Ironclad Clouds: How Linux Is Improving Infrastructure Security
Agenda

- Introduction
- Definitions
- Evolution of Linux Security Features
- Cloud Security Problems
- Linux is Still Evolving
- Two Features You Only Think You Don't Want
- Trusted Computing
- SELinux
- Combining Trusted Computing with SELinux
- Still More Security Is Needed
- Conclusion
- Disclaimers
IBM Linux Technology Center Security Team

H/W Crypto  Trusted Computing  Certifications

S/W Crypto  Virtualization & Cloud Computing
So What Do I Mean by “Cloud”?

- IT outsourcing / Modern Large Data Center
- Dynamic Infrastructure
- Virtualization – Specifically KVM
- Principles Apply to IaaS, PaaS, or SaaS
What Do I Mean by Ironclad?

Linux Has Evolved to Solve Many Security Problems

- **Features**
  - PAM: Identification and authentication
  - Cryptography: Confidentiality and integrity of data transit and at rest
  - Access Control: Separation of guests
  - Netfilter, vLANs, ebtables: Separation of guest network traffic
  - Audit: Monitoring, billing, attack event reconstruction
  - Cgroups: Resource control

- **Characteristics**
  - Innovative: Modern set of features
  - Many Eyes: Continuous code inspection
  - Source Code Availability: No hidden mysteries

- KVM Takes Advantage of Linux Security
Cloud Security Problems

- Regulatory Compliance
- Policy Enforcement
- Authorization
- Authentication
- Provenance
- Forensics
- Locality
- Privileged User Access
- Multitenancy
- Key Management
- Secure Lifecycle
- Auditability
But . . .

- How do I know I’m deploying on the hardware I think I am?
- How can I make sure my VM images are intact?
- How do I protect against guest privilege escalation?
- How do I ensure that guests are adequately separated?
- How do I securely migrate guests?
- How do I know that the required policy is being enforced?
- How can I decompose root privileges?
- How can I control guest access to storage?
- How can guests see what is going on beneath them?
- How can I prove to an auditor compliance with polices?
Linux Is Still Evolving to Address Security Problems

- Trusted Computing Controls for Integrity
- Multitenant Guest Storage Access Control
- Meaningful Role Separation and Role Semantics
- Extension of MAC Controls to Storage
- Audit Trail Centralization
- Centralized Access Control, Authorization, Key, and Integrity Management
- Signed Kernel Modules -> Asymmetric Crypto Modules
- Hardware Crypto Acceleration
- Cryptographic Domain Separation
- Minimization of libvirt Privileges
- Key Management
Two Much-Maligned Security Features You May Want in Your Cloud

- **Trusted Computing**
  - Trusted boot
  - Measurement of kernel
  - Measurement of kernel modules
  - Measurement of userspace
  - Integrity snapshotting and image alteration detection
  - Remote integrity verification
  - Unambiguous workload location

- **SELinux**
  - Complete mediation
  - VM separation
  - VM access control to host objects
  - Network controls – vLANs or labeled networking
  - Remote storage controls via file privilege separation or (someday) labeling
  - Administrative role separation
What Is Trusted Computing?

- The purpose: Determining if you can trust a platform
  - Is a remote machine running software I trust?

- How: The Trusted Platform Module (TPM)
  - Comprises a cryptographic engine and secure storage
  - Not necessarily hardware based
  - Support integrated into all levels of a platform, from firmware through user-space
    - As machine boots, it inserts cryptographic hashes of the software it runs into the TPM chip in PCRs (Platform Configuration Registers)
    - Signed sets of PCRs are sent to remote machines, who then determine whether they can trust the given configuration
    - The data is signed using a key tied to a certificate authority, certifying the key resides in a TPM
Static Measured Boot

- As the machine boots, its software state is stored in the TPM

Steps:
0. Hardware is powered on; BIOS begins execution; machine hashes its own BIOS and PCI card ROMs, storing them in the TPM in PCR0
1. The BIOS hashes the boot loader and stores it in PCR1; control is then transferred to the boot loader
2. The boot loader hashes the kernel and transfers control to it
3. As the kernel runs, it hashes all applications/libraries, etc and stores them
Dynamic Measured Host Boot

Steps

0. Hardware is powered on; BIOS executes

1. BIOS transfers control to boot loader

2. Boot loader transfers control to dboot module

3. dboot calls into UEFI to quiesce chipset

4. dboot calls into UEFI to place chipset into secure state

5. UEFI extends PCRs and then measures kernel & initramfs; makes go/no-go decision on kernel & initramfs and extends PCRs

6. As the kernel executes, it measures usespace and stores the measurements via Integrity Measurement Architecture (IMA) protected by the Extend Verification Module (EVM)

