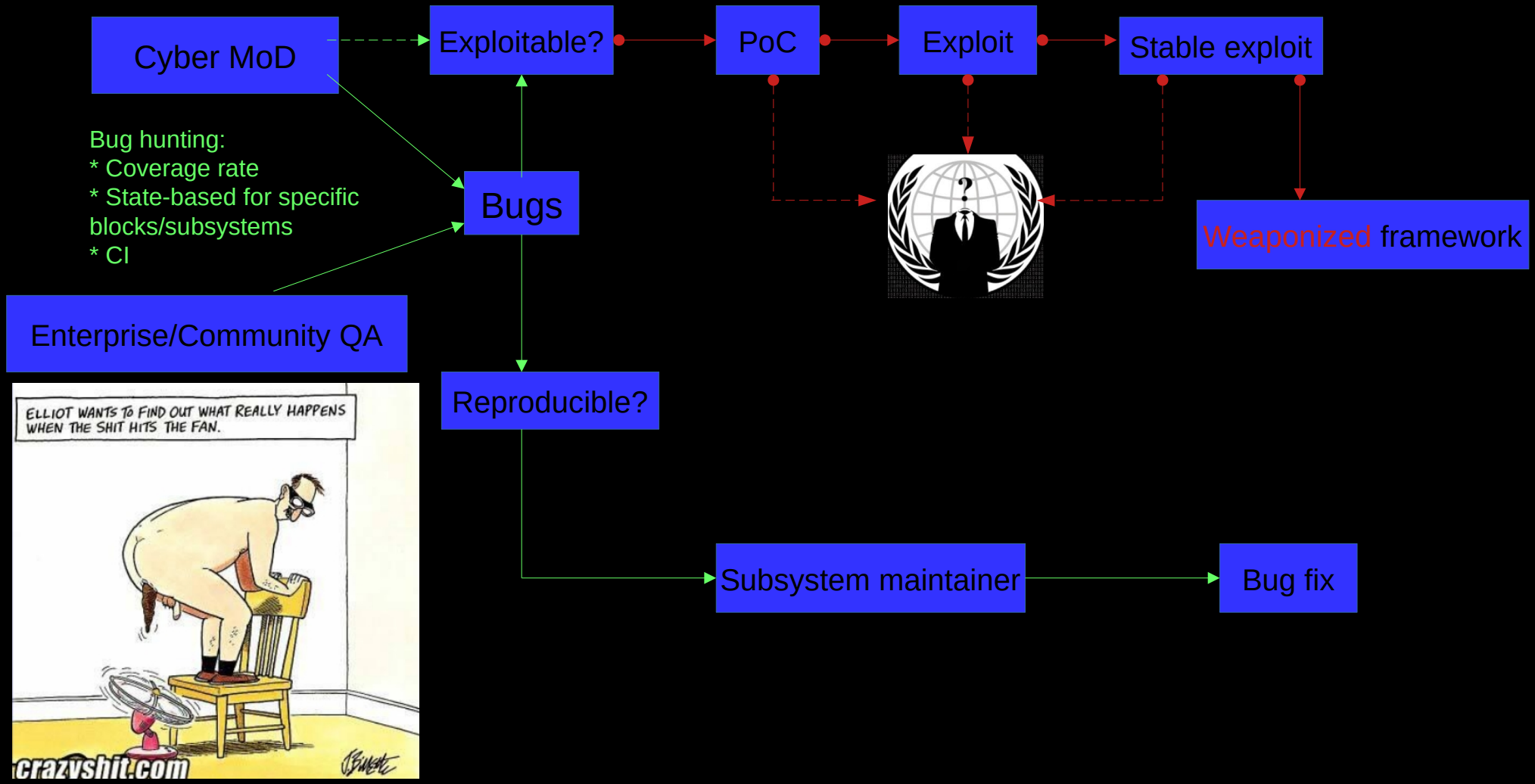
Security defense	PaX/Grsecurity	x86	armv7	arm64	coreboot
Non-executable stack	PAGEEXEC(2000)	NX bit(2004)	XN bit(2011)	UXN(2012)	?
ret2usr protection	KERNEXEC(2004)	SMEP(2011)	PXN(2014)	PXN(2012)	SMRR(year?)
ret2usr protection	UDEREF(2007)	SMAP(2014)	N/A	PAN(2019?)	
CFI	RAP(2015)	CET(2018?)	N/A	PA(2019?)	

