Keylme - Securing your Slice of the Cloud

Andrew Toth
Red Hat
August 2019
What is Keylime
Path To Community

- **2013**: MIT Lincoln Laboratory Seed project for MOC
- **2015**: Red Hat Research and Red Hat Collaboratory at Boston University
  - Red Hat Interns work in collaboration with BU for MOC
- **2018**: Keylime community formed
  - Open Source Community formed to advance the Keylime technology
- **2019**: Keylime.dev Community Landing site introduced
Keylime answers the cloud trust problem

Keylime provides:

- Access to Hardware Root of Trust
- Remote Trusted Secure (measured) Boot
- Remote Integrity Verification Management
- Remote Encrypted Payload execution
Keylime answers the cloud trust problem

Keylime employs:

- Trusted Computing Group (TCG) TPM 2.0 hardware
  - 1.2 hardware is not supported

- Linux IMA
  - Collect, Store, Attest, Appraise, Protect, Audit

- Shim (first-stage boot loader)

- Trusted Software stacks (TSS and TCG spec)
  - tmp-tools built on TSS
Trusted Platform Modules

● Introduced by TCG
  ○ Version 1.2 introduced in 2011
  ○ Version 2.0 introduced in 2014

● TPM Chips are sealed units that are nearly ubiquitous in today’s computing hardware

● TPMs perform basic crypto operations (key and random number generation, object signing)

● Each TPM chip has a unique Private Encryption Key, that cannot be accessed externally
  ○ This key signs objects to show they have not been tampered with.

● TPM’s public key can be used anywhere to verify non tampering

● Maintains system measurements in on-chip Program Control Registers (PCRs)
TPM usage

- Disc Encryption
  - Utilities, such as dm-crypt, LUKS can use TPMs to protect the keys used to encrypt the computer’s storage devices (clevis).

- Password / Key protection

- Machine Identification

- Digital rights management

- Protection and enforcement of software licenses

- Prevention of cheating in online games

- Platform integrity (a.k.a remote attestation)
Keylime answers the cloud trust problem

Users View of Stack Trust

Cloud user
PaaS (openstack / openshift)

Hybrid Cloud Infrastructure

VM / Container
qemu / runc / CRI-O / docker...
Kernel
Bootloader and shim
firmware
hardware

Cloud user
IaaS (bare metal, ironic)
Keylime answers the cloud trust problem

Ripped from the Headlines!!!!

Resident evil: Inside a UEFI rootkit used to spy on govts, made by you-know-who (hi, Russia)
Deep dive into motherboard firmware-lurking code
By John Leyden 28 Sep 2018 at 02:07

A UEFI rootkit, believed to have been built by Kremlin spies from an anti-theft software program to snoop on European governments, has been publicly picked apart by researchers.

When a lower layer is compromised, the entire stack is compromised.

<table>
<thead>
<tr>
<th>Trust</th>
<th>VM / Container</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>qemu / runc/CRI-O / docker</td>
</tr>
<tr>
<td></td>
<td>Kernel</td>
</tr>
<tr>
<td></td>
<td>Bootloader and shim</td>
</tr>
<tr>
<td></td>
<td>firmware</td>
</tr>
<tr>
<td></td>
<td>hardware</td>
</tr>
</tbody>
</table>
Keylme answers the cloud trust problem

Actual Trust of Stack

Cloud user

PaaS
(openstack / openshift)

Hybrid Cloud Infrastructure

IaaS
(bare metal, ironic)

Cloud user

VM / Container

qemu / runc / CRI-O / docker...

Kernel

Bootloader and shim

firmware

hardware

Cloud user

Cloud user
Keylime - Distributed Architecture

Cloud Provider Hardware Node
- Keylime Agent
- TPM Software Stack
- TPM 2.0

Cloud Provider Infrastructure / Mobile Edge Site

TLS &/or IPSec Connection (TPM Trust Root)

Cloud User / Tenant Site
- Management system scheduler
- Keylime Verifier
- Keylime Registrar
- Keylime CA
- Keylime Tenant
- Plugin API (Revoke CA, Blacklist, Notify...)