7. Platform Trust Services (PTS) attests using TPM-signed measurements
**Dynamic Measured Guest Boot**

**Steps**

0. Hypervisor is started; BIOS executes

1. BIOS transfers control to boot loader

2. Boot loader transfers control to dboot module

3. dboot calls into vUEFI to quiesce chipset

4. dboot calls into vUEFI to place chipset into secure state

5. vUEFI extends PCRs and then measures kernel & initramfs; makes go/no-go decision on kernel & initramfs and extends PCRs

6. As the kernel executes, it measures usespace and stores the measurements via Integrity Measurement Architecture (IMA) protected by the Extend Verification Module (EVM)

7. Platform Trust Services (PTS) attests using TPM-signed measurements
What Is Special about PCRs

- Extend operation: $\text{PCR}_n = \text{SHA1}(\text{concat}(\text{PCR}_n, \text{measurement}))$
- Computationally infeasible to fabricate a PCR state: same measurements in same order are required to set a particular state
- Data, including keys, can be sealed to particular set of PCRs in particular states
- Sealed data will not be released by the TPM unless PCRs to which it was sealed are in the same states as when sealing occurred
- Set of PCR states can be signed by a private key known only to the TPM
- Signed PCRs can then be provided to a challenger that can check signature with TPM's public key
- Challenger can replay measurements to recreate PCR states, and check against quoted PCR values
- Measurement list is authentic if calculated PCRs match quoted PCRs
Remote Attestation

- Machine A challenges machine B, determines a trust level, decides whether it wants to “do business”
Cloud Computing Turns Trusted Computing Upside Down

- IBM LTC Specifically Avoids Implementation of DRM Use Cases for Trusted Computing
- We Don't Want You to Have to Attest To Connect to Your ISP
- We Don't Want to Limit Your Freedom to Develop and Run What You Want
- In the Cloud, However, You May Need to
  - Know that you are using your real provider
  - Know that you aren't sharing a physical machine with your competitor
  - Deploy workloads in a specific geographic location
  - Verify that your VM image is the image you think it is
  - Verify that your VM image hasn't been maliciously or inadvertently altered
  - Know that your policies are being enforced
- This Protects and Enhances YOUR Security and Privacy!
Linux Trusted Computing Ecosystem Today

Remote Attestation

Userspace
- OpenPTS
- TrouSerS

Integrity Console
- tpm-tools
- OpenPTS
- TrouSerS

QEMU
- dboot
- GRUB
- vTPM
- D-RTM
- vUEFI
- Linux
- Initramfs
- TrouSerS

Measured Launch Event

Today
- TPM
- D-RTM
- UEFI

Future
- Userspace
- Integrity Console
- QEMU
- Measured Launch Event

Management
- Linux
- tpm-tis
- IMA
- EVM

- TrouSerS

- dboot
- GRUB
- vTPM
- D-RTM
- vUEFI

EVM
Trusted Computing Ecosystem Key

- **CRTM** – Core Root of Trust for Measurements – Immutable early firmware that starts the static integrity trust chain at reset
- **dboot** – Open Source TCG D-RTM Spec Compliant GRUB Module – Calls UEFI for late (D-RTM) measured launch
- **D-RTM** – Dynamic Root of Trust Measurement – “Late launch”; measurement begins at measured launch event, not reset
- **EVM** – Extended Verification Module – Protects IMA appraisal extended attributes on filesystem
- **GRUB** – Grand Unified Bootloader – De facto standard Linux bootloader by Free Software Foundation
- **IMA** – Integrity Measurement Architecture – Linux kernel feature to measure the integrity of files
- **OpenPTS** – Open Source TCG Platform Trust Services – TCG standard mechanism for remote attestation
- **TCG** – Trusted Computing Group – Standards body that oversees TPM and its ecosystem
- **TPM** – Trusted Platform Module – Small, inexpensive embedded security module accretes integrity measurements
- **tpm-tis** – TPM Device Driver – The Linux component that communicates with the TPM over LPC or I2C bus
- **TrouSerS** – Open Source TCG Software Stack – Component that applications use to communicate with the TPM
- **UEFI** – Unified Extensible Firmware Interface – Standard interface between OS and system firmware
SELinux

Subject ➔ syscall() ➔ Object

DAC
Permit ➔ Hook ➔ Deny

Access Vector Cache ➔ Security Server ➔ Policy DB

Policy Compiler

Binary Policy
8c ff 7c f9 08 00 00 00 53 45 20 4c 69 6e 75 78 12 00 00 00 00 00 00 00 06 00 00 00 00 00 00 00

Policy

Text Policy
class security
class process
class system
class capability
# file-related classes
class filesystem
class file
class dir
class fd
class lnk_file
class chr_file
class blk_file
class sock_file
class fifo_file

