Technologies to Improve Platform Security

Ernie Brickell
Intel Corporation

9/29/2011
Intel is positioning itself to lead in three areas:
- energy-efficient performance silicon,
- connectivity, and
- security.

There’s an urgent need for security innovation as people are spending more time online and the amount of data is growing.
Intel Security & Trust Pillars

Identity Protection & Fraud Deterrence

Detection & Prevention of Malware

Securing Data and Assets

Recovery and Enhanced Patching
A reusable circuit that provides an autonomous/self contained, complete DRNG

Provides a hardware source of high quality, high performance entropy to be embedded across Intel products. It is composed of

- An all-digital Entropy Source, (3 Gbps, 90% Entropic)
- Runtime Entropy Source health measurement via Online Health Test,
- Conditioning (via AES CBC-MAC mode) and DRBGing (via AES CTR mode) post processing and
- Built In Self Test (BIST) and Test Port

Standards compliant (NIST SP 800-90) and FIPS 140-2/3 Level 2 certifiable as such
RDRAND Performance

Preliminary data from pre-production Ivy Bridge sample

- RdRand – new CPU instruction which provides access to DRNG
- Up to 70 million RdRand invocations per second
- 500+ Million Bytes of random data per second
- Throughput ceiling is insensitive to number of contending parallel threads
  - Steady state maintained at peak performance

---

1Data taken from Intel® processor codename Ivy Bridge early engineering sample board. Quad core, 4 GB memory, hyper-threading enabled. Software: LINUX* Fedora 14, gcc version 4.6.0 (experimental) with RdRand support, test uses pthreads kernel API
RdRand Response Time
- ~150 clocks per invocation
- Little contention until 8 threads
  - (or 4 threads on 2 core chip)
- Simple linear increase as additional threads are added

DRNG Reseed Frequency
- Single thread worst case: Reseeds every 4 RdRand invocations
- Multiple thread worst case: Reseeds every 23 RdRand invocations
- At slower invocation rate, can expect reseed before every 2 RdRand calls
  - NIST SP 800-90 recommends $\leq 2^{48}$

\[\text{Number of Parallel Threads} \rightarrow \text{Response Time (Clock cycles)}\]

\[\text{Number of Parallel Threads} \rightarrow \text{DRNG Reseed Frequency}\]

\[\text{Number of Parallel Threads} \rightarrow \text{DRNG 128-bit Outputs}\]

\[\text{Data taken from Intel® processor codename Ivy Bridge early engineering sample board. Quad core, 4 GB memory, hyper-threading enabled. Software: LINUX® Fedora 14, gcc version 4.6.0 (experimental) with RdRand support, test uses pthreads kernel API}\]
Software Side Channels

- Not Hardware Side Channel where adversary has physical access.
- Not Software Covert Channel where adversary has malware in a high security partition and a low security partition.
- Software Side Channel – Adversary has malware executing in a spy process, and tries to obtain information about an uncompromised target process executing on same platform.

![Diagram showing the relationship between Target App with Protected Asset, Spy App, and OS. The Spy App is inside a red square, indicating it is outside the trust boundary. The Target App is inside a green square, indicating it is inside the trust boundary. The OS is also inside a green square.](image-url)
Protection from software side channels

- Platform approach for software side channels
  - AES-NI: CPU instructions for a round of AES
  - PCLMULQDQ: CPU instructions for GF(2) Multiplication
  - Recommend side channel mitigated implementations of other crypto algorithms
    - No secret key or data dependent
      - memory access (at coarser than cache line granularity)
      - code branching
    - Ex: RSA implemented with <6% performance reduction in OpenSSL
Crypto Performance

• Software improvements
  – Multi-buffer
  – Function Stitching

• Hardware improvements
  – AES-NI
  – PCLMULQDQ
  – Microarchitecture improvements
Multi Buffer Performance – 1 WSM Core

Multi-buffer: Perform the same function on multiple independent data buffers

Excellent performance on AES CBC Encrypt

*See Intel technical papers for full description of methodology and results.
Function Stitching

- Protocols such as SSL/TLS and IPsec apply two functions, confidentiality and integrity
- Improved performance by using multiple execution units more efficiently
- Fine grain integration achieves higher performance
- 1.4X Speedup on AES128 CBC-Encrypt with SHA1 (Cycles/Byte)

\[ T_{\text{Stitch}} < (T_A + T_B) \]

Method to speedup combined Encrypt/Authenticate
Sandy Bridge Performance

- SNB 2\textsuperscript{nd} Generation Intel® Core\textsuperscript{TM} improves:
  - AES-NI Throughput
  - SIMD Processing via AVX ISA extensions
  - Large-integer processing (public-key crypto)

- Multi Buffer Performance (Cycles/byte)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>i5-650</th>
<th>i7-2600</th>
<th>i7-2600 Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD5</td>
<td>1.46</td>
<td>1.27</td>
<td>1.15</td>
</tr>
<tr>
<td>SHA1</td>
<td>2.96</td>
<td>2.2</td>
<td>1.35</td>
</tr>
<tr>
<td>SHA256</td>
<td>6.96</td>
<td>5.27</td>
<td>1.32</td>
</tr>
<tr>
<td>AES128-CBC-Encrypt</td>
<td>1.52</td>
<td>0.83</td>
<td>1.83</td>
</tr>
</tbody>
</table>

