Complexity in the Cloud

- App
- OS
- Policy
- Guidance
- Best Practices
- App Virt
- Web Service
- OS
- Hypervisor
- BLADE
- SAN

Coherence

Security Posture and Behavior Coupling

Governance/Risk

Workload

EC2

Risk
Fabric: Lots of Configuration!
The cloud is very different

- Deeper stacks
- ... each layer has its own vulnerabilities
- More intimately coupled
- More dynamic workloads
- Multi-tenant
- ... each with different (evolving) governance
- ... under potentially different (evolving) regulatory domains
- ... accountable for different (evolving) due care
But... (Variation on the Gartner 7)

- Am I compliant? (at every level in any state)
- Trust Stack: Physical or Logical or.....
- What is shared? (coupling)
- Where is the problem? (context via connect the dots)
- How well is my deployment working? (at all levels)
- How should I re-provision? (next desired state)
- How can I improve? (good citizens vs. problem children)

- Issue: Black Box Abstraction of Complex Activity:
  - Can’t Manage what you can’t measure. Drucker
  - Can’t Measure what you can’t see. Deming
  - => Automation, of any kind, without feedback inevitably does the wrong thing very efficiently.
Solution Direction: Visibility

- Visibility vs. Transparency
- Configuration Visibility
  - Sec/Op Relationships
  - Trust/health X tenant
  - Root Cause
- Behavioral Visibility
  - Root/Cause
  - Alignment
  - Improvement

"You can't manage what you can't measure", Drucker
"You can't measure what you can't see", Deming
Solution Direction: Models

- Model-based controls (SML, OVF, OSLO, SDM, UCA …)
  - Tie constraints to intentional relationships
  - Service lifecycle: design – de-provisioning
  - Dynamics (autonomics)
  - Inform “next desired state” (design impact of change)
  - XCCDF – OVAL, … but in model vocabulary
Solution Direction: Small is Good

- (much) Smaller Virtualization Kernels
  - Hyper Guard, sHype, Flask, …
Appendix
Virtualization Specific Vulnerabilities

<table>
<thead>
<tr>
<th>Vulnerability Summary CVE-2008-1944</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Original release date:</strong> 5/14/2008</td>
</tr>
<tr>
<td><strong>Last revised:</strong> 6/4/2008</td>
</tr>
<tr>
<td><strong>Source:</strong> US-CERT/NIST</td>
</tr>
</tbody>
</table>

**Overview**

Buffer overflow in the backend framebuffer of XenSource Xen Para-Virtualized Framebuffer (PVFB) Message 3.0 through 3.0.3 allows local users to cause a denial of service (SDL crash) and possibly execute arbitrary code via "bogus screen updates," related to missing validation of the "format of messages."

**Impact**

None specified.

**Vulnerable software and versions**

**Configuration 1**
- Xensource, Xen, 3.0
- Xensource, Xen, 3.0.3
- Running on Redhat, Desktop, 5
- Running on Redhat, Enterprise_linux, 5, Unknown, Client
- Running on Redhat, Enterprise_linux, 5, Unknown, Server
- Running on Redhat, Virtualization_server, 5
Virtualization Specific Vulnerabilities
XenSploit

Resulting Guidance:
- Encrypt Dynamic Migration channels
- Restrict access
- Tightly control vNIC configuration
- Isolate LANs (Management, Transactional, Dynamic Migration)

Proposes using vulnerabilities, like CVE-2007-4993, CVE-2007-5497 to gain root in dom0 from unprivileged dom.

Stop packet queue with kernel function netif_tx_disable()

Using DMA to create a backdoor:

Reading: set a transmit ring entry so that the data pointer points to <arb addr>, and the receive ring entry data pointer points to buffer we can read

Writing: set a transmit ring entry so that the data pointer points to our data, and the receive ring entry data pointer points to <arb addr>

Can be implemented as a kernel module that gets the address dev get by name() macro

Demo code works for all NIC cards supported by the Linux tg3.c driver.

Addresses bypassing IOMMU and VT-d...
Server Virtualization
Vulnerabilities

<table>
<thead>
<tr>
<th>2005</th>
<th>2006</th>
<th>2007</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2005-4459</td>
<td>CVE-2006-2481</td>
<td></td>
</tr>
<tr>
<td>CVE-2005-4082</td>
<td>CVE-2006-2662</td>
<td>CVE-2007-1876</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CVE-2007-1270</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CVE-2007-0948</td>
</tr>
</tbody>
</table>

So monitoring emergent configuration controls is always necessary.