TPM Software Stack
Keylime supports multiple distribution scenarios:

- Single site - single node (multi-user)
- Single site - multi-node (Datacenter, IoT)
- Multi-site - multi-node (Distributed Datacenter, Network Edge equipment, IoT)
- Multi-tenant (Cloud)
- Baremetal
- VM
Keylime - Operational Model

Remote System Integrity

- Measure of “Golden System” used to attest “like” systems
- TPM signs a cryptographic hash of all objects (firmware, bootloader, kernel,..)
- Hash list (SML) made public & used to remotely ‘attest’ a non-tampered boot.

<table>
<thead>
<tr>
<th>VM / Container</th>
<th>534670349c97163af557f43b0e5a791506440eef</th>
</tr>
</thead>
<tbody>
<tr>
<td>qemu / runc / CRI-O / docker..</td>
<td>b179e1a853b0bc757037a33ef6e7fbc5b0a6440b</td>
</tr>
<tr>
<td>Kernel</td>
<td>f6871775a38e2bc6e9326eccb4fe50bfde13b6fc</td>
</tr>
<tr>
<td>Bootloader and shim</td>
<td>b4ec92745434aa6717e500b12d6134f00a993799</td>
</tr>
<tr>
<td>firmware</td>
<td>02e98aba9aeee157b3e6c6e5bbfab3270bf5ad3c</td>
</tr>
</tbody>
</table>

Each part of the stack is “measured” by computing a cryptographic hash and extended (one way hash).
Remote File Integrity

<table>
<thead>
<tr>
<th>File to protect</th>
<th>TPM Signed hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/passwd</td>
<td>02e98aba9aeed157b3e6c6e5bbfab3270bf5ad3c</td>
</tr>
<tr>
<td>/etc/shadow</td>
<td>b4ec92745434aa6717e500b12d6134f00a993799</td>
</tr>
<tr>
<td>/home/$user/my_network_layout.yml</td>
<td>f6871775a38e2bc6e9326eccb4fe50bfde13b6fc</td>
</tr>
<tr>
<td>/home/$user/Fedora-Cloud-Base-29.qcow2</td>
<td>b2a83b0ebf2f8374299a5b2bdfc31ea955ad7236</td>
</tr>
<tr>
<td>/home/$user/docker_image.tar</td>
<td>b179e1a853b0bc757037a33ef6e7fbc5b0a6440b</td>
</tr>
</tbody>
</table>

Hash list can be publicly ‘attested’ for non tampering.
Keylime - Operational Model

Remote Trusted Secure Boot

Hey Mr Cloud Provider, can I trust your infrastructure?

Here is my Stored Measurement Log signed by the TPM

Nice! Here please Instantiate my VM / Container

Cloud User (deploy workload)

TPM Public key

Keylime Attestation Service

TPM Private key

Cloud Provider (Infra)
Hey Cloud Node, can I trust what you have loaded and running?

Here is my latest IMA Measurement signed by the TPM

You've been Compromised, you're cut off
Part II

Operational Walkthrough
Keylime Components

- **Cloud Verifier**
  - Tenant-owned verifier checks node system integrity

- **Tenant Registrar**
  - Stores public keys of TPMs in tenant’s environment

- **Cloud Node**
  - Machine the tenant wants to use securely
  - Tenant-controlled
  - Provider-controlled

- **Tenant**
  - Cloud user

- **Agent**
  - TPM
Secret Keylime Sauce

Tenant-controlled
Provider-controlled

Delegate Integrity Check

Tenant Cloud Verifier

Check TPM Keys

Check Node Integrity

Tenant Registrar

Demonstrate Intent

Cloud Node

Agent

TPM

Key shares recombined to obtain bootstrap key
Keylime key definitions

- **EK**
  - Endorsement Key, Permanent, certifies valid TPM
- **AIK**
  - TPM key to sign quotes
- **$K_e$**
  - Ephemeral challenge key to certify AIK
- **$K_b$**
  - Bootstrap key to deliver to node
- **U,V**
  - Split shares of $K_b$ to pass to node during node onboarding phase
- **NK**
  - Protector of U and V transmission
Keylime Attestation: Identity Key Registration

