Using IMA for Integrity Measurement and Attestation

David Safford, Mimi Zohar, Reiner Sailer
Integrity Measurement Architecture (IMA)

- **What is IMA?**
  - New kernel feature as of 2.6.30

- **Enabling IMA**

- **Using IMA**
  - Standalone or with TPM
    - health check - PTS
    - Network admission - 802.1x-TNC-PTS

- **Future Work**
  - EVM
**Trusted Computing: architecture & opensource components**

IMA maintains a list of file hash values and, if the system has a TPM chip, maintains an aggregate integrity value over this list inside the TPM hardware. The integrity measurement list can be read through a securityfs file, typically mounted at /sys/kernel/security/ima. The aggregate integrity value, normally in Platform Configuration Register(PCR) 10, can be signed using the TPM quote, so that the measurement list can be cryptographically verified. Together the measurement list and the signed aggregate integrity value can be used to attest to a system's runtime integrity.

IMA's maintenance of a TPM hardware anchored file measurement list is fundamental to TCG's Platform Trust Services(PTS) and, not shown here, Trusted Network Connect(TNC) standards.

<table>
<thead>
<tr>
<th>Applications</th>
<th>spec</th>
<th>info</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTS</td>
<td>spec</td>
<td>OpenPTS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tpm-tools</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Libraries</th>
<th>spec</th>
<th>info</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSS</td>
<td>spec</td>
<td>TrouSerS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Linux Kernel</th>
<th>spec</th>
<th>info</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMA</td>
<td>spec</td>
<td>TPM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tpm-1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TPM driver</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boot</th>
<th>spec</th>
<th>info</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOS</td>
<td>spec</td>
<td>GRUB-IMA, TBOOT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware</th>
<th>spec</th>
<th>info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>spec</td>
<td>TPM</td>
</tr>
</tbody>
</table>
A Blatant Plug

- Programming
  - BIOS
  - Device Driver
  - TPM
  - TSS

- Applications
  - Trusted Boot
  - Key Management
  - Authentication
  - Attestation
Trusted Platform Module (TPM)

- **RSA crypto**
  - key generation, signature, encrypt, decrypt

- **Secure storage**
  - private keys
  - master keys (eg loopback)

- **Integrity measurement**
  - Platform Configuration Registers (PCR)
  - compromise detection
  - Tie key use to uncompromised environment

- **Attestation**
  - host based integrity/membership reporting
  - (RSA 2004 Demo)
TPM as a Root of Trust

- **Static Root of Trust (SRTM)**
  - Immutable BIOS measures mutable BIOS
  - Each step thereafter measures the next stage

- **Dynamic Root of Trust (DRTM)**
  - Atomic measure/load/execute bootstrap
  - Not dependent on BIOS
SMM Attack on DRTM (Rutowska, Blackhat 2009)

TXT attack sketch (using tboot+Xen as example)

1. Attacker patches the bootloader (e.g. GRUB). The patched code injects a shellcode to SMM
2. Evil shellcode will infect the Xen hypervisor later...
3. After xen.gz gets successfully loaded, the evil code from SMRAM can easily infect it...

Notes:
- Diagram is not in scale!
- SENTER also resets and extends PCR17 with hash of SINIT/BIOSACM/(STM) LCP
Addressing the generalized problem

- “Fixing SMM” addresses just this set of attacks
- If you run someone else's program on your computer, it's no longer your computer
- If you don't know what's running on your computer, you cannot know if it is still your computer
- This includes ALL programs, at ALL levels

BIOS  GRUB  tboot  kernel  initrd  init  daemons  applications......

