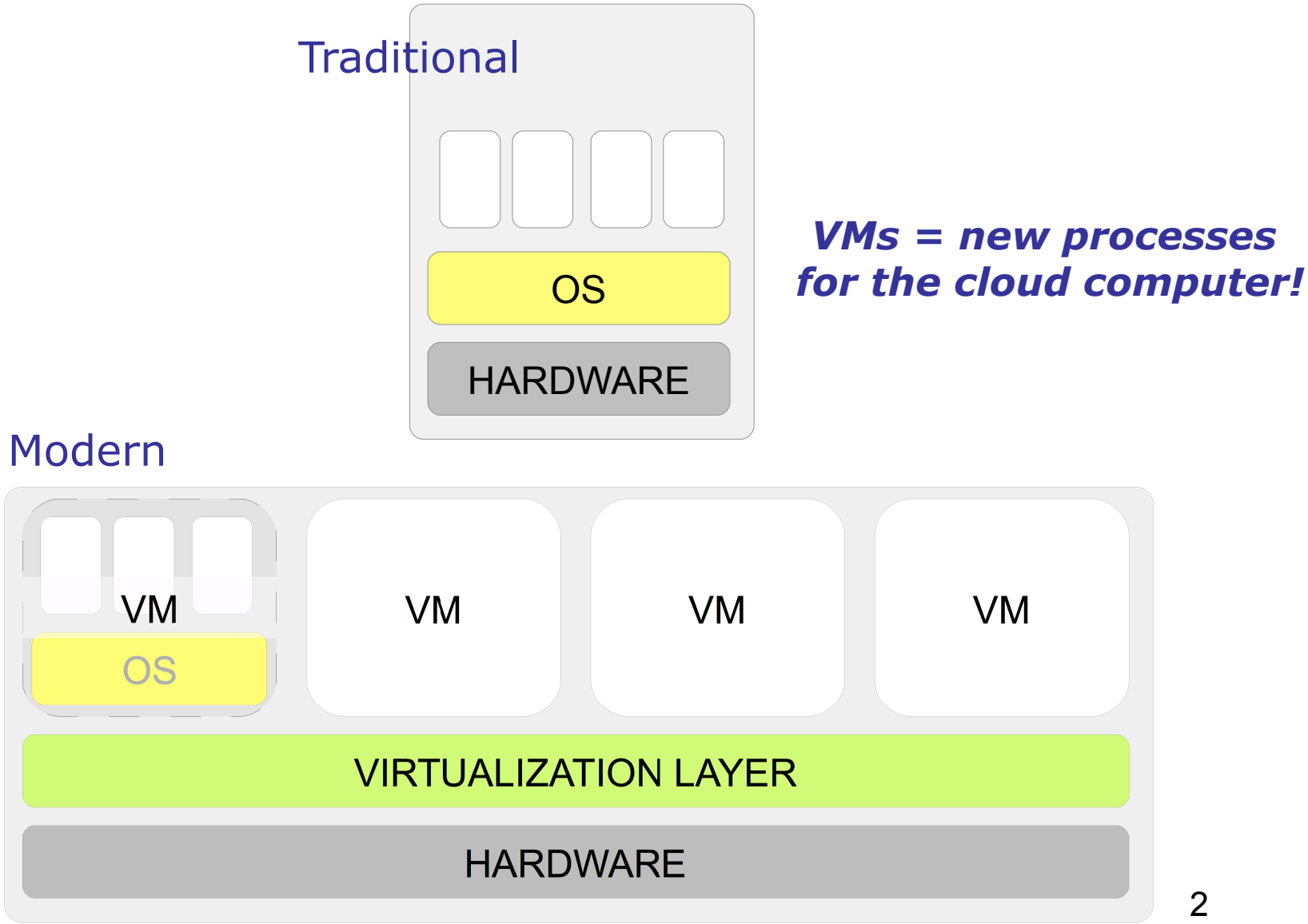
Non-intrusive, Out-of-band and Out-of-the-Box Systems Monitoring in the Cloud

