
COST 1304 MC meeting, Warsaw, Poland, May 2016.

Kurt Tutschku*, Vida Ahmadi Mehri*, Anders Carlsson*, Krishna Chivukula†, Johan Christenson†

*Blekinge Institute of Technology (BTH), Faculty of Computing, Department of Communication Systems (DIKO) Karlskrona, Sweden

†City Network Webbhotell AB Karlskrona, Sweden
What are the Issues?

- Cloud as a Service, e.g. CityConnect

- Efficient sharing resources – exploiting economy of scale in DCs
- Simplicity in administration
- Use of standard hard- and software
- Smartness founded in use of standard tools
What are daily tasks in performance management in virtualized environment?

- Quality/load relationship between “inside (virtual appliance/user)” and “outside (infrastructure/admin)” is not obvious.
- Different operating systems, numerous on-board monitoring tools, various hypervisors
- What are the capabilities of these software components and tools?
What are the Resource Description Capabilities of Onboard Tools?

1. Is there a predictable, scaling i.e. strong positive correlation of load and utilization between guest and host?  
   → Here, "scaling" means, whether an increase in guest load results in a predictable increase of the host.

2. How strong do the observed utilization values vary when observing them during a small sampling interval? Does the accuracy of the prediction depend on guest load levels?

3. Does the scaling depend on the number of stressed vCPUs, i.e. on the resources available to the VM?

4. Does the scaling depend on host operating systems?

5. Are there significant differences between the monitoring tools?
### Resource Types

- **Assumption:** virtualization is done by VMs → `virt-install` command options

<table>
<thead>
<tr>
<th>Area</th>
<th>Option</th>
<th>Description</th>
<th>Used here</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td><code>-vcpus</code></td>
<td>Number of vCPUs</td>
<td>X</td>
</tr>
<tr>
<td>Memory</td>
<td><code>-ram</code></td>
<td>RAM to allocate (MB)</td>
<td></td>
</tr>
<tr>
<td>Storage</td>
<td><code>-disk</code></td>
<td>specify storage media</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>-filesystem</code></td>
<td>export a host directory to guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>-file</code></td>
<td>Installation media location</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>-location</code></td>
<td>Installation via distribution tree</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td><code>-network</code></td>
<td>Host network</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td><code>-os-type</code></td>
<td>OS type</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>-os-variant</code></td>
<td>OS variant/version</td>
<td></td>
</tr>
<tr>
<td>I/O</td>
<td><code>-graphics</code></td>
<td>Graphical display method</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>-nographics</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td><code>-name</code></td>
<td>Virtual machine name</td>
<td></td>
</tr>
</tbody>
</table>

→ Probably higher-dimensional resource descriptions necessary!
Resource Description

- Notions for CPU load and utilization
  a. Load:
     - Concurrent number of processes using the computational resources
     - Computed by sum of running and waiting threads
     - Sometime: between 0 and the numbers of available CPU cores
     - Typically: weighted average of served processes at subintervals during a measurement
  
  b. Utilization:
     - Ratio of accumulated CPU time within observation time interval
     - Each tool applies its own calculation and may not necessarily calculate similar
     - Typically a weighted average of the number of served processes at subintervals during a measurement.
Onboard CPU Resource Monitoring Tools

- **MPstat:**
  - Part of sysstat utilities
  - Used for utilization monitoring
  - Command: `mpstat -P ALL T1 T2`
  - Output: percentages of CPU utilisation at %user/%system/%iowait levels, and %idle ratio.
  - Used here: \((100 - \%idle)\)

- **top:**
  - Default tool
  - Used for load and utilization monitoring
  - Command: `top -b -d T1 -n T2`
  - Load average for intervals of 1, 5, and 15 minutes
  - 1-minute-average considered
A Resource Monitoring Concept

1. Timescale requirements
   - Observation on short time-scales, i.e. seconds or minutes
   → Enables OPEX saving, e.g. by shutting down/starting of servers

2. Sampling concept and statistical characterizations
   a. Sampling
      
      ![Diagram](image)

      start load generation
      set-up phase Δs
      collect of data every Δt
      Collection phase
      stop load generation
      stop phase Δp
      t_{start} t_{start\_collection} Time [Second] t_{stop\_collection} t_{stop}

   b. Statistical values for sampling interval
      
      \[ E[X] = \frac{\sum_{i=1}^{N} x_{i,i}}{N} \]  \hspace{1cm} (1)
      \[ Min[X] = x_{\text{min}} \quad \text{with} \quad \forall x_i \in X : x_{\text{min}} \leq x_i \]  \hspace{1cm} (2)
      \[ Max[X] = x_{\text{max}} \quad \text{with} \quad \forall x_i \in X : x_{\text{max}} \geq x_i \]  \hspace{1cm} (3)
      \[ \sigma[X] = \sqrt{E[(X - E[X])^2]} \]  \hspace{1cm} (4)
      \[ cov[X] = \frac{\sigma[X]}{E[X]} \]  \hspace{1cm} (5)
Correlation Analysis

