Using the TPM to Solve Today’s Most Urgent Cybersecurity Problems

May 20, 2014 10:00AM PDT
No Sound?
This is a streaming audio event. Make sure the sound on your PC is turned on.

Questions?
Type your questions using the Ask a Question Text Box
Today’s Presenters

Stacy Cannady, Technical Marketing – Trustworthy Computing, Cisco

Stacy Cannady, CISSP, is technical marketing - Trustworthy Computing TRIAD (Threat Response, Intelligence, and Development) for Cisco and a member of the Trusted Computing Group's Embedded Systems Work Group. He also serves as the primary board member representing Cisco.

Stacy has worked in the field of trusted computing for a number of years. As a subject matter expert in trusted computing, his responsibilities require an in-depth understanding of the trusted computing market, including advances in hardware and software security as well as vendor and customer market dynamics.

Monty Wiseman, Security Architect, Intel Corporation

Monty is a Security Architect for Intel's Data Center Group (DCG). His current projects include architecture for TCG, Intel's TXT(R) Technologies, Boot Guard(R) and other security initiatives.

Monty has participated (for chair of) the TCG PC Client working group, Security Evaluation working group and is Intel's representative in the TCG Technical Committee. Monty has 20 years’ experience in Desktop, Network and Mainframe environments holding security related and other engineering positions at Novell, Fujitsu, and Control Data. Monty has been developing hardware and software for computers ranging from mainframe to microcomputers since 1975.
Use Cases
TPM and Core Root of Trust for Measurement (CRTM) measure BIOS at boot.
Measurements are internally verified.
If a mismatch is found, the offending module is rolled back to the Last Known Good.
Then boot continues.

1. Power on
2. CRTM measures BIOS
3. Mismatch found
3.5 Rollback bad module to last known good copy
4. Execute BIOS and boot

Copyright 2014 Trusted Computing Group
Windows 8 TPM Usage

1. Secure boot prevents running an unknown OS loader.
2. ELAM starts first and enforces its policy.
3. Boot measurements were recorded during boot.
4. Signed TPM boot measurements can be sent to an off-box service for analysis.

Secure boot prevents running an unknown OS loader.
ELAM starts first and enforces its policy.
Boot measurements were recorded during boot.
Signed TPM boot measurements can be sent to an off-box service for analysis.

Slide duplicated from Microsoft //build presentation - September 2011.
AFRL’s SecureView 1.2 Architecture

The integrity of blue components is validated Intel HW and the TPM
Applying the SecureView Model to Cloud Servers

- **SecureView uses Intel TXT and the TPM to validate BIOS and the Hypervisor at start**
  - You always know you started a trusted Hypervisor and all of its services
  - If one of those services continuously validates runtime integrity, you know the hypervisor remains trusted

- **If VMs are also integrity checked at start, then the VM is also trusted**
  - Same deal – if one of the services in a VM validates runtime integrity, you know the VM remains trusted

- **Defense against zero-day attack: Integrity validation is focused only on detecting ANY uncontrolled change – not any specific change**
Example Cloud Stack at System Start

Integrity of VM measured at start and runtime integrity validated as for hypervisor

Runtime integrity validated by Trapezoid or the like

TXT and TPM validate integrity of Hypervisor

Copyright 2014 Trusted Computing Group
Protection of the Cloud Stack in Execution

Runtime integrity validation monitors uncontrolled changes to processes in execution - any change is seen, therefore we have (limited) Zero-Day defense

- Continuous integrity validation gives no insight into the nature of the change.
- Operationally, that is generally unimportant.
- The rapid discovery that uncontrolled change occurred, IS important.