SIGMETRICS
June 18, 2014

Canturk Isci
Vasanth Bala
Todd Mummert
IBM T.J. Watson Research

Sahil Suneja
Eyal de Lara
University of Toronto

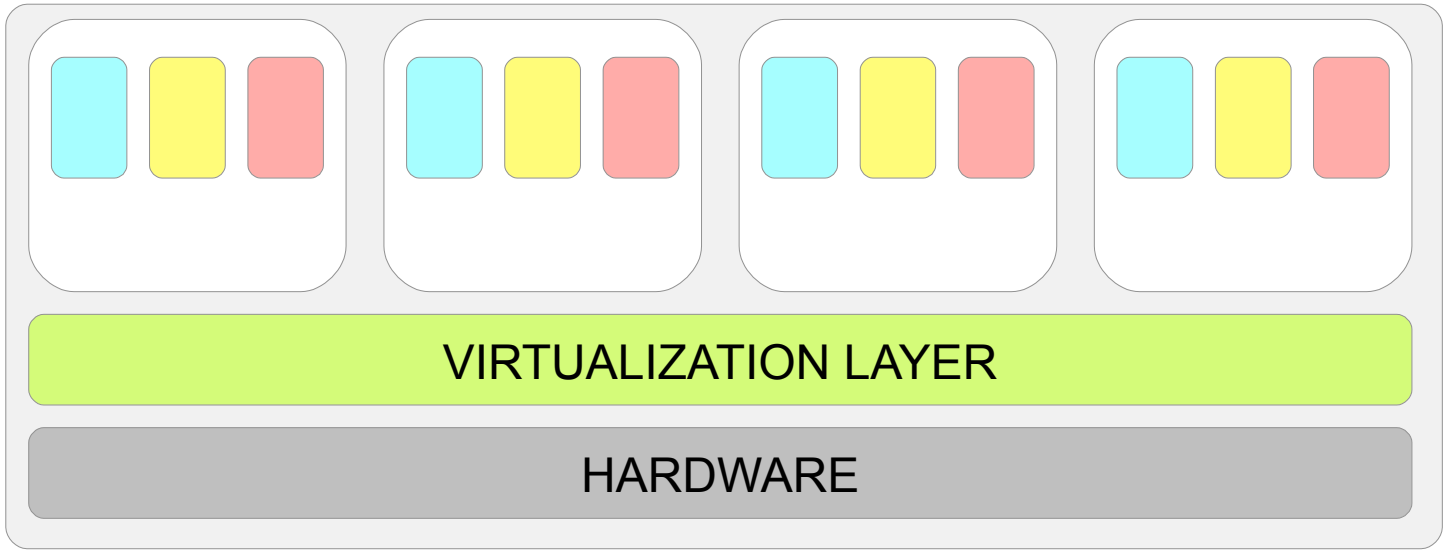
Data Center Machines



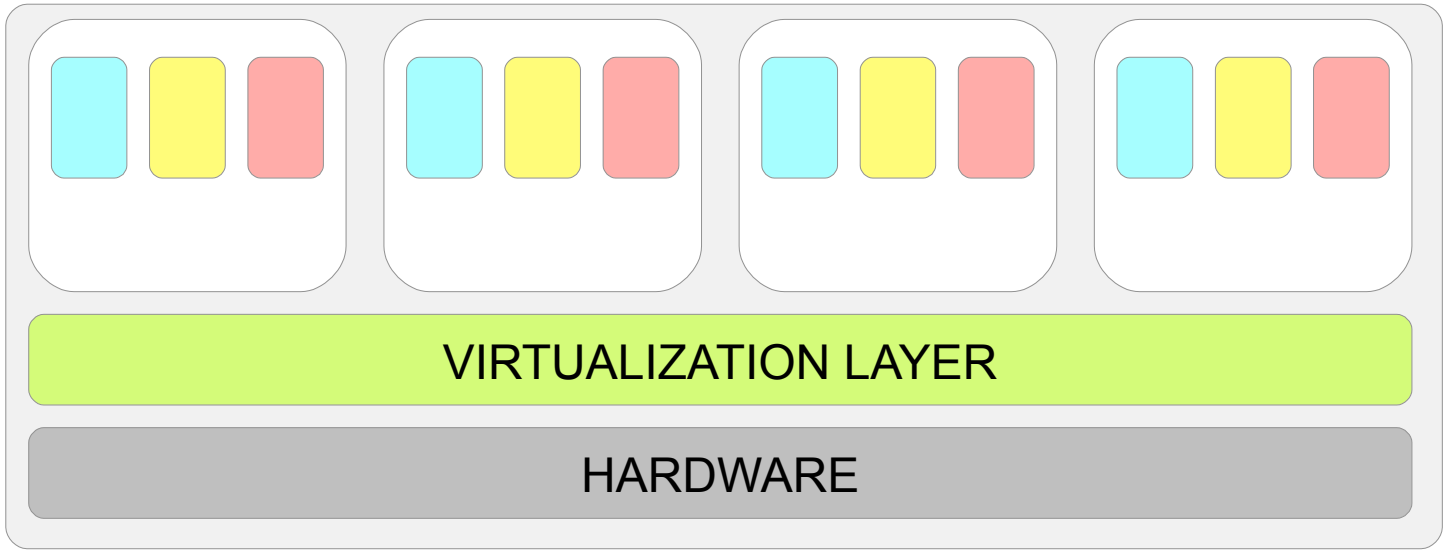
VMs = new processes for the cloud computer!

