Non-intrusive, Out-of-band and Out-of-the-Box Systems Monitoring in the Cloud

SIGMETRICS
June 18, 2014

Canturk Isci
Sahil Suneja
Eyal de Lara
University of Toronto

Vasanth Bala
Todd Mummert
IBM T.J. Watson Research
Data Center Machines

VMs = new processes for the cloud computer!

Traditional

Modern

VIRTUALIZATION LAYER

HARDWARE

OS

HARDWARE

VM

OS

VM

VM

VM
Traditional Systems Monitoring
Traditional Systems Monitoring

![Diagram showing Traditional Systems Monitoring with layers: Hardware, Virtualization Layer, Virtual Machines (VMs)].
Introducing - Near Field Monitoring
Near Field Monitoring (NFM)
NFM's Advantages

- **Always-on:** Works for unresponsive or compromised systems
- **Out-of-the-box:** Unmodified guest
  No agent or hook installation
- **Non-intrusive:** No guest cooperation
  No interference with guest operation
- **Out-of-band:** Outside guest's context
  Decouple execution and monitoring
- **Virtualization-aware:** Holistic knowledge
  Accurate and efficient monitoring
NFM's Architecture
1. Exposing VM Memory State
   - Gain access to VM’s memory image from outside
     - Unmodified VM
     - Unmodified hypervisor

2. Exploit VM Memory State
   - Reconstruct VM's runtime state from the memory image
   - In-memory kernel data structure traversal
Approach | Exposing VM Mem State

- **Memory dump**
  - Dump / migrate guest memory to file
  - KVM-QEMU *pmemsave* or *migrate*-to-file
  - High overhead: VM paused for dump duration

- **Live R/O memory handle**
  - Xen
    - Map guest memory into crawler process- *xc_map_foreign_range()*
  - KVM
    - No default support
    - New live handle, read VM mem directly via
      - QEMU process' `/proc/<pid>/mem + /proc/<pid>/maps`
    - Negligible impact on VM
Approach | Exploiting VM Mem State

- Extract system information by traversing linux kernel's C structs in exposed memory image
  - Different structs for different kinds of information
    - `task_struct`, `mm_struct`, `files_struct`, `net_device` etc.

- Requirements:
  - Starting addresses for structs
    - `/boot/System.map`
  - Offsets for particular struct fields
    - Linux source or `vmlinux`
    - `/boot/<Build.config>`
Cloud Analytics

Crawl Logic

Frames

Frame Datastore
Structured view of VM states

APP

Analytics Apps

VM Mem/Disk handle

CPU NumCores, Hz, CacheSize, ...
OS Nodename, Release, Arch, ...
N/W device HWaddr, Ipaddr, TX/RX bytes, ...
Modules Name, State, ...
Process PID, Command, RSS, ...
Open files FD \(\rightarrow\) filename, ...
Memory Mapping MappedFiles, VA \(\rightarrow\) PA mappings, ...
N/W connections SocketState, \{Src, Dst, Ports\}, ...
1. **CTop**: Cloud-wide consolidated resource monitoring

2. **PaVScan**: Hypervisor paging aware virus scanner

3. **RConsole**: Remote console

4. **TopoLog**: Network topology discovery
Evaluating NFM

- Latency / monitoring frequency?
- Accuracy?
- Overhead?
- Advantages?
NFM's High Monitoring Frequency

- Safe: 10Hz
- KVM: 20Hz
- Xen: 200Hz
NFM's Accuracy: Cloud Top vs. top

```
NFM's Accuracy: Cloud Top vs. top

---

```

---

```

Every 0.5s: ./topUpdate.sh

CPU up time: 4461430125 jiffies

```

---

```

16
```
NFM's High Accuracy

<4% variation
NFM's Low VM Overhead

+ 256MB WS in paper
NFM's Advantages:
Analyze Dysfunctional Systems

- **Via RConsole** - Out-of-band console-like R/O interface
- **Supported functions:** `ls`, `lsmod`, `ps`, `netstat`, `ifconfig`, ...
- **Time travel:** `sync` and `seed` APIs
- **Analyzes unresponsive systems:** kernel panic, misconfigured n/w
- Detects (some) rootkits:

<table>
<thead>
<tr>
<th>In-VM Console:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active Internet connections (servers and established)</strong></td>
</tr>
<tr>
<td><strong>Proto</strong></td>
</tr>
<tr>
<td>tcp</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RConsole:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active Internet connections</strong></td>
</tr>
<tr>
<td><strong>Proto</strong></td>
</tr>
<tr>
<td>tcp</td>
</tr>
<tr>
<td>tcp</td>
</tr>
</tbody>
</table>
NFM's Advantages: Better Accuracy

- Distributed Application
  - 3 LAMP instances

<table>
<thead>
<tr>
<th></th>
<th>VM1</th>
<th>VM2</th>
<th>VM3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reservation</td>
<td>30%</td>
<td>30%</td>
<td>30%</td>
</tr>
<tr>
<td>Allocation</td>
<td>100%</td>
<td>70%</td>
<td>30%</td>
</tr>
</tbody>
</table>
NFM's Advantages: Better Accuracy

NFM's holistic view enables more accurate monitoring
Conclusion

- Current monitoring techniques unfit for modern virtualized Cloud

- Introducing Near Field Monitoring - Leverage virtualization for a fundamentally different VM monitoring approach
  - Eliminates in-VM hooks, provides same fidelity monitoring out-of-band
  - Decoupled VM monitoring - execution architecture
  - Alleviates concerns with existing techniques
    - Always-on, non-intrusive, holistic view, ...

- Evaluation:
  - High frequency
  - Low overhead
  - Better accuracy
  - Higher efficiency