- Modular Exponentiation Performance (Cycles)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>i5-650</th>
<th>i7-2600</th>
<th>i7-2600 Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>512-bit Modular Exponentiation</td>
<td>360,880</td>
<td>246,899</td>
<td>1.46</td>
</tr>
<tr>
<td>1024-bit Modular Exponentiation</td>
<td>2,722,590</td>
<td>1,906,555</td>
<td>1.43</td>
</tr>
</tbody>
</table>

1.2-1.8X additional performance gain on SNB!
### Summary of reduction in trust boundary by virtualization and measured launch

<table>
<thead>
<tr>
<th>Component in Trust Boundary</th>
<th>Mitigation with virtualization and measured launch</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS with kernel additions</td>
<td>Use VT-x and Require fewer kernel additions and device drivers in some VMs</td>
</tr>
<tr>
<td>Devices that can DMA</td>
<td>Restrict DMA to a single VM through VT-d</td>
</tr>
<tr>
<td>Apps installed by user</td>
<td>Restrict apps in protected VM, and sandbox suspect code</td>
</tr>
<tr>
<td>Virtualization layer underneath the OS</td>
<td>Allow only acceptable VMMs to launch using TXT launch control policy</td>
</tr>
<tr>
<td>BIOS, including SMM</td>
<td>Remove some or all of the BIOS from trust boundary using TXT and/or STM capability</td>
</tr>
<tr>
<td>Option ROMS</td>
<td>Remove Option ROMS from trust boundary using TXT</td>
</tr>
</tbody>
</table>
Intel Security & Trust Pillars

- Identity Protection & Fraud Deterrence
- Detection & Prevention of Malware
- Securing Data and Assets
- Recovery and Enhanced Patching
IPT 1.0: One Time Password (OTP)

The first generation of Intel® IPT is a dynamic code generated on 2nd generation Intel® Core™ processor-based PCs that is protected from malware in the OS.

- Single use, (i.e. 30 second, time-limited code → OTP )
- A hardware level 2nd factor of authentication
- Works with leading OTP Solutions from Symantec & Vasco

No system can provide absolute security under all conditions. Requires an Intel IPT enabled system, including a 2nd generation Intel Core processor, enabled chipset, firmware and software. Available only on participating websites. Consult your system manufacturer. Intel assumes no liability for lost or stolen data and/or systems or any resulting damages. For more information, visit [http://ipt.intel.com].
Intel Security & Trust Pillars

- Identity Protection & Fraud Deterrence
- Detection & Prevention of Malware
- Securing Data and Assets
- Recovery and Enhanced Patching

Intel Architecture Group
Supervisor Mode Execution Protection (SMEP)

• Prevents attacks when executing user-mode code in ring-0
  – Extends Intel eXecute Disable capability

• Available on Intel CPUs starting 2012

• Example where SMEP benefits: ‘Stuxnet’ worm
  – SMEP would have prevented one method of attack by ‘Stuxnet’ –> Escalation of Privilege attack

How does SMEP work?

SMEP can prevent malware exploiting EoP vulnerabilities from executing
McAfee DeepSAFE

- Technology platform co-developed with Intel
  - Not a product, the foundation for new solutions
- Hardware-assisted, security-focused, system monitor
- Sits below the OS to provide a new vantage point for security
- Solutions to be announced soon.
Intel Security & Trust Pillars

- Identity Protection & Fraud Deterrence
- Detection & Prevention of Malware
- Securing Data and Assets
- Recovery and Enhanced Patching
Recommend BIOS Hygiene

• NIST SP800-147 – BIOS protection Guidelines
  – Use digital signatures to verify the authenticity of BIOS updates.
  – BIOS updates verified using a Root of Trust for Update which includes:
    – The key store used to verify signatures on updates.
    – The digital signature verification algorithm.
  – Use of NIST-approved crypto algorithms.
  – Recommend rollback protection.

• Minimize TCB for system boot
Intel Security & Trust Pillars

Identity Protection & Fraud Deterrence
Detection & Prevention of Malware
Securing Data and Assets
Recovery and Enhanced Patching

Stay tuned for future improvements in all pillars
Feedback

• What else should Intel be doing?
• Are we on the right track?
Technical Papers - 1

Breakthrough AES performance with Intel AES New Instructions
http://software.intel.com/file/26898

Processing Multiple buffers in parallel

Fast Cryptographic computation on IA processors via Function Stitching
http://download.intel.com/design/intarch/PAPERS/323686.pdf

Fast and Constant-time Implementation of Modular Exponentiation

Fast CRC Computation for iSCSI Polynomial using CRC32 Instruction

Optimized Galois-Counter-Mode Implementation on IA Processors
http://download.intel.com/design/intarch/PAPERS/324194.pdf

High Performance Storage Encryption on Intel® Architecture Processors
http://download.intel.com/design/intarch/PAPERS/324310.pdf
Fast CRC Computation for Generic Polynomials using PCLMULQDQ Instruction
http://download.intel.com/design/intarch/papers/323102.pdf


Cryptographic Performance on the 2\textsuperscript{nd} Generation Intel Core Processor http://download.intel.com/design/intarch/PAPERS/324952.pdf

Fast Parallel CRC Computation using the Nehalem CRC32 instruction http://drdobbs.com/cpp/229401411