The HW is mostly out of the loop during runtime (until TCG’s DRTM spec is implemented)
Design:
- Authentication protocol enhanced to require
  - Certificate exchange
  - Integrity report exchange
- At session start, each side
  - Signs a nonce and their integrity report using a HW protected key
  - Validates the provided report
- No match, no session
  - No session, no hack

Central ICS console

GPU includes a TPM

Unknown operator

No key and no valid integrity report
Rationale
BIOS resides in privileged space: Invisible to Anti-Virus SW
  - Malware in BIOS is “persistent”: Does not go away by rebooting or reinstalling the OS

CIH/Chernobyl – 1998 – killed between 1M and 2M machines

Increase in security research…

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rootkit in BIOS via ACPI</td>
<td>John Heasman</td>
<td>2007</td>
</tr>
<tr>
<td>Hijacked ACPI to inject Rootkit into system BIOS</td>
<td>Sherry Sparks</td>
<td>2008</td>
</tr>
<tr>
<td>SMM rootkit with covert packet exfiltration</td>
<td>Alfredo Ortega/Anibal Sacco</td>
<td>2009</td>
</tr>
<tr>
<td>Undetectable Key Logger</td>
<td>Butterworth/Kallenberg/Herzog/Kovah</td>
<td>2013</td>
</tr>
<tr>
<td>Problems with the Static Root of Trust for Measurement</td>
<td>51 byte patch to SCRTM to subvert TPM measurements</td>
<td>2013</td>
</tr>
<tr>
<td>A Tale of One Software Bypass of Windows 8 Secure Boot</td>
<td>Bulygin/Furtak/Bazhaniuk</td>
<td>2013</td>
</tr>
</tbody>
</table>

NIST SP 800-155

BIOS Flashing Trojan “in the wild”
  - Trojan.Mebromi – Sept 2011

Research is a leading indicator of future exploits. Secure boot bypass provides motivation.
Components: Hardware + Firmware

Outside World’s view of the Component.

FW Component

HW Component

Mutable Components
e.g., Firmware, software, configuration settings

Immutable Components
i.e., hardware
Issue with Trusting Firmware / Software

FW/SW Component V1

Version? V_1

HW Component

FW/SW Component V2

Version? V_2

HW Component

FW/SW Component V1 (Attacked)

Version? V_2

HW Component
Adding a Root to an Immutable Component

Outside World’s view of a **Trusted Component**.

- **FW Component**
- **Root**
- **HW Component**

**Mutable Components**
- e.g., Firmware, software, configuration settings

**Immutable Components**
- i.e., hardware

Visible & measureable behavior

Undetectable behavior

Copyright 2014 Trusted Computing Group
Glossary:

A component that performs one or more security-specific functions, such as measurement, storage, reporting, verification, and/or update. It is trusted always to behave in the expected manner, because its misbehavior cannot be detected under normal operation.

Explanation:

Must begin by trusting a set of components to perform necessary functions to support Trusted Computing in the expected manner.

Misbehavior of these components is not detectable by the evidence and protocols provided by Trusted Computing.

For example: A root fails to properly initialize a platform and measurements begin incorrectly. While this misbehavior might be detectable by physical inspection (e.g., in a lab) this misbehavior cannot be detected using Trusted Computing methods (i.e., attestation).
TPM Shielded Locations
location on a TPM that contains data that is shielded from access by any entity other than the TPM and which may be operated on only by a Protected Capability

TPM Protected Capabilities
operation performed by the TPM on data in a Shielded Location in response to a command sent to the TPM
• **Verified Boot:**
  - Boot policies are enforced during the boot process. Starting with the Root of Trust for Verification the currently executing module verifies the next module against a policy (i.e., manifest).

• **Measured Boot:**
  - Integrity Measurements are placed into the TPM. Starting with the Root of Trust for Measurement the currently executing module places the Integrity Measurement of the next module into the TPM.

• **Secure Boot:**
  - A boot process which implements either Verified Boot or Measured Boot or both.

• **Clarification:**
  - Verified Boot is often referred to as Secure Boot. Measured Boot is often referred to as Trusted Boot. Verified Boot (i.e., Secure Boot), however, only provides assurance that the boot policy was enforced and does not provide any assurance or evidence that the components are “secure”. Therefore, the moniker “secure” in this context is misleading.
Measured & Verified Boot

Unsecured Boot
- Reset
- Assumption
- Execute

Verified Boot
- Reset
- Execute
- Verify

Measured Boot
- Reset
- Execute
- Measure

IT
- Verifies against manifest
- Has no proof of proper boot

TPM
- IT
- Verifies against manifest
during / after boot
- Local Attestation: TPM Enforces Policy
- Remote Attestation: TPM Key signs measurements.