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>EK</td>
<td>RSA 2048</td>
<td>Permanent, certifies valid TPM</td>
</tr>
<tr>
<td>AIK</td>
<td>RSA 2048</td>
<td>TPM key to sign quotes</td>
</tr>
<tr>
<td>$K_e$</td>
<td>AES-256</td>
<td>Ephemeral challenge key to certify AIK</td>
</tr>
</tbody>
</table>

**Cloud Node**

- **Agent**
  - **TPM**
    - **EK** \(_{priv}\)
    - **AIK** \(_{priv}\)
- **HMAC\(_{K_e}(ID)\)**
- **Enc\(_{EK}(H(AIK_{pub}),K_e)\)**

**Tenant Cloud Verifier**

- **ID, AIK_{pub}, EK_{pub}**

**Tenant Registrar**

- **Challenge node to decrypt $K_e$ with EK**
- **Prove Node knows $K_e$**
- **AIK_{pub} successfully tied to EK identity**

**Legend**

- **Tenant-controlled**
- **Provider-controlled**

EK: Endorsement Key
AIK: Attestation Identity Key

---

- **Provide public keys**
- **Challenge node to decrypt $K_e$ with EK**
- **Prove Node knows $K_e$**
- **AIK_{pub} successfully tied to EK identity**
### Keylime Bootstrap: Checking Integrity (Part 1)

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIK</td>
<td>RSA 2048</td>
<td>TPM key to sign quotes</td>
</tr>
<tr>
<td>$K_b$</td>
<td>AES-256</td>
<td>Bootstrap key to deliver to node</td>
</tr>
<tr>
<td>$U,V$</td>
<td>256bit rand</td>
<td>Trivial secret shares of $K_b$</td>
</tr>
<tr>
<td>NK</td>
<td>RSA-2048</td>
<td>Protects secret shares $U,V$ in transit</td>
</tr>
</tbody>
</table>

**Legend**
- **Tenant-controlled**
- **Provider-controlled**

**AIK**: Attestation Identity Key
**PCR**: Platform Configuration Register, where TPM stores integrity measurements

**Notify verifier of new node. Delegate key share**

**Quote** proves system integrity. Provide transport key for $V$

**Check validity of AIK that signed the quote**

**Send key share if integrity and TPM identity good**

**Tenant Cloud Verifier**

**Cloud Node**

**Tenant Registrar**
Keylime Bootstrap: Checking Identity (Part 2)

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIK</td>
<td>RSA 2048</td>
<td>TPM key to sign quotes</td>
</tr>
<tr>
<td>K_b</td>
<td>AES-256</td>
<td>Bootstrap key to deliver to node</td>
</tr>
<tr>
<td>U,V</td>
<td>256bit rand</td>
<td>Trivial secret shares of K_b</td>
</tr>
<tr>
<td>NK</td>
<td>RSA-2048</td>
<td>Protects secret shares U,V in transit.</td>
</tr>
</tbody>
</table>

**Legend**
- **Mutual TLS**
- **Server TLS**
- **No TLS**
Keylime System Integrity Monitoring

Tenent

Tenant Cloud Verifier

Valid AIK?