Traditional Systems Monitoring

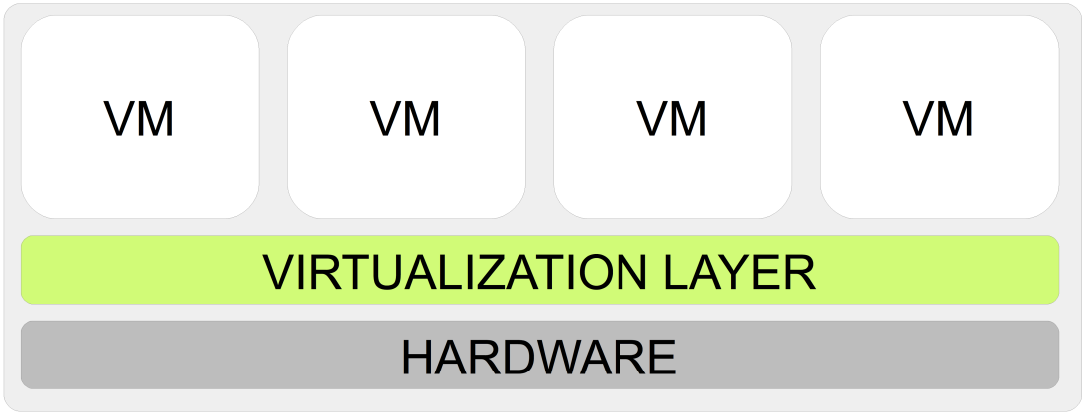
Traditional Systems Monitoring



Introducing- *Near Field Monitoring*



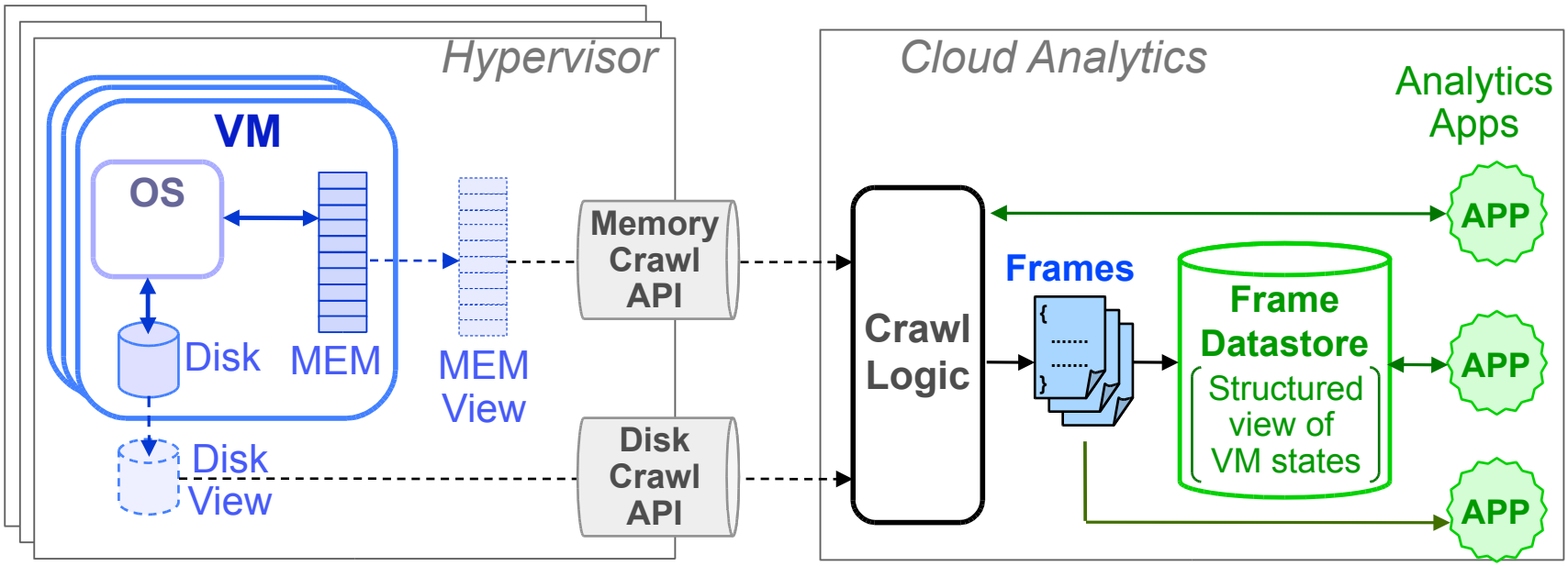
Near Field Monitoring (NFM)



