Not-So-Random Numbers in Virtualized Linux and the Whirlwind RNG

Adam Everspaugh, Yan Zhai, Robert Jellinek, Thomas Ristenpart, Michael Swift
University of Wisconsin — Madison
Random Number Generators

Inputs
system events

RNG

Outputs
unpredictable
Random Number Generators

Inputs
- system events

Previous Failures

RNG

Outputs
- unpredictable

Cryptanalysis of Windows RNG [DGP07]

Linux RNG [GPR08]

Factorable RSA Keys [HDWH12]

Linux RNG Revisited [LRSV12]

/dev/random not Robust [DPRVW13]

Taiwan National IDs [BCCCHLS13]
RNGs in Virtual Environments

Inputs  
\[\text{system events}\]  
\[\text{virtual machine}\]

RNG

Outputs  
\[\text{unpredictable}\]

1. Are there operational issues that cause problems for system RNGs? [GR05] [RY10]

2. Are input sources entropy-poor inside a virtual machine? [SBW09]
Our Contributions

• First study of system RNGs in modern virtualized settings

• Operational issues? -> YES
  Bad RSA keys from OpenSSL

• Entropy-poor inputs? -> NO

• New clean-slate RNG design — Whirlwind
VM Use Cases

Boot-from-image

disk

Snapshot

Resumption

Amazon EC2
Rackspace
Microsoft Azure

Snapshot-Reset

disk

Snapshot

Resumption

vmware
Xen
Parallels
Security Problems with VM Resets

VM Reset Vulnerabilities [GR05] [RY10]

App starts /dev/urandom

Initialization

Derives key

Read

Use key

[RY10] Suggested countermeasure:
Narrow gap between deriving and using random numbers

Are system RNGs reset secure?
Linux RNG Not Reset Secure

Experiment
• Boot VM, idle for 5 minutes
• Start measurement process, capture snapshot
• Resume from snapshot, read 512-bits from /dev/urandom every 500 us

Repeat for 8 snapshots; 20 resumptions/snapshot

Result: 7/8 snapshots generated 1 or more identical 512-bit output
Reset Vulnerabilities on Other Platforms

FreeBSD
/dev/random produces repeat outputs
Up to 100 seconds after reset

Microsoft Windows 7
Produces repeat outputs indefinitely
rand_s (stdlib)
CryptGenRandom (Win32)
RngCryptoServices (.NET)
Our Contributions

• First study of system RNGs in modern virtualized settings

• Operational issues?  -> YES

• Entropy-poor inputs?

• New clean-slate RNG design — Whirlwind
Our Contributions

• First study of system RNGs in modern virtualized settings

• Operational issues? -> YES

• Entropy-poor inputs?

• New clean-slate RNG design — Whirlwind
Estimating Input Entropy

- Instrumented Linux RNG
- Collected all inputs, outputs on boot
- Gathered data from: native, Xen, VMware, and EC2
- Statistical hypothesis testing to determine entropy count per input
# Results: Boot Security

<table>
<thead>
<tr>
<th>Output #</th>
<th>Native</th>
<th>Xen</th>
<th>VMware</th>
<th>EC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>129</td>
<td>129</td>
<td>784</td>
<td>134</td>
</tr>
</tbody>
</table>

Entropy estimate ($\log_2$) for Linux `/dev/(u)random` during boot:

No inputs before first output: constant value
Our Contributions

• First study of system RNGs in modern virtualized settings

• Operational issues? -> YES

• Entropy-poor inputs? -> NO

• New clean-slate RNG design — Whirlwind
Our Contributions

• First study of system RNGs in modern virtualized settings

• Operational issues? -> YES

• Entropy-poor inputs? -> NO

• New clean-slate RNG design — Whirlwind
Whirlwind RNG

Goals
1. Simplicity
2. Fast Input Processing
3. Cryptographically Sound
4. Drop-in Compatibility
5. Reset Security
Whirlwind RNG

Goals

1. Simplicity
2. Fast Input Processing
3. Cryptographically Sound
4. Drop-in Compatibility
5. Reset Security

Use environmental information (in addition to state data) when generating outputs.

-> Prevents reset vulnerabilities
Conclusions

• Linux, FreeBSD, and Windows are vulnerable upon snapshot resumption

• Linux /dev/(u)random has boot-time entropy hole

• Virtual settings have sufficient entropy

• Whirlwind RNG gives reset security by design