#Fuzzing as a Service

* Fuzzing is the term somewhat equivocal

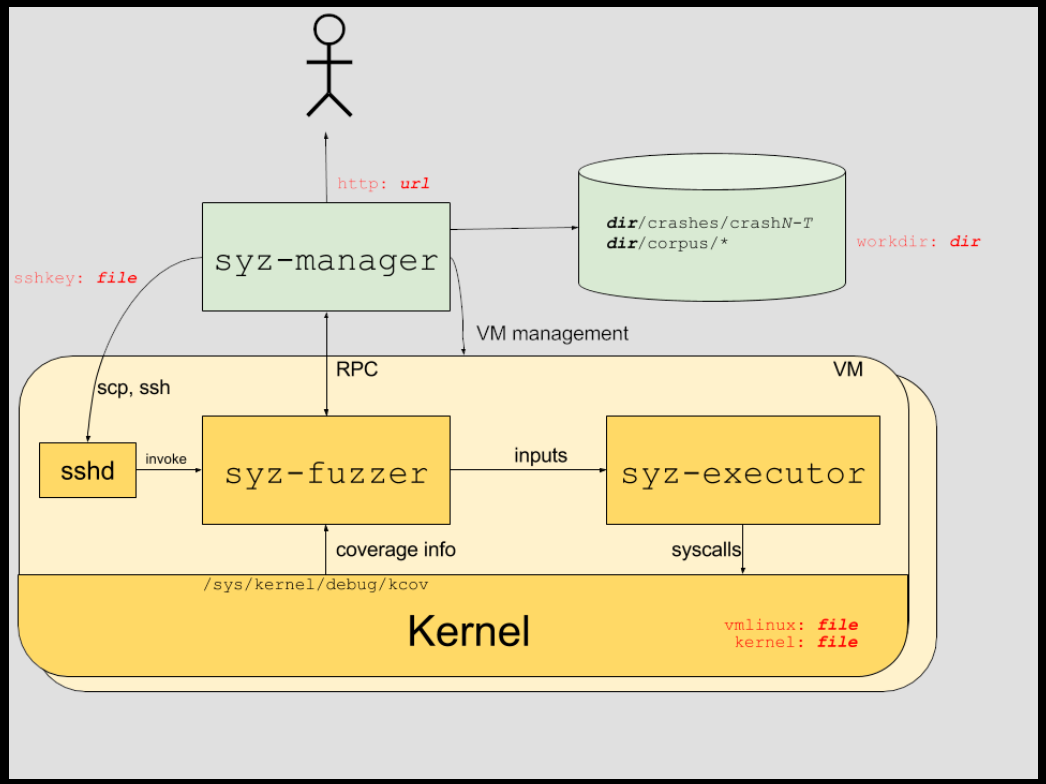
** Security-oriented approach

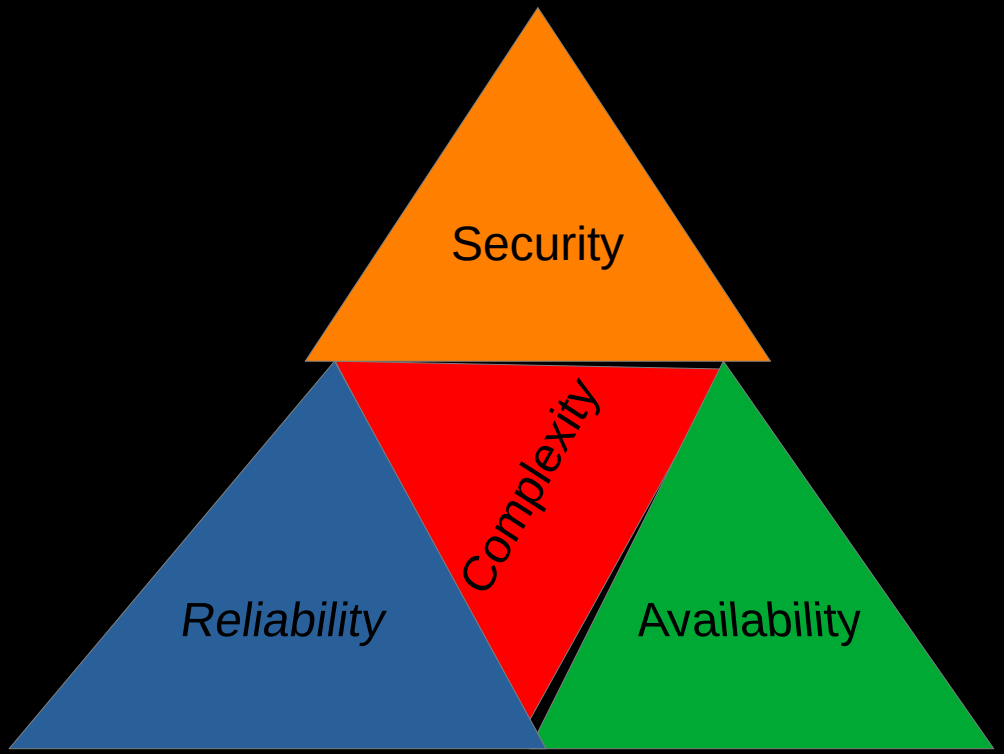
** QA-oriented approach



#Fuzzing as a Service

* Why QA/fuzzing still matters under cutting-edge defense?
* Your business is high rely on the consequences of complexity





#Supply chain threats

* The solid proof from Bloomberg's story won't matter only because we know the threat is real.

"We know it's possible both because the NSA has apparently done it, but also because I'VE done it."

Bloomberg

The Long Hack: How China Exploited a U.S. Tech Supplier

For years, U.S. investigators found tampering in products made by Super Micro Computer Inc. The company says it was never told. Neither was the public.

By Jordan Robertson and Michael Riley
February 12, 2021, 5:00 AM