Permit ➔ DAC ➔ Permit

Deny ➔ DAC ➔ Deny

Audit

Policy Compiler

Text Policy
class security
class process
class system
class capability
# file-related classes
class filesystem
class file
class dir
class fd
class lnk_file
class chr_file
class blk_file
class sock_file
class fifo_file
SE Linux Policy Attestation

1. Request Signed Measurements
2. Check Certificate
3. Check Measurements
4. Make Decision

openPTS → TrouSerS

1. Request Signed Measurements

TPM

TPM

SELinux Policy

8c ff 7c f9 08 06 00 00
53 45 20 4c 69 6e 75 78
12 00 00 00 00 00 00 00
06 00 00 00 07 00 00 00

tpm-tis IMA EVM

PCR     template-hash                           filedata-hash                           filename-hint
10 7971593a7ad22a7ce5b234e4bc5d71b04d96fa4 ima b5a166c10d153b7c3e5bf41eab1f71672b7c524 boot_aggregate
10 2c7020ad9ca667419e497311e5704b8df52f77 ima e09e048c48301268f8b6459c3063137e42351d0 .init
10 e17a0affa03d46002ebd13d1d7894453656db3b ima 0f8b34325350e9ab912add3b474407635e3617c1 .init
10 247dbb0c62b346803660382d1973c019243e5f9 ima 747acba066d906392a62734916e0b39c6540931 ld-2.9.so
10 341de30e46fa5574b26e55e01e9ac22b6712dcb ima 326d45c3d74d88b23a3cbec9a4d93f1ac29818a ld.so.cache

TrouSerS

openPTS

3. Check Measurements

sshd

2. Check Certificate
Where Trusted Computing + SELinux Gets Us

- Trusted Computing
  - Can verify code and data before they are first utilized
  - Can verify integrity before interacting

- SELinux
  - Can control access to fine granularity
  - Can separate administrator roles

- Together
  - Trusted Computing verifies first use
  - A correctly written policy controls how memory is altered after first use
  - And the policy integrity can be checked

- There Are Still Bind Spots
  - Kernel vulnerabilities
  - Physical attack
  - And it adds even more code

- But we're created yet another barrier to attack
Combining Trusted Computing with SELinux in the Cloud
Other Ongoing Cloud Security Work

- VM Image Privilege Separation
- QEMU Network Helper
- Investigate Application of Seccomp to QEMU
- Host Audit Record Feedback for Guests
- Investigation of QEMU Fuzz Testing
Still More Effort Is Needed

- Correctness
- Hardening
- Attack Surface Reduction
- Fuzz Testing
- Static Analysis
- Memory Protection
- Separation Kernel
- Cryptographic Domains and Policy
- Fully Homomorphic Cryptography
- Secure Hardware
Conclusion

- Linux Has Evolved a Strong Set of Security Features
- Many of the Security Features Are Highly Forward Thinking
- Some Seemingly Less Desirable Measures Are Actually Useful for Securing Cloud Offerings
- Trusted Computing in the Cloud Inverts the DRM Scenario
- SELinux Can Augment Trusted Computing's Integrity Enforcement
- Trusted Computing Can Measure SELinux Policy
- A Number of Ongoing Projects Continue to Improve Linux for Cloud Infrastructure
- We Still Need to Do More
The End

Thank You!

George Wilson
<gcwilson@us.ibm.com>
Disclaimers

- This work represents the view of the authors and does not necessarily represent the view of IBM.
- IBM is a registered trademark of International Business Machines Corporation in the United States and/or other countries.
- Linux is a registered trademark of Linus Torvalds.
- Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Other company, product, and service names may be trademarks or service marks of others.
References

- KVM Security, [http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaat/liaatseckickoff.htm](http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaat/liaatseckickoff.htm)
- Red Hat Enterprise Linux Version 5 Virtualization with KVM Common Criteria Certification (in Evaluation), BSI-DSZ-CC-0724, [https://www.bsi.bund.de/ContentBSI/EN/Topics/Certification/newcertificates.html](https://www.bsi.bund.de/ContentBSI/EN/Topics/Certification/newcertificates.html)
Stay current on Linux and Open Virtualization at IBM

**Linux**

Follow us on Twitter:  [@Linux_at IBM](http://twitter.com/Linux_at IBM)

Like us on Facebook:  [Linux at IBM](http://facebook.com/Linux_at IBM)

[www.ibm.com/linux](http://www.ibm.com/linux)

---

**Open Virtualization & KVM**

Follow us on Twitter:  [@OpenKVM](http://twitter.com/OpenKVM)

Like us on Facebook:  [KVM at IBM](http://facebook.com/KVM_at IBM)


---

**Linux At IBM**

@linux_at_ibm

Linux is at the forefront of smarter solutions. IBM provides complete Linux solutions: top-to-bottom, end-to-end.


---

**KVM at IBM**


5 hours ago via Spredfast  ·  Like ·  Comment