CloudAP: Improving the QoS of Mobile Applications with Efficient VM Migration

**Renyu Yang**, Ph.D. Student
School of Computing, Beihang University
yangry@act.buaa.edu.cn
In IEEE HPCC, Zhangjiajie, China, 2013

Joint work with Yunkai Zhang, Tianyu Wo, Chunming Hu, Junbin Kang and Lei Cui
Mobile Computing and Cloud

• **Mobile device**
  - the amount of smartphone users worldwide exceeded 1 billion in the third quarter of 2012 and continues to grow sharply.
  - Shortcomings: low battery volume, poor resource configuration

• **Cloud Computing**
  - elastic resource management
  - application hosting
  - resource poverty compensation

• **“Mobile + Cloud”**
  - Virtualization techniques & Remote Execution
    - Execution separates from display and the results are transmitted to the thin device client
    - Cross-platform enhanced execution mechanism. Ex: make Windows application run on Android mobile phones
  - Challenges: **WAN(Wide Area Network) latency!**
Motivation

• How to mitigate the WAN latency?
  – trusted, resource rich computers or clusters near mobile users
    – *Cloudlet proposed by Mahadev*
  – Initialize custom VMs on Cloudlet → Software is run on cloudlets
  – Drawbacks:
    • the software can only be run on one single cloudlet
    • discard the previous execution environment before initializing a new one.

• Requirements:
  – Pervasive and seamless continuous software service provision.
    • Short service interruption time.
  – service’s continuous execution
    • Rapid and transparent execution environment switch.

• *CloudAP*
CloudAP

- *Inspired by “Access Point” in communication areas.*
- Small-scale cloud infrastructure close to the mobile devices and can be connected to Cloud datacenter

- **CloudAP + Cloud Center**

- **Typical Scale**
  - Cloud Center – CloudAPs – Servers – VMs
  - VM – User

- **Example**

<table>
<thead>
<tr>
<th>Options</th>
<th>CloudAP</th>
<th>Cloud Center</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Network</strong></td>
<td>LAN</td>
<td>WAN</td>
</tr>
<tr>
<td><strong>Sharing</strong></td>
<td>Few users</td>
<td>Large numbers of users</td>
</tr>
<tr>
<td><strong>Software Execution</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Latency</strong></td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td><strong>Bandwidth</strong></td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Scale</strong></td>
<td>Small</td>
<td>Large</td>
</tr>
</tbody>
</table>
Scenario

How to make these functionalities come true?
How to maintain the execution environment by using an efficient mechanism?
Proposed Two-tier Architecture

Cloud center
- Portal
- Cloud Agent
  - Registry
  - monitoring
  - Local scheduler

Resource pool

CloudAP
- VM Schedule Manager
- VM Migration Controller
- AP Agent

User
- VNC viewer
- VM
- VMM
- PM

Detect the geographical location
Parse messages and send operation instructions
Design of Whole-system VM live migration

• Objective: shorten the switch time
  – The duration from when the migration initiates to when the VM runs in the destination.
  – Prompt execution environment switch into a better QoS surroundings.

• Solutions
  – COW(copy-on-write) file system: base + cow image
    • Base image(large size) : read only and well-planned in advance.
    • Cow image: the added or modified.
  – Hybrid Copy Strategy
    • Disk : post copy but only for the cow image
    • Memory : pre & post hybrid copy
      – Considering the high frequency of access to memory pages, pure post-copy will lead to frequent memory page fault and performance degradation.
Design

- Only one iteration
- Dirty-bitmap is used to record the modified memory

- **Pull on demand**: dirty page pulling request occurs once a dirty page in the dirty-bitmap is accessed to.
- **Push actively**: the source pushes dirty memory pages to the destination according to the dirty-bitmap

Similarly, all the sectors in the disk image file should be set dirty in a block-bitmap
Optimization

• Most dirty sectors are pushed in advance rather than pulled when they are operated?
  – Analyze the disk I/O characteristics
  – Predict the I/O behaviors
  – Push the suitable sectors intelligently

• **Strong locality and continuity for read operations.**
  – If a read operation occurs to one dirty sector, the backward dirty sectors adjacent to this sector have a high probability to be accessed.

• **Data Compression Approach**
  – A long serial bit 0 or 1 in most dirty sectors.
  – RLE9(Run-Length Encoding) strategy.
Experimental Setup

Cloud center

16 dell servers with Intel(R) Core(TM) i7 860 2.80GHz CPU, 4GB memory and 320GB disk

QEMU-KVM kvm-84
512MB memory
10GB base disk image
250MB cow image

≤100Mbps

3 dell servers

CloudAP

simulated 3G network (average 5Mbps bandwidth and 35 ms latency)

WiFi(54Mbps)

CloudAP
Metrics

• **Downtime** *
  – the time interval during which software service is entirely unavailable

• **Switch time** *
  – the duration from when the migration is initiated to when the VM resumes and starts to run in the destination

• **Total migration time**
  – the duration from when the migration starts to when the VM runs independently in the destination

• **Amount of migrated data**
QoS improvement in two-tier architecture

We emulate the typical behavior of a user who tries to use Abiword, a popular text editing remote executed software on his own mobile phone.

<table>
<thead>
<tr>
<th>Operations</th>
<th>Cloud center(s)</th>
<th>CloudAP(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page scrolling</td>
<td>1.08</td>
<td>0.54</td>
</tr>
<tr>
<td>Text input</td>
<td>0.32</td>
<td>&lt;0.10</td>
</tr>
<tr>
<td>Text selection</td>
<td>0.51</td>
<td>&lt;0.10</td>
</tr>
<tr>
<td>Color change</td>
<td>0.53</td>
<td>&lt;0.10</td>
</tr>
<tr>
<td>Table insertion</td>
<td>0.52</td>
<td>&lt;0.10</td>
</tr>
<tr>
<td>Text dragging</td>
<td>1.04</td>
<td>0.34</td>
</tr>
<tr>
<td>Text centering</td>
<td>0.32</td>
<td>&lt;0.10</td>
</tr>
</tbody>
</table>

only takes less than 0.1s, which is acceptable for users
Efficient execution environment migration

Execution environment migration may cause a temporal downtime and a moment of turbulent service.

<table>
<thead>
<tr>
<th>Workload</th>
<th>Downtime(ms)</th>
<th>Switch time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abiword</td>
<td>9.5</td>
<td>24.16</td>
</tr>
<tr>
<td>Openoffice</td>
<td>10</td>
<td>25.93</td>
</tr>
<tr>
<td>Firefox</td>
<td>9.6</td>
<td>29.91</td>
</tr>
</tbody>
</table>

- The service downtime can be limited in 10 milliseconds, which can be nearly neglected.
- The total VM switch time is limited within 30s, which is acceptable for user to sacrifice this time for obtaining a better software service.
Migration performance comparison

Firstly realize the TPM[11](Three Phase Migration) and Deltas-apply[13] scheme in the hypervisor KVM

Reduced significantly to only 10 milliseconds in comparison to 2-3s

Switch time reduction
Migration performance comparison

**Migration time**

Although takes a slight longer time (13.6s) during disk post synchronization, the total migration time is greatly reduced by at most 58% in all circumstance.

**Migrated Data**

The entire migration data is only 481.66MB reaching roughly 55.4% and 56.5% decrease.
Conclusion

• Two-tier software execution architecture considering CloudAP and Cloud center (Public cloud).

• Optimized whole-system VM migration approach which fulfills rapid migration and effective execution environment switch.
Thank you for your attention!