In 2010, the U.S. Department of Defense found thousands of its computer servers sending military network data to China–the result of code hidden in chips that handled the machines' startup process.

Supermicro audit finds no evidence of back doors

Written by
Gareth Halfacree

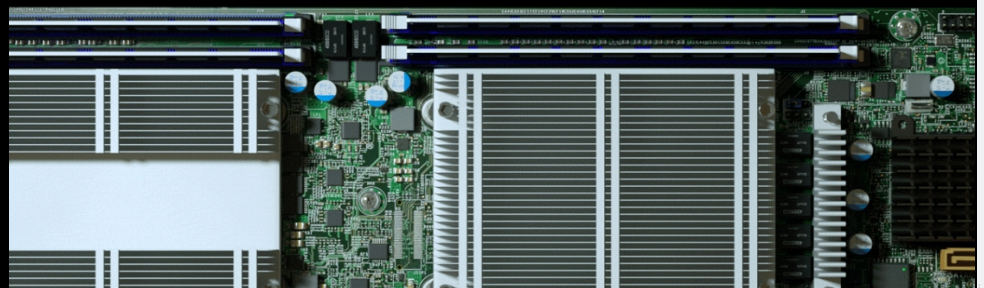
December 12, 2018 | 10:39

Tags: #charles-liang #david-weigand #hardware-back-door #insecurity #motherboards #raju-penumatcha #security #server #supermicro #supply-chain

Companies: #amazon #apple #bloomberg #super-micro-computer

The Big Hack: How China Used a Tiny Chip to Infiltrate U.S. Companies

The attack by Chinese spies reached almost 30 U.S. companies, including Amazon and Apple, by compromising America's technology supply chain, according to extensive interviews with government and corporate sources.