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

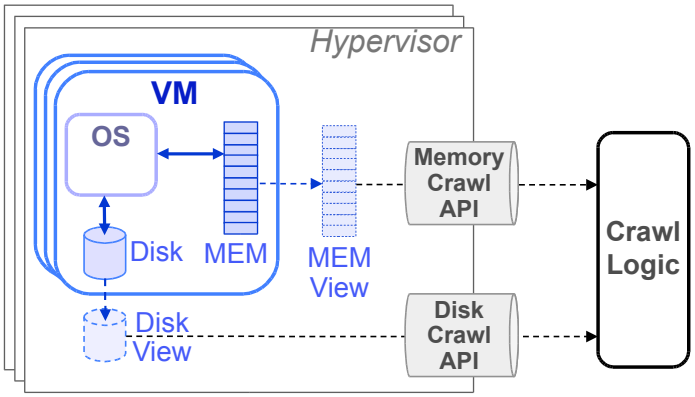
NFM's Architecture



Frontend

Backend

Approach: VM Memory Introspection



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

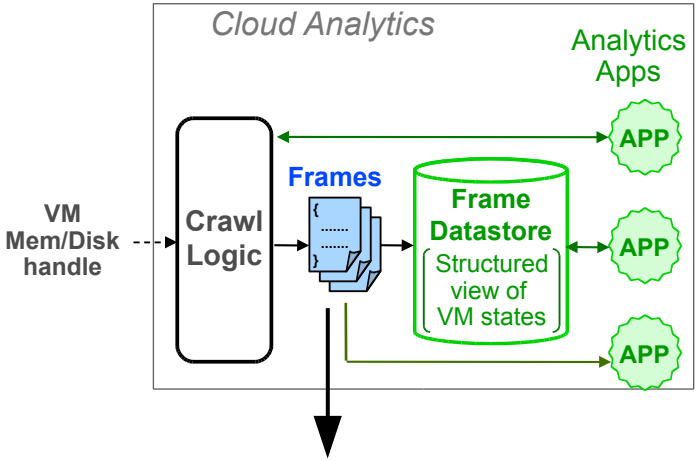
- Memory dump
  - Dump / migrate guest memory to file
  - KVM-QEMU *pmemsave* or *migrate*-to-file
  - High overhead: VM paused for dump duration
- Live R/O memory handle
  - Xen
    - Map guest memory into crawler process- *xc_map_foreign_range()*
  - KVM
    - No default support
    - New live handle, 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>

Backend | Crawl Output



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 → filename, ...
Memory Mapping	MappedFiles, VA → PA mappings, ...
N/W connections	SocketState, {Src, Dst, Ports}, ...