Policy applied starting here

Copyright 2014 Trusted Computing Group
Attestation

- **Proof of a Device’s Identity**
  - Proof (e.g., Authentication) of a platform’s identity
    - TPM adds a cryptographic binding between the platform and a key

- **Proof of a Device’s [boot-time] Integrity**
  - Proof of the identity of the platform’s firmware, software components and configuration
Components of Attestation

- **Integrity Measurement**
  - Recorded metrics of the platform’s characteristics
  - Typically done using a series of cryptographic recordings
    - At Reset PCR[X]:=0 then
    - PCR[X] = hash{PCR[X] | Integrity Metric of component 1} then
    - PCR[X] = hash{PCR[X] | Integrity Metric of next component}, etc.

- **Roots**
  - Elements which must be trusted
  - Misbehavior is not detectable
  - Objective: Keep these small
Specifications
- A cohesive set of measurement started by RTM
- Provides an “audit” of boot sequence
• Maps the BIOS components to PCRs
  • E.g., PC Client Specification for UEFI
Implementation Options (1 of 2)

Discrete IP block (a chip)

Integrated IP block

Separate HW Domain

Copyright 2014 Trusted Computing Group
Implementation Options (2 of 2)

Separate Virtual Domains
Specifications Layers

Applications

TSS

TDDDL

Driver

TPM Defined Interface

TPM Defined Internal Resources

TPM X.X Family Functions

Platform TPM Profile
(e.g., PC Client PTP, aka: TIS)

TPM 2.0 Library

Device Driver Writers Guide

Platform Specific Firmware Specifications
(e.g., PC Client Firmware Specification for UEFI)

RTM
(e.g., ROM, BIOS Boot Block)

TPM
(e.g., discrete chip, integrated IP Block)

Boot FW
(e.g., BIOS)

OS

Byte Stream

Byte Stream
Software Stack (Anticipated)

Applications

TSS
Feature / Environmental API

Linux Keyring

Java classes:
crypto / keyring

PKCS#11

OS or Application Specific Key Managers

TSS

General Interfaces and code

Expert-level Interfaces and code

TDDL

Driver

PTP Defined Interface

PTP Defined Internal Resources

TPM XX Family Functions

TPM
Usages
<table>
<thead>
<tr>
<th>Usage</th>
<th>Description</th>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Attestation (Remote Attestation)</td>
<td>IT wants data is sent only to authorized device. Assume user only executes authorized FW and OS.</td>
<td>Prevent sending data to unauthorized devices. Need to determine the authenticity of an endpoint’s claimed identity when: • Device is outside IT’s physical control • Connected via untrusted connection</td>
<td>• Protect key authenticating device using a TPM key • Use a signing key with Properties: • Non-duplicateable • HMAC authorization • Assume FW and OS installed by user will protect assets at runtime</td>
</tr>
<tr>
<td>Device &amp; Software Attestation (Remote Attestation)</td>
<td>IT wants data is sent only to authorized device. Do not assume user only executes authorized FW and OS. Enforce authorized FW and OS was executed before sending data.</td>
<td>Above + Enforce device executing authorized firmware / OS.</td>
<td>• Use a signing key with Properties: • Above + • PCR Policy • Enforce FW and OS installed by user to protect assets at runtime</td>
</tr>
<tr>
<td>Device Asset Protection (Local Attestation)</td>
<td>Device lost or stolen. Attacker can mount and read device’s media. Assume device executes only authorized FW and OS which provides runtime asset protection.</td>
<td>Protect data from access when in the physical possession of attacker. Protect data with key but attacker can break user’s low entry PIN protecting key using offline methods. Users will not use high-entropy PINs. Assume device executes only authorized FW and OS which provides runtime protection of key.</td>
<td>• Protect device encryption key using a TPM key • Use a signing key with Properties: • Duplicateable (For recovery) • HMAC authorization • TPM’s anti-dictionary attack mechanisms allow low-entropy, user-friendly(er) “PINs” • Note: Above protections come into play when attacker attempts to reboot device. Runtime protections are provided by OS.</td>
</tr>
<tr>
<td>Device &amp; Software Asset Protection (Local Attestation)</td>
<td>Device lost or stolen. Attacker can mount and read device’s media. Enforce device executes only authorized FW and OS which provides runtime asset protection.</td>
<td>Above + Enforce device executes only authorized FW and OS which provides runtime asset protection.</td>
<td>• Protect device encryption key using a TPM key • Use a signing key with Properties: • Above + • PCR Policy • Note: As above, but now PCR Policy enforces FW and OS.</td>
</tr>
</tbody>
</table>
Simplified Protocol