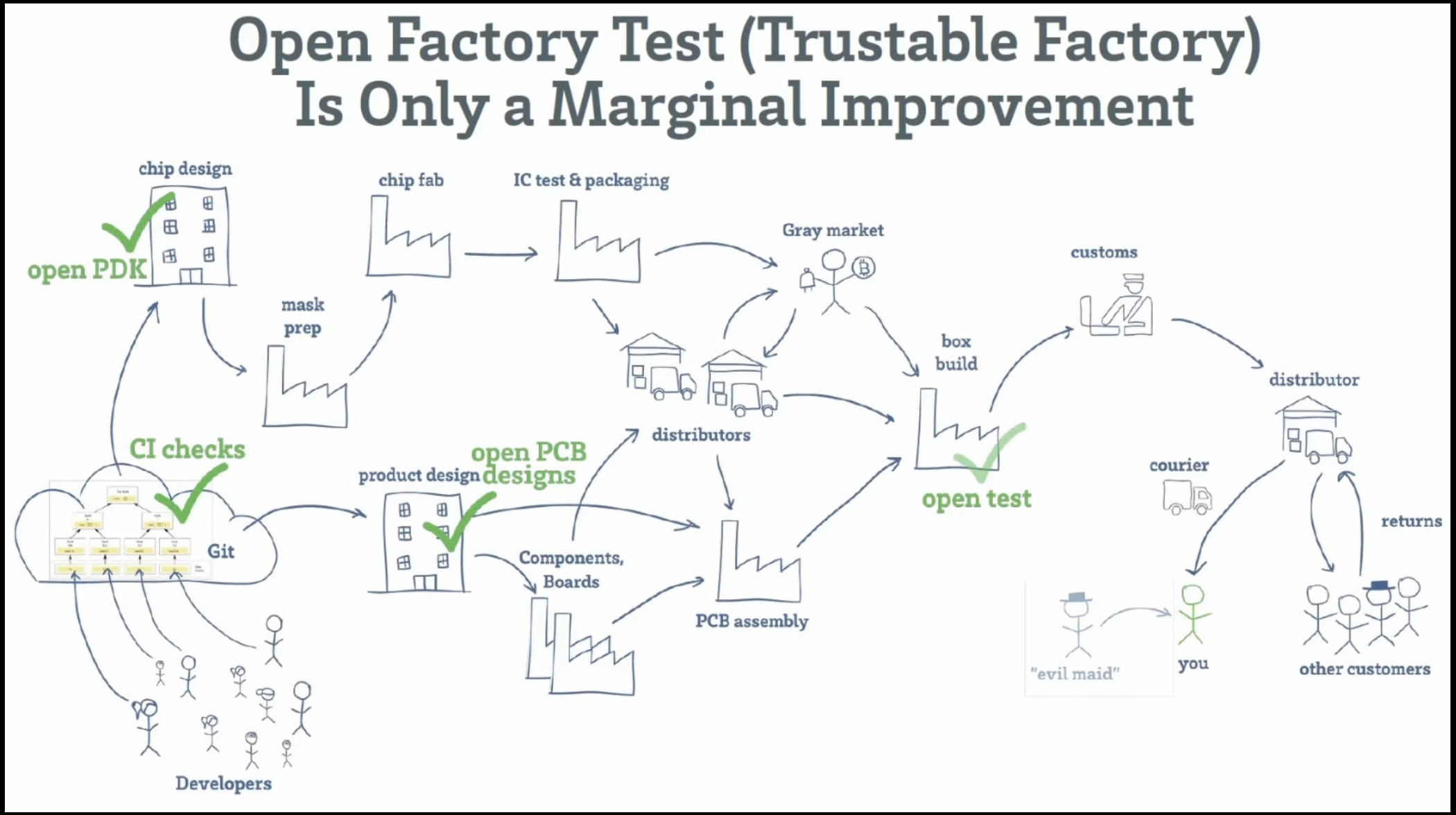


Apple deleted server supplier after finding infected firmware in servers [Updated]

Report: Siri, internal development servers affected by fake firmware patch.