Check validity of AIK that signed the quote

Tenant Registrar

Quote_{AIK}(nonce, PCRs)

Quote proves system integrity

Random number ensures quote is fresh

Cloud Node

Agent

TPM

Tenant-controlled

Provider-controlled

Legend

- - - Mutual TLS
- Server TLS
- No TLS

AIK: Attestation Identity Key
PCR: Platform Configuration Register, where TPM stores integrity measurements
Keylime System Integrity Monitoring

Tenant

Cloud Verifier

Tenant Registrar

Cloud Node

Quote_{AIK}(nonce, PCRs)

Quote proves system integrity

Random number ensures quote is fresh

Tenant-controlled

Provider-controlled

AIK: Attestation Identity Key
PCR: Platform Configuration Register, where TPM stores integrity measurements

Legend

- Mutual TLS
- Server TLS
- No TLS

Report integrity violation to listeners

Random number ensures quote is fresh

Quote proves system integrity

Agent

TPM

PCRs

AIK
Cloud Usage: Certificate Authority and Revocation

Legend
- Mutually Trusted TLS (Mutual TLS)
- Client TLS (Server TLS)
- No TLS

Software CA

Tenant

Cloud Node:
- Private key and certificate
- TPM
- Agent

Tenant-controlled
Provider-controlled

CA: Certificate Authority
Cloud Usage: Certificate Authority and Revocation

Software CA

Revoke signed certificate

Tenant Cloud Verifier

Integrity measurement

Cloud Node

Agent

TPM

Legend

- - - Mutual TLS
- - - Server TLS
- - - No TLS

Bootstrap Key Enables

- Secure config management
- Encrypted hard disks
- Provenance authenticity
- Secure databases
- Data integrity
- IPsec encryption

Tenant-controlled

Provider-controlled

CA: Certificate Authority
Cloud Usage: IPsec Encryption with Revocation

Tenant Cloud Verifier

Quote proves system integrity

Cloud Node #1
Agent
TPM

Cloud Node #2
Agent
TPM

Quote proves system integrity

IPsec node-to-node communication

Tenant-controlled
Provider-controlled
Cloud Usage: IPsec Encryption with Revocation

Tenant

Cryptographic certificate revocation

Tenant Cloud Verifier

Cloud Node #1

Cloud Node #2

Agent

TPM

Agent

TPM

Node #2 cannot communicate with Node #1

Tenant-controlled

Provider-controlled
Demonstration
Demo info

https://keylime-docs.readthedocs.io/en/latest/user_guide/user_selected_pcr_monitoring.html

Wrap up
Current Keylime community activities:

- Heavy development activity for version 5.0 release
  - Porting from Python2 to Python3
  - Porting to pycryptography from pycryptodome
    - Pycryptography is FIPS compliant

- Fedora Packaging

- Documentation validation and improvements

- vTPM development for KVM hypervisor
Xen hypervisor supports vTPM since version 4.3
- Each vTPM is a separate Xen VM
- Trust of vTPMs rooted in hardware TPM of the hypervisor
- DeepQuote operation to obtain hardware TPM quote from a vTPM

Apply the principle of least privilege to all provider-layer resources
Virtualizing Keylime – Architecture

- **Tenant Cloud Verifier**
  - Use DeepQuote instead of quote
  - Use DeepQuote to bind vTPM keys with hardware TPM

- **Tenant Registrar**
  - Validate AIK from DeepQuotes with provider

- **Provider Registrar**
  - Stores public keys of TPMs in hypervisor hosts

- **Cloud Node**
  - Hypervisor
  - vTPM
  - Agent

- **Legend**
  - Tenant-controlled
  - Provider-controlled
  - Mutual TLS
  - Server TLS
  - No TLS

**AIK:** Attestation Identity Key
**vTPM:** Virtual Trusted Platform Module
What’s going on in the Keylime community

Future Keylime community activities:

● Future development activity
  ○ Porting non-Agent components from python to Rust as needed
  ○ Integration with deployment and management tools

● OS and platform compatibility
  ○ RHEL 8
  ○ CentOS 8
  ○ Ubuntu
  ○ ARM
  ○ TPM module testing
  ○ System diversity testing

● Containerization
  ○ Keylime itself
  ○ Integrity checking containers
Protect Your Slice
Come join us

https://keylime.dev
Thank you

Red Hat is the world’s leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat  facebook.com/redhatinc

youtube.com/user/RedHatVideos  twitter.com/RedHat