**Device Manufacturer**

IT gets TPM/Platform Certificates

IT gets list of reference PCR values for authorized FW / OS

**Retail Store**

Ship Device

**Employee**

Employee sends device attestation

Employee gets list of IT authorized devices

**IT**

IT verifies PCR values against manifest provided by Manufacturer

IT verifies device is from authorized Manufacturer

IT verifies PCR values against manifest provided by Manufacturer

IT verifies encryption key meets policies. Now sends data to device

IT verifies PCR values against manifest provided by Manufacturer

TPM generate media encryption key w/ PCR values above

CertifyingKey{Enc Key properties}

IT verifies encryption key meets policies. Now sends data to device

E.g., per NIST 800-155

IT requests signed PCR values

Key{ PCR[] }
TPM Features: Summary

- An open, defined set of building block protocols providing platform-level attestation and storage protections
  - Discussed previously

- Flexible set of Policies
  - Policies are applied to objects when they are created
  - Many policies types supported
  - Can combine with AND and OR operations

- Algorithm Agility
  - Protocols provide selectable (at implementation or use time) algorithms
    - Symmetric keys (DEC, AES, etc.)
    - Asymmetric keys (RSA, ECC, etc.)
    - Hash algorithms (SHA-1, SHA-256, SHA-384, etc.)

- Enhanced Provisioning
  - Separation of assets for the platform and user
Examples

- **Password**: Simple and in the clear – used for trusted environments
- **HMAC**: A shared key using HMAC to protect secret in transit
- **PCR**: Platform must match expected configuration (measured firmware and software)
- **Time**: Can restrict a key to be used only for a certain period of time
- **Count**: Can restrict a key to be used only one time (or a set number)
- **Signed**: Verifies a signature from an external private key (e.g., SmartCard)
  - This allows stronger two factor authorization (i.e., authorization is: Using the authorized device (TPM) AND have possession of the SmartCard + PIN.

**ANDs and ORs**

- Can create a policy such as:
  - (PCR) AND (Secret) : Normal user access TPM protected data. Only authorized if authorized software and user are present
  - OR
  - (Signed) : An IT admin uses a SmartCard (even remotely) in case user forgets Secret
TPM 2’s library structures are not hard-coded to a particular algorithm

- TPM may support more than one
  - Protocols provide fields for selecting algorithms for each function
- TPM vendor implement selected algorithms
  - Platform TPM Profile specifies minimum sets of algorithms
    - TPM vendor may implement more than minimum set

Supports types such as

- Symmetric keys (DEC, AES, etc.)
- Asymmetric keys (RSA, ECC, etc.)
- Hash algorithms (SHA-1, SHA-256, SHA-384, etc.)

Extensible

- TCG Registry published identifying each TCG enumerated algorithm
Enhanced Provisioning

Platform Hierarchy
- Platform Assets

Storage Hierarchy
- User Assets for signing and storage

Endorsement Hierarchy
- User Assets for Attestation

Independent Controls for each hierarchy
Questions?

Post your question now.
• TPM 2.0 Library Specification: http://www.trustedcomputinggroup.org/resources/tpm_library_specification

• Infographic - Protect Your Data and Enhance Security: http://www.trustedcomputinggroup.org/resources/protect_your_data_and_enhance_security


• Root of Trust: Foundation for IT Security Article: http://www.trustedcomputinggroup.org/resources/root_of_trust_foundation_for_it_security

• Become a TCG Member – Join Now!: http://www.trustedcomputinggroup.org/join_now/membership_benefits