SEAN GALLAGHER - 2/24/2017, 4:49 PM

#Complexity in supply chain



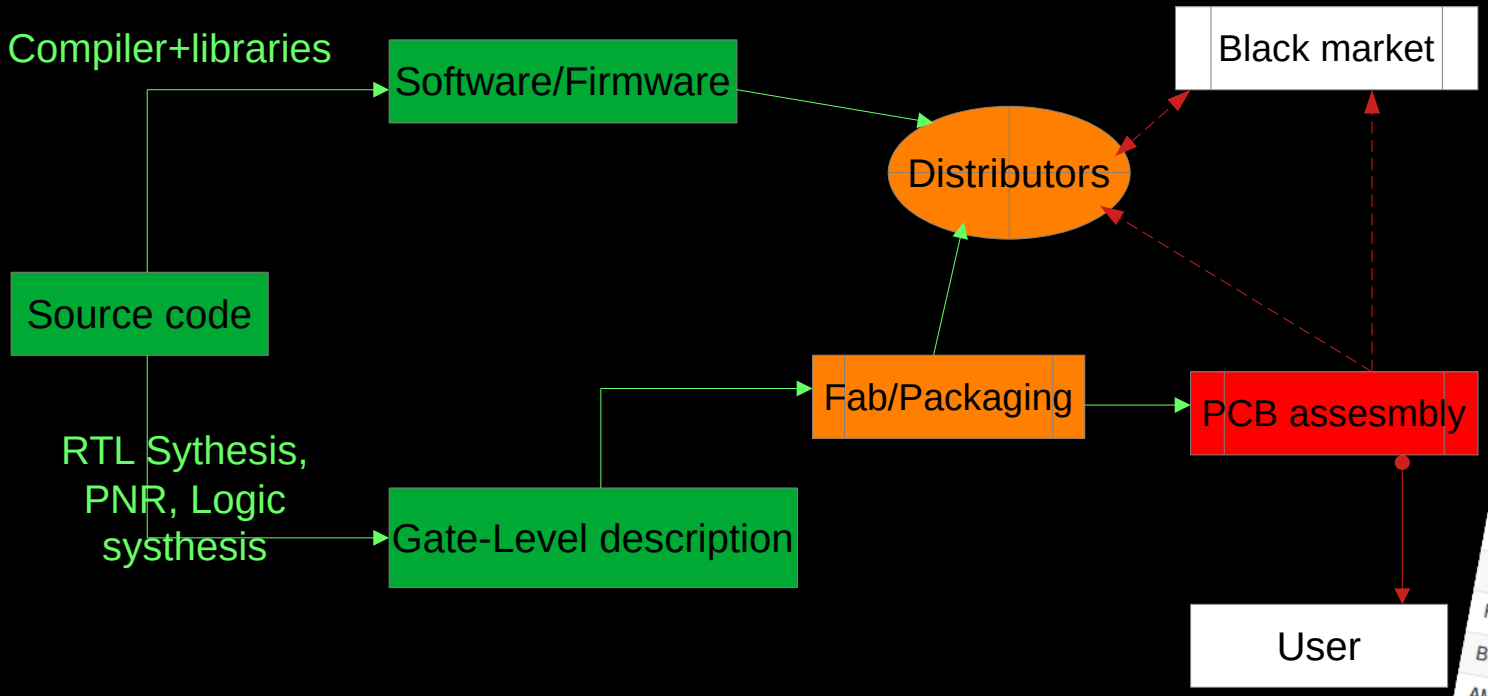
#Complexity in supply chain

* Real-life example analysis: CheaPCB

** Less $10

** Must-TODO even for regional supply chain: EU(Estonia/Sweden/Germany/Croatia, etc), Asia(Shenzhen/Hongkong/Vietnam), Americas(New Jersey/Mexico city), etc.