Backend | Prototype Apps

1. *CTop*: Cloud-wide consolidated resource monitoring

2. *PaVScan*: Hypervisor paging aware virus scanner

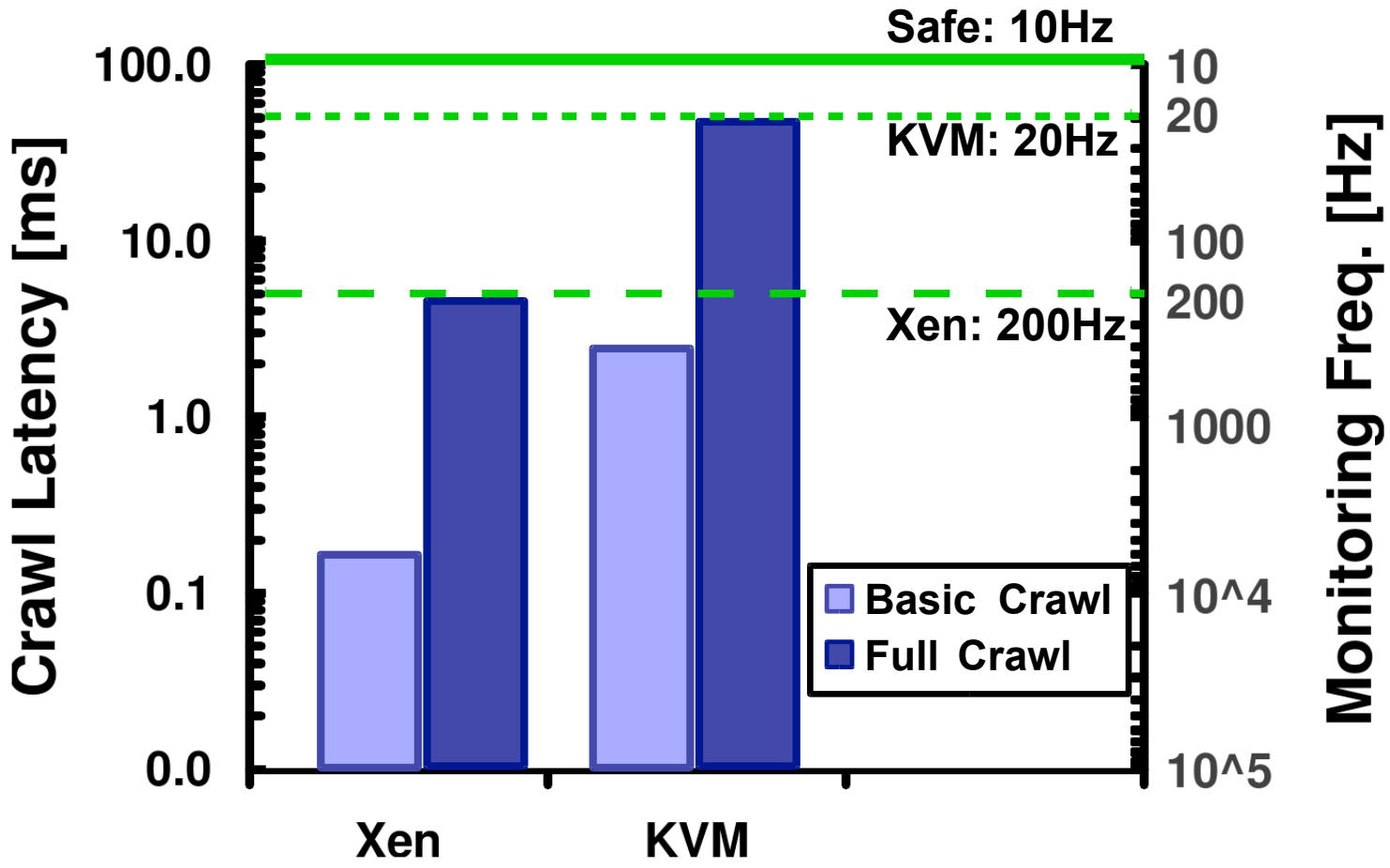
3. *RConsole*: Remote console

4. *TopoLog*: Network topology discovery

Evaluating NFM

- Latency / monitoring frequency?
- Accuracy?
- Overhead?
- Advantages?

NFM's High Monitoring Frequency



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

```
top - 11:58:42 up 1 day, 22:19,  1 user, load average: 0.90, 0.22, 0.11
Tasks: 57 total,   3 running,    54 sleeping,  0 stopped,    0 zombie
Cpu(s): 99.0%us, 0.0%sy, 0.0%ni, 0.0%id, 0.0%wa, 0.7%hi, 0.0%si, 0.3%st
Mem:  2052104k total, 1976340k used,   75764k free,     3996k buffers
Swap: 6160380k total,  304068k used, 5856312k free,     1868k cached
```

top

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1942	root	20	0	1028m	1.0g	188	R	49.9	51.0	0:08.98	malloc
1940	root	20	0	1028m	780m	136	R	49.5	38.9	0:11:91	malloc
1	root	20	0	56220	1164	408	S	0.0	0.1	0:00.71	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd

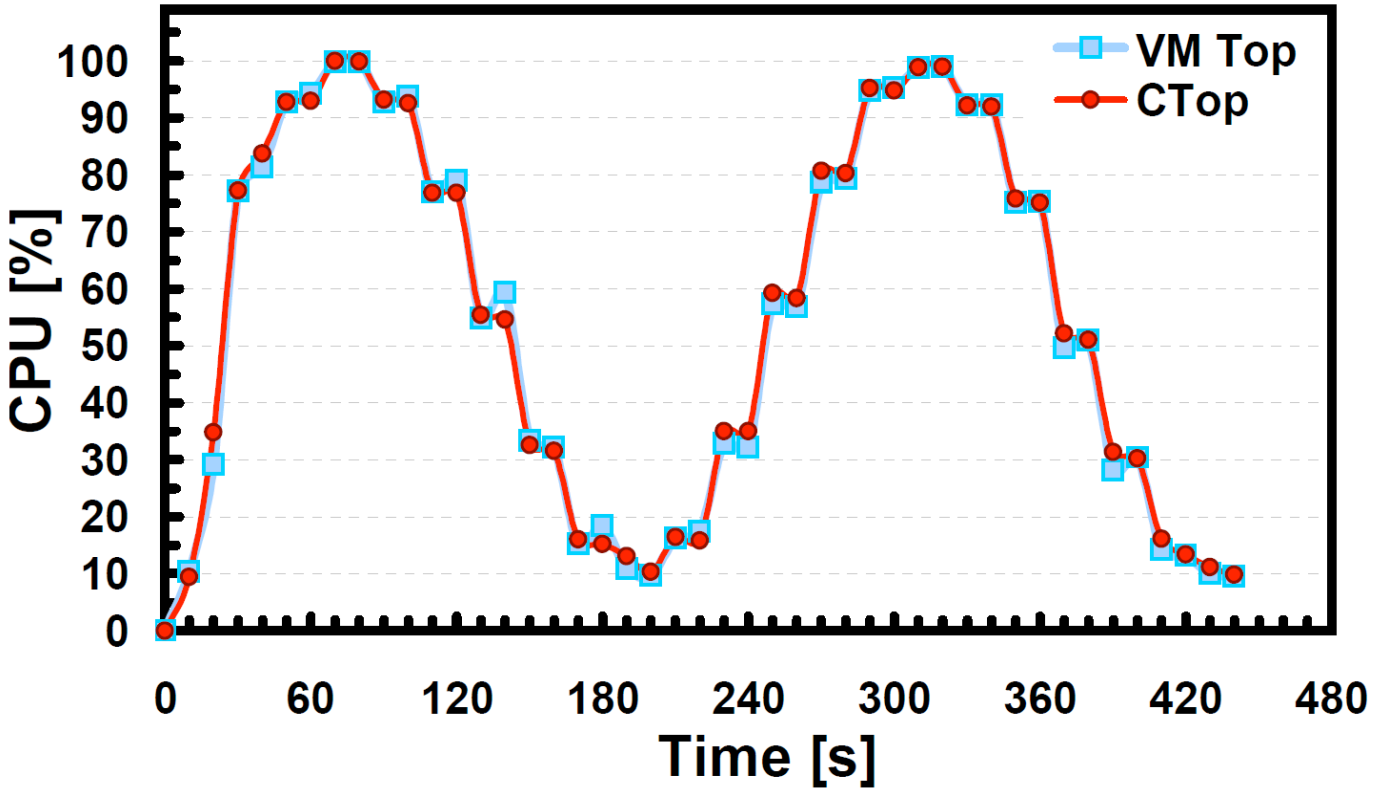
```
Every 0.5s: ./topUpdate.sh

CPU up time: 4461430125 jiffies
```