• Establishing Relationships between Guest and Host

- Hypervisor (e.g. KVM)
- Host Kernel
- Host OS
- CPU
- RAM
- I/O

• Scatterplots
  - Each point is a measurement/observation during a collection phase

(a) Weak Positive Correlation
(b) Strong Positive Correlation
Experiment Setup

- Test environment close to CityNetwork’s DC servers

- Measurements of tests in an experiment:
  - Hardware: E3-1230, 3.30GHz, 8 cores, 8 GB RAM
  - Disk: 500GB SAS
  - Host OS: Ubuntu Desktop 12.04 LTS, CentOS 6.6
  - Virtualization: kvm
  - Guest OS: Ubuntu 14.04 server
  - VM configuration: 2 vCPUs
  - Load monitoring: mpstat, top
  - Load generators: stress-ng
Load Generation

- **stress-ng tool**
  - Clean room implementation of original stress tool
  - loads (virtual)CPUs by calculating \( \sqrt{\text{(double)rand()}} \)
  - used command: `stress-ng -c N1 -l P -t N2`
  - Option \(-c\): number of processes that load the (virtual)CPUs
  - Option \(-l\): specifies the load imposed on CPUs by process; given in percentage (%)
Predictability of Scaling

- Weak but quite predictable but scaling
- Observed utilizations not deterministic and vary strongly
- Weakens predictability when load on guest is reduced.

Utilization: one loading process on guest; host: Ubuntu; tool: mpstat

Utilization: two loading processes on guest; host: Ubuntu; tool: mpstat
Predictability of Load

- Load: one loading process host: Ubuntu; tool: top
- Load: two loadings processes on guest; host: Ubuntu; tool: top

- Similar weak positive correlation as for utilization.
- Other effects that lead to variations and are obviously systematic → deeper collaboration with operating system developers required
Dependency on Host OS

- Utilization: two loading processes on guest; host: Ubuntu; tool: mpstat
- Utilisation: two loading processes on guest; host: CentOS; tool: mpstat

- CentOS has typically higher cov values at higher utilization values than Ubuntu
- However, CentOS more often considered as a “better” choice for data centers → Is this really true?
- Ubuntu seems to be a better choice when it comes to load optimization.
Differences between Tools

- Indication of a small advantage for top due to less visual variation
- Probably, we will always multiple tools → Makes orchestration probably more complex

Utilization: one/two loading process on guest; host: Ubuntu; tool: mpstat
Load: one/two loading process on guest; host: Ubuntu; tool: top
Dimensions of Resource Mgmt in Cloud Networking

- Vertical integration: layering concept (even recursive)
- Horizontal integration: end-to-end (multiple hops)
A Framework for Performance Data Integration – Overall Concept

Cloud Networking Control Application Layer
- CN (Umbrella) Controller
- Cloud Orchestrator (e.g. OpenStack)
- Network Controller (e.g. OpenFlow Controller)
- CN Profiler (provides vertical and horizontal integration)

Data Integration Layer (part of future Cloud Networking Operating System)
- Data Representation and Storage Service
  - (Establish global Cloud Networking view)
  - (coherent data structure and semantics)
- Data Translation Service
  - (Provides interoperability by translation)
- Data Collection Service
  - (Reliable gathering of data)

Cloud Networking Component Layer
- SDN-based OpenFlow switch
- IPv4/v6 router
- (Virtualized) Network Function Server
- Storage Server
- Cloud Computing Server
- Application Specific Virtual Machine
A Framework for Performance Data Integration – Data Integration Layer

Data Representation and Storage Service

Data Object A: QoE for VNF
Data Object B: Host Load (OS XYZ)
Data Object C: VM Load (Guest OS XYZ)

Data Translation Service

Translations Module A: QoE for VNF
Raw Data

Translations Module B: Host Load on Compute Server using OS XYZ
Raw Data

Translations Module C: Host Load on Compute Server using OS XYZ
Raw Data

Data Collection Service
Summary

- Performance management in virtualized systems is of increased complexity.
- Interpretation of simple observations/measurements is still complex or in the beginning → Reliable and und reasonable tools urgently needed to described load.
- Significant differences for similar guest load among different operating systems.
- Term “network load” in virtual infrastructures might comprise computational as well as network load in physical infrastructures.
  → Thus, what is “load” and how can we generate it?
  → “Integration” of load monitoring values needed, i.e. combination of values into easy description required.
- Orchestration architecture: horizontal integration not yet addressed.
- Data integration layer might enable comparability of load descriptions.
Tack så mycket!
Frågor?