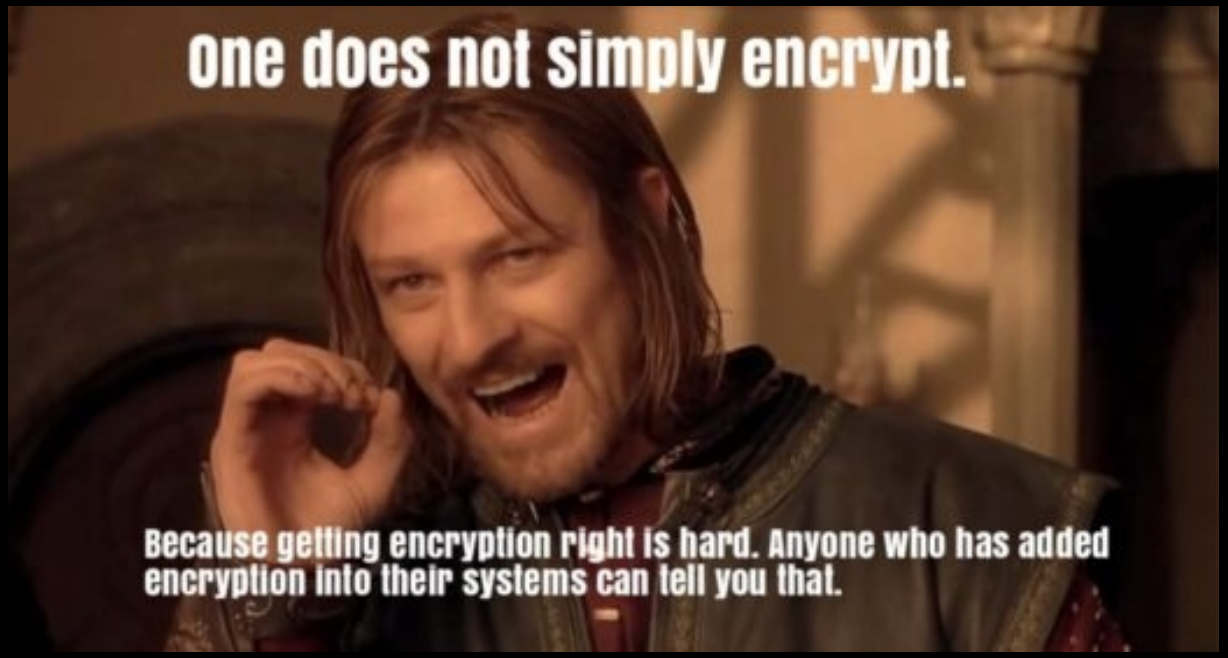
** Tame the “beast” of complexity for server/workstation manufacture?



device	package	quantit
Capacitor 10uF	0603	2
Capacitor 10nF	0603	1
Capacitor 100nF	0603	8
Capacitor 22pF	0603	2
Schottky diodes 1N5817WS	SOD-323	1
light-emitting diode Red	0603	1
light-emitting diode Blue	0603	1
light-emitting diode Yellow	0603	1
Fuse 500mA	0805	1
Fuse 100mA	0805	1
USB C Receptacle	Palconn UTC16-G	1
2×4 2.54mm PinHeader		1
1×25 2.54mm PinHeader		2
Transistor S9012	SOT-23	2
Resistor 1K	0603	1
Resistor 10K	0603	6
Resistor 5.1K	0603	3
Resistor 22	0603	2
Resistor 1K5	0603	2
Button KMR241G		1
AMS1117-3.3	SOT-89	1
STM32F103C8T6	LQFP-48	1
Crystal 8MHz	SMD_3225	1

#Weakest link: The twin

* System security/hardening won't work well without cryptography



#Case: Survive in post-LadderLeak

* Issue: Non-deterministic ECDSA

  ** Neither can't keep the k-value secret nor "maliciously" repeat it

  ** Worst-case: Cost 300k USD for P-224

Table 1: Comparison with the previous records of solutions to the hidden number problem with small nonce leakages. Each row corresponds to the size of group order in which the problem is instantiated. Each column corresponds to the *maximum* number of leaked nonce bits per signature. Citations in green (resp. purple) use Bleichenbacher's approach (resp. lattice attacks).

	< 1	1	2	3	4
256-bit	–	–	[64]	[64]	[45, 57, 58, 70]
192-bit	This work	This work	–	–	–
160-bit	This work	[6, 14], this work (less data)	[13], [41]	[47]	–

Table 3: Summary of the experimental results. The "Thread" columns are of the format #shared-memory threads × #distributed-memory nodes. The "Recovered MSBs" were computed with respect to the relative error from the actual secret sk, i.e., $\lfloor \ell - \log |sk - w| \rfloor$, where w is an estimated secret key and ℓ is the bit-length of group size. We remark that the large body of memory consumption is due to the parallelization overhead, and in fact, the per-thread RAM usages were below 6GB in the collision search phase and 32GB in FFT, respectively.