SRTM  DRTM  IMA
Integrity Measurement Architecture (IMA)

- Trusted Computing Group Trust Architecture
  - Chain of trust – measure files before accessed/executed
  - Store measurements in kernel list
  - Extend measurements into TPM/vTPM PCR
  - Attest all measurements to third party, signed by TPM/vTPM
  - Malware cannot take measurements back from TPM/vTPM

- IMA is linux kernel module which implements this model
  - Policy based for which files to measure
  - High performance with measurement caching
Using IMA – the basics

- Config/install the kernel
  - Integrity Measurement Architecture (IMA) (NEW)
  - (IMA and TPM drivers must be compiled in, not modules)
- Add “ima_tcb=1” to kernel boot
- Mount securityfs
- If desired load tailored policy
  - /sys/kernel/security/ima/policy
- Read measurement list
  - /sys/kernel/security/ima/ascii_measurement_list
IMA Raw output

After typical boot (Fedora 11), 1600 measurements.
Overhead at boot time, 10% (3-5 seconds).
Slight performance improvement at runtime (a lot prefetched).
IMA Measurement Policy

- Want to measure all files, but unacceptable performance
- Some measurement decisions are easy:
  - All executed files, #! scripts (bprm hook)
  - All files mmap'ed executable (mmap hook)
- Some read()'s are sensitive, but not all...
  - scripts, config files are sensitive
  - NOT – log files, LARGE files (KVM images...)

Need a measurement policy integrated with LSM, to take advantage of selinux subject, object, type labels
IMA Policy language

action: measure | dont_measure
condition:= base | lsm

base:  [[func=] [mask=] [fsmagic=] [uid=]]

   func:= [BPRM_CHECK][FILE_MMAP][PATH_CHECK]
   mask:= [MAY_READ] [MAY_WRITE] [MAY_APPEND] [MAY_EXEC]
   fsmagic:= hex value (or NAME)
   uid:= decimal value

lsm:   [[subj_user=] [subj_role=] [subj_type=]
        [obj_user=] [obj_role=] [obj_type=]]

   these are LSM specific

Omitted conditions match any, if no matching rule then dont_measure
IMA ima_tcb=1 default policy

dont_measure fsmagic=PROC_SUPER_MAGIC
dont_measure fsmagic=SYSFS_MAGIC
dont_measure fsmagic=DEBUGFS_MAGIC
dont_measure fsmagic=TMPFS_MAGIC
dont_measure fsmagic=SECURITYFS_MAGIC
dont_measure fsmagic=SELINUX_MAGIC
measure func=BPRM_CHECK
measure func=FILE_MMAP mask=MAY_EXEC
measure func(PATH_CHECK mask=MAY_READ uid=0
Example LSM Specific Measurement Policy

SELinux:
dont_measure obj_type=var_log_t
dont_measure obj_type=auditd_log_t

Smack:
measure subj_user=_ func=INODE_PERM mask=MAY_READ
IMA messages – when IMA Can’t Measure a file

- The kernel prohibits writing and executing a file concurrently
  - Other files can be read and written concurrently
- “open_writers” - file already open for write, is opened for read
- “ToMToU” (“open_reader”) – file already open for read is opened for write
- In these two cases, IMA cannot know what is actually read, and invalidates the measurement with all zeros
- Applications that do this have no idea what they are reading
  - Possibility of failure, cross domain exploit?
  - Gnome-pty-helper -> utmp, configure -> sh
- In cases so far, just tweak policy not to measure
IMA with TPM

- TPM device driver (TPMDD) – in kernel
- TPM library (TSS) – Trousers
  - TPM tools, utilities – included with Trousers
  - TPM_QUOTE can be used to sign measurement chain pcrs
Detecting and Isolating Compromised Systems

- **Boot Time Integrity Measurement and Attestation**
  - TPM based SRTM and DRTM

- **Run Time Integrity Measurement and Attestation:**
  - IMA – Integrity Measurement Architecture (2.6.30)
    - Measure all files before they are used
    - TPM based attestation of measurement list (PTS)

- **Network Admission/Isolation Time**
  - 802.1x-TNC-PTS standards compliant attestation for network admission
    - Trusted Network Connect (TNC)
    - Platform Trust Services (PTS)
Using Platform Trust Services (PTS)

- TCG XML standard for reporting integrity measurements
  - IMA measurement list and TPM_QUOTE
- How do you know “good” measurements?
  - Managed clients
    - Comparison to managed image
    - Reference Manifest vs current report (ie what’s changed?)
    - Comparison to reference database (nsrl.nist.gov)
Attestation 802.1x-TNC-PTS