cTop

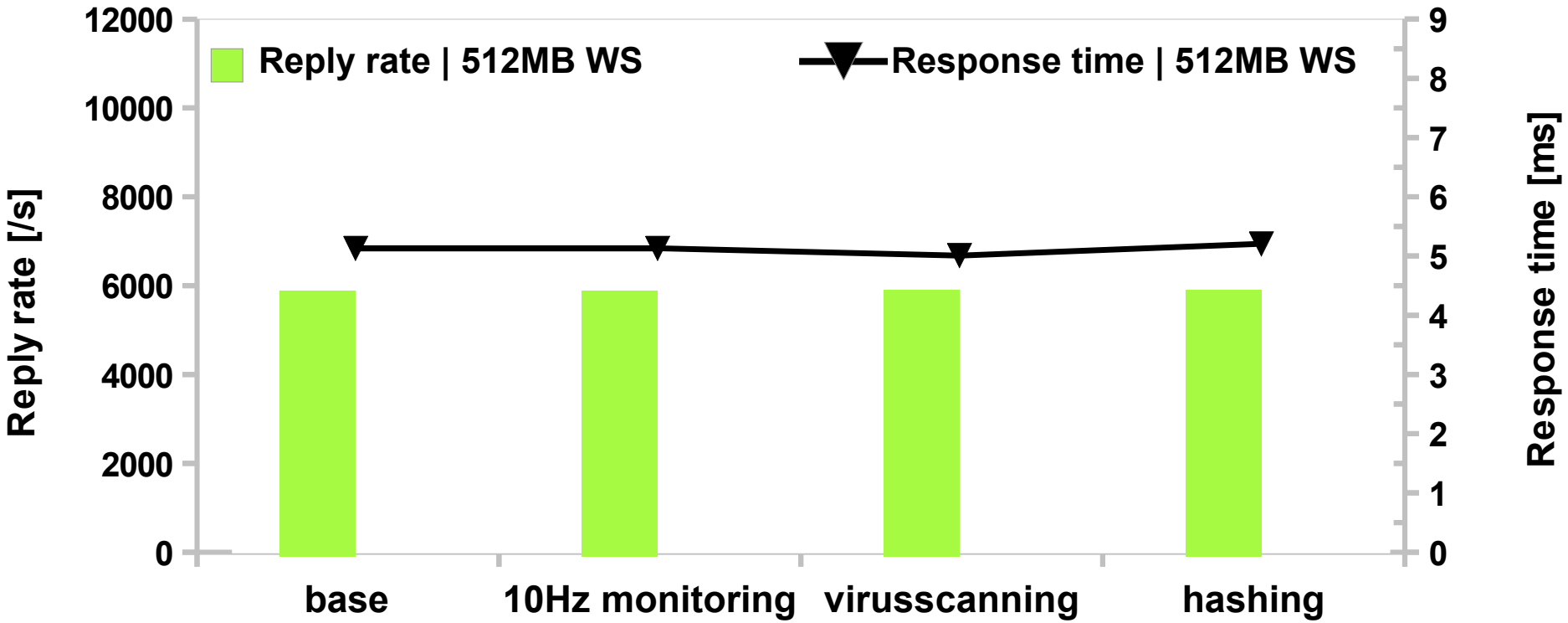
PID	VIRT	RES	%CPU	%MEM	TIME+	COMMAND
1942	1052704KB	1047368KB	45.8	51.0	0:08:33	malloc
1940	1052704KB	798816KB	45.8	38.9	0:11.92	malloc
1	56220KB	1164KB	0.0	0.1	0:00.70	systemd
2	0	0	0.0	0.0	0:00.00	kthreadd
:						

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:

In-VM Console:

```
Active Internet connections (servers and established)
Proto Local Address        Foreign Address    State
tcp   127.0.0.1:25         0.0.0.0:*          LISTEN
tcp   9.XX.XXX.110:52019   9.XX.XXX.109:22    ESTABLISHED
:
tcp   9.XX.XXX.110:22      9.XX.XXX.15:49845 ESTABLISHED
```

RConsole:

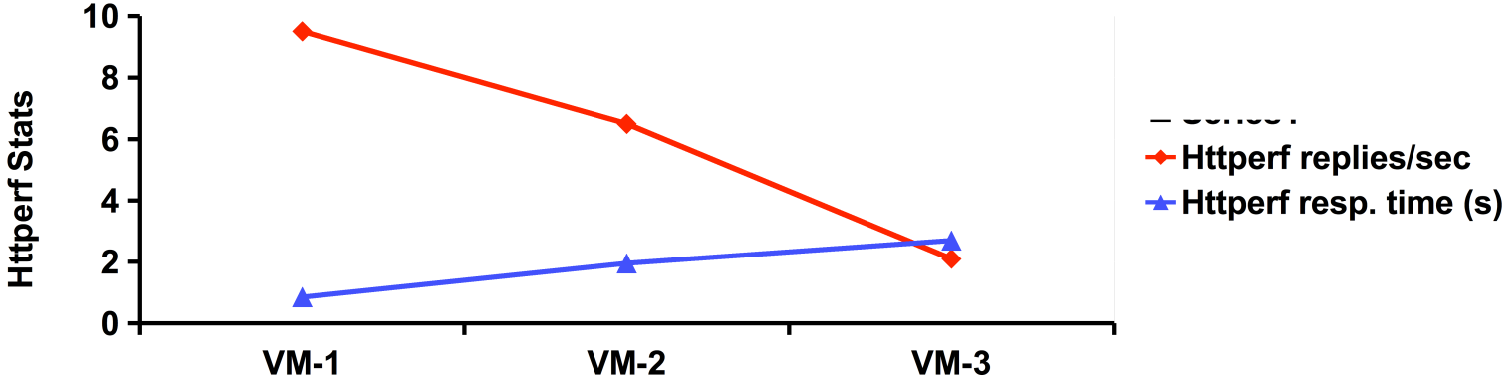
```
Active Internet connections
Proto   Local Address       Foreign Address    State             PID Process
tcp     127.0.0.1:25        0.0.0.0:0          SS_UNCONNECTED    741 [sendmail]
tcp     9.XX.XXX.110:52019  9.XX.XXX.109:22    SS_CONNECTED     6177 [ssh]
:
tcp     9.XX.XXX.110:22     9.XX.XXX.15:49845  SS_CONNECTED    14894 [sshd]
tcp     0.0.0.0:2476        0.0.0.0:0          SS_UNCONNECTED  23304 [datacpy]
```