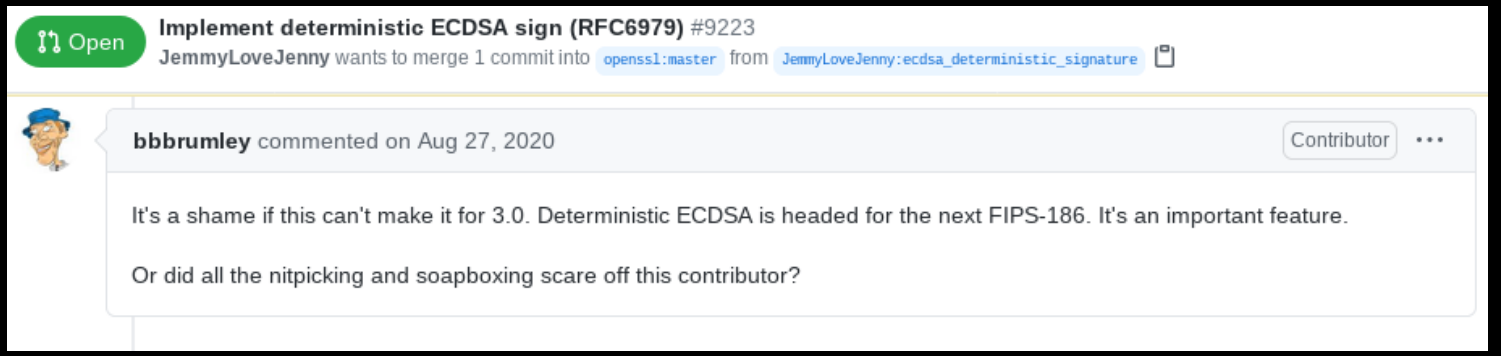
Target	Facility	Cost	Error rate	Input	Output	Thread (Collision)	Time (Collision)	RAM (Collision)	L_{FFT}	Thread (FFT)	Time (FFT)	RAM (FFT)	Peak Height	Max Noise	Recovered MSBs
NIST P-192	AWS EC2	$16,429	0	2^{29}	2^{27}	96×24	113h	492GB	2^{38}	128×2	0.5h	4TB	7.28×10^{-4}	4.48×10^{-4}	39
NIST P-192	AWS EC2	$7,870	0.010	2^{35}	2^{30}	96×24	52h	492GB	2^{37}	1	12h	4TB	5.04×10^{-4}	1.55×10^{-4}	39
sect163r1	Cluster	–	0	2^{23}	2^{27}	16×16	7h	80GB	2^{35}	8×8	1h	128GB	4.92×10^{-4}	4.29×10^{-4}	36
sect163r1	Workstation	–	0.027	2^{24}	2^{29}	48	42h	250GB	2^{34}	16	1h	512GB	2.82×10^{-4}	2.21×10^{-4}	35

#Doom to die?

* Long-term solution:

  ** Waiting for the standardization of EdDSA in FIPS-186-5

  ** Deterministic ECDSA implementation ==> It didn’t go well



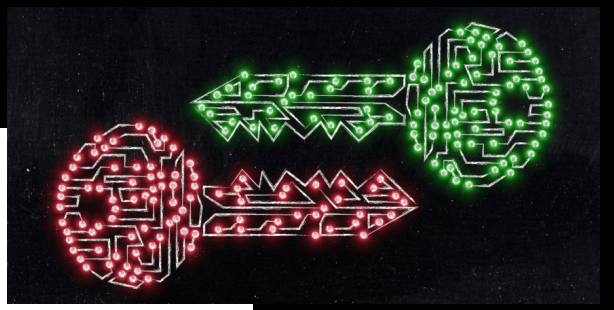
* Short-term audit:

  ** Make sure all of your certificates are using >= P-256 or RSA(another problematic component)

  *** Don’t miss the spots where doesn’t have internet connection;-)

#Conclusion: Twin

* “Dear Shalashaska, stop the fuc* nonsense!”
* Need both of them: Solid and Liquid Snakes



#CSPs comes into play

* Co-op with technical community, including academia research, vendors and open source project development

* Build the security solution for CSPs infrastructure and the customer.