Figure 5: The TNC Architecture with the Trusted Platform Module (TPM)
802.1x-TNC-PTS in Virtualized Environments
802.1x-TNC-PTS-IMA Resources

- OpenPTS (Seiji Munetoh, TRL)

- 802.1x-TNC Client: wpa_supplicant
  - http://hostap.epitest.fi/wpa_supplicant/

- 802.1x-TNC Server: tnc@fhh
  - Fachhochschule Hannover
    (University of Applied Sciences and Arts)
  - http://trust.inform.fh-hannover.de/joomla/

- Libtnc
  - http://sourceforge.net/projects/libtnc
Future work – EVM – Local “Appraisal”

- Verification of a file’s data and metadata (Ism labels)
- Original EVM
  - Xattr stored HMAC on data hash
  - Initial labeling slow, and directory level attacks
- Fixing Design
  - Scalability - policy extension – appraise/dont_appraise
  - HMAC or RSA signatures in xattr
  - Appraise directories – performance critical path
Summary

- IMA maintains trust chain in kernel
- Enabling IMA
- Using IMA
  - Standalone or with TPM
    - health check - PTS
    - Network admission - 802.1x-TNC-PTS
- Future Work
  - EVM
BACKUP
Introduction: State of Computer Security

“The sky isn't falling ... it fell a few years ago.”
Roger Grimes, Infoworld Security Advisor, 2006
The modern threat

“Nation states, however, have the technical and operational capabilities to orchestrate the full range of adversarial cyber operations through a combination of such means as recruiting insiders, setting up front companies, establishing signals collections systems, implanting damaging hardware or software in communications networks and subverting telecommunications, cryptographic defenses and supply chains.”

The Ease of Application Hacking

• Attacking Servers:
  – 97% web sites vulnerable to SQL injection or XSS.
    - IBM ISS

• Attacking Clients:
  – Chinese Hacking
    • Spear phishing with Word and PDF exploits
    • 1,295 (known) PC's in 103 Countries
    • High value targets
    • Remarkably simple, effective attacks
We've Lost the Software Vulnerability War

Vulnerabilities Discovered Per Year (CERT)

More and More Vulnerabilities (roughly 20 per day)

Less and Less Time to Patch (zero day exploits)
Secure Software is HARD

- So far, every software system has failed
  - Apollo Command Module Computer (16K words) failed every flight
- Studies shows at least 1 bug per K lines of code (LOC)
  - IBM internal study, 2000
  - Information Week Jan 21 2002, p23
  - Reasoning, Inc 2003
  - coverity.com 2008
- Linux and WinXP with Office each have > 200MLOC
  - 400K bugs would take 80 years @ 5000/year to fix
  - But we are writing roughly 50K new ones per year!
- Can model this as an infinite supply of security bugs.
  - Must design our systems to handle vulnerable software
The Failure of Secrecy

• “Three may keep a secret, if two of them are dead.”
  Benjamin Franklin, 1735

• Benjamin was a hopeless optimist.
  – Individuals seem delighted to give away their secrets.
  – Phishing/pharming
  – Gartner: $3.2B losses, 3.6M victims of phishing in 2007

• “One may keep a secret, if he doesn't know what it is.”
  Dave Safford, 2004   -   TPM
Hardware Based Security
IBM PCI-X Cryptographic Coprocessor

- Announced in September, 2003
- Greatly improved performance
- PCI-X and network interface
- Same physical / logical security feature set as 4758
- Received FIPS 140-2 Level 4 validation
- Support for IBM zSeries (mainframes) today
Secure Hardware

Integrity Aware Parallelizable Mode (IAPM)

- Originally developed for network communications
  - With “almost free” integrity
- Use “whitening” with pairwise independence
  - Builds “location sensitivity” into ciphertext
- Processed in parallel and/or pipelined engines
  - Both encryption & decryption
- Submitted to NIST for evaluation as a block cipher mode