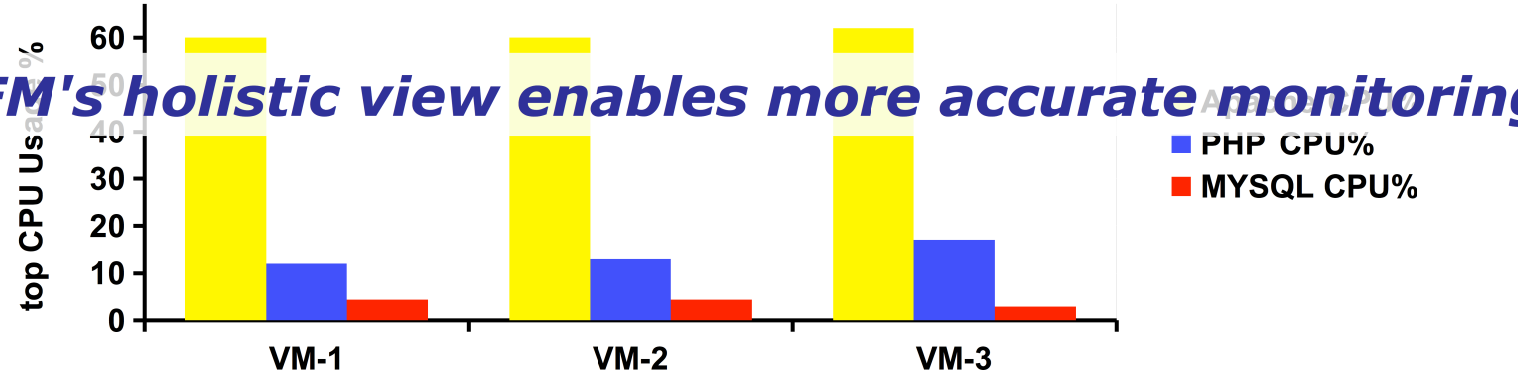
NFM's Advantages: Better Accuracy

- Distributed Application
  - 3 LAMP instances

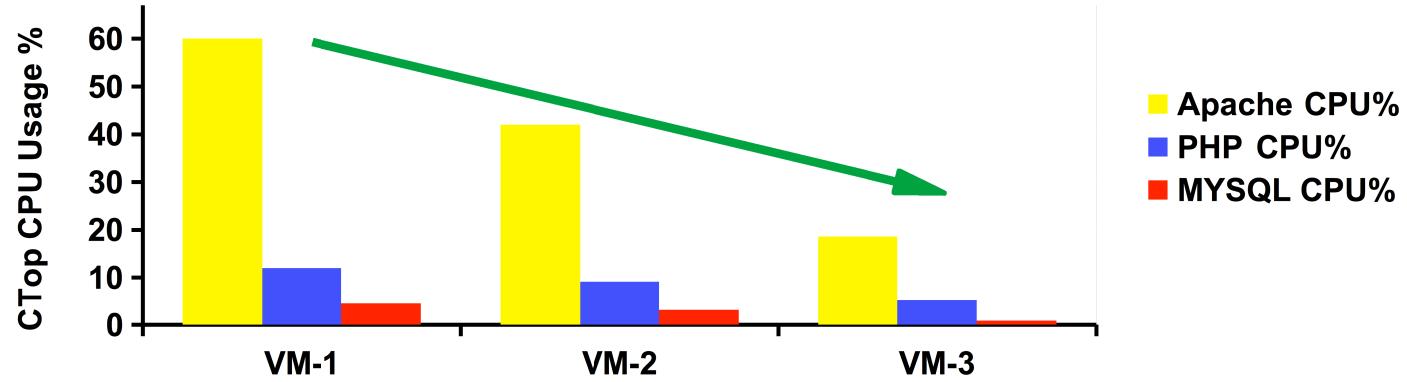
	VM1	VM2	VM3
Reservation	30%	30%	30%
Allocation	100%	70%	30%

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