How penguins manage to survive?

#Major players – Hardware RoT

Name	Vendor Support	Description	Open source implementation	Possible use-case for CSPs
Nitro Security Chip	AWS	Trap all I/O to NVRAM. Any operations on firmware must be perform by AWS.	N/A	N/A
Titan	Google cloud platform	Interposer between PCH/BMC and SPI flash	OpenTitan	N/A
Cerberus	Microsoft	Interposer between PCH/BMC and SPI flash	OCP Cerberus	N/A
Converged and Bootguard Technology	Intel	Burn the hash of pubkey into efuse in PCH	Partially, but lack of provioning tool!	Yes
Platform Firmware Resiliency	Intel	PFR CPLD ROM work as preboot environment are manage verify/update/recover ops on firmwares	N/A	Yes
Platform Security Processor	AMD	Similar to PFR	N/A	Yes

#Major players – Next-Gen Firmware Project

Name	Main contibutors	Description	Possible use-case for CSPs
coreboot	9elements, Intel Google, Facebook,3mdeb, HardenedLinux, etc	Popular open source firmware framework	Yes
Linuxboot	Google, Two Signal, etc	Can work as both hardware initialization and payload.	Yes
Tianocore	Intel	Reference model for UEFI	N/A
OREBOOT	Google	Experimental alternative for some stages of coreboot written in Rust	N/A
HardenedBoot	HardenedLinux	Integrated both verifiedboot and measuredboot	Yes
TrenchBoot	3mdeb	Support secure launch solution from Intel and AMD	Yes
u-bmc	Google, 9elements	Open source implementation for BMC	Yes

Reference

* Building Secure Firmware

https://www.apress.com/gp/book/9781484261057

* The Huawei and Snowden Questions

https://link.springer.com/book/10.1007/978-3-319-74950-1

* Intel Visa: Through the Rabbit Hole

https://github.com/ptresearch/IntelVISA-BH2019/blob/master/asia-19-GORYACHY-REMOLOV-Intel-Visa-Through-the-Rabbit-Hole.pdf

* The Long Hack: How China Exploited a U.S. Tech Supplier

https://www.bloomberg.com/features/2021-supermicro/

* Open Source is Insufficient to Solve Trust Problems in Hardware

https://media.ccc.de/v/36c3-10690-open_source_is_insufficient_to_solve_trust_problems_in_hardware

Reference

* RFC6979

https://tools.ietf.org/html/rfc6979

* LadderLeak: Breaking ECDSA With Less Than One Bit Of Nonce Leakage

https://eprint.iacr.org/2020/615

* FIPS-186-5

https://csrc.nist.gov/publications/detail/fips/186/5/draft

* Try Harder 2 Be Yourself

http://2012.zeronights.org/includes/docs/Keynote%20FX%20-%20Try%20harder%202%20be%20yourself.pdf

Reference

* PaX/Grsecurity:

https://grsecurity.net/

* Coreboot:

http://coreboot.org/

* Linux kernel mitigation checklist:

https://hardenedlinux.github.io/system-security/2016/12/13/kernel_mitigation_checklist.html

* Ring 0: Linux kernel vulnerablity & exploitation & silent fixes

https://github.com/hardenedlinux/grsecurity-101-tutorials/blob/master/kernel_vuln_exp.md

* Virtualization security:

https://github.com/hardenedlinux/grsecurity-101-tutorials/blob/master/virt_security.md

Reference

* Firmware security:

https://github.com/hardenedlinux/firmware-anatomy/blob/master/hack_ME/firmware_security.md

* Reproducible builds for PaX/Grsecurity

https://github.com/hardenedlinux/grsecurity-reproducible-build

* Hardened Boot:

https://github.com/hardenedlinux/Debian-GNU-Linux-Profiles/tree/master/docs/hardened_boot

* Neutralized ME stats

https://github.com/hardenedlinux/hardenedlinux_profiles/tree/master/coreboot

* Intel ME info:

https://github.com/hardenedlinux/firmware-anatomy/blob/master/hack_ME/me_info.md

QA

Thanks