Follow the sun through the Clouds

Live VM Migration for Geographically Shifting Workloads
Supercloud
Live VM (virtual machine) migration.

- The process of moving a running virtual machine between physical machines without disconnecting the client or application.
- Memory and storage are transferred from original guest to destination machine.

- **Pre-Copy memory migration:**
  - Copy all the memory pages from source to destination while the VM is still running on the source (dirty pages will be copied later).
  - Stop the VM in source and copy the remaining pages that have been changed during migration.
  - Resume the run on destination host.

- **Post-Copy memory migration:**
  - VM is suspended at the source, and only CPU state, registers and non-pageable memory copied to the destination.
  - The VM is resumed at the destination and the source actively pushes memory pages to the VM.
  - If the destination gets page-fault, it redirects to the source for the memory.
Supercloud

- A **uniform cloud service** that supports live VM migration between data centers of all major public cloud providers.

- Provides a **scheduler** that automatically determines when and where to move VMs for optimal performance.

- **Shifting workloads** effectively, with low downtimes and transparently to both services and the clients.

- Supporting **cross-cloud** storage and networking.
Application migration?

● Distributed consensus or transaction protocol in order to support data replication, distributed locking and distributed transactions.

● Designed to be fault-tolerant while adding and removing nodes, good performance during reconfiguration is non-trivial.

● Examples:
  ○ MongoDB does not allow changing the sharing key on the fly, whole database must be exported and then imported again to change key distribution.
  ○ Re-configuring a cluster running an old version of ZooKeeper requires a procedure of shutdown and restart.

● Applications must be migrated several times a day by adding and removing nodes, that is suboptimal.
Applications make migration decisions by themselves through the Supercloud scheduling framework.

With N different data centers \( d_1, d_2, ..., d_N \) using Supercloud, and an application with \( k \) nodes \( P = \{p_1, p_2, ..., p_k\} \) (\( p_i \) is the \( i \)th node location).

Latency matrix \( L \), where \( L(i,j) \) is the round trip time from \( d_i \) to \( d_j \).

\( S \) as statistic workload report for this unique application (read, write, location), will be explained for each application.

The algorithm uses three parameters, evaluation function \( f(P,S,L) \) and a threshold \( T \) (minimal score that triggers VM migration).

Iterates over all placements in \( P \) and returns a set of placement plans \( D \), that maximize the score:

\[
D = \text{argmax} \{ f(P,S,L) \mid f(P,S,L) \geq f(P_{current},S,L) + T \}
\]

Selects the one that requires the fewest migrations from \( P_{current} \).
Scheduling Framework

- **Non-Distributed applications:**
  - Deploy a set of front-ends located in different data centers to collect user requests and forward them to the VM application.
  - The placement plan is the location $P = \{p\}$, where we have the running VM.
  - The goal is to minimize average latency for all front-ends, $S = \{(s_1, d_{s_1}), (s_2, d_{s_2}), \ldots, (s_n, d_{s_n})\}$. Where $n$ is the number of front-ends, $s_i$ is the number of active clients of the $i$th front-end and $d_{s_i}$ is its location.
  - We get an evaluation function $f(P, S, L) = -\sum s_i \cdot L(d_{s_i}, p)$, lower latencies result in a higher score.
Scheduling Framework

- **Distributed applications with Replicated State:**
  - One server acts as a leader that communicates with a majority of servers to agree on updates.
  - Read requests are handled by any node and write requests must be broadcast by the leader and agreed upon by the majority.
  - For an ensemble with m nodes, a placement plan is the location of all nodes $P = \{p_1, p_2, ..., p_m\}$ where $p_i$ is the leader location.
  - We consider read and write request to properly evaluate placement plan $S = \{(r_1,w_1,d_{s1}),(r_2,w_2,d_{s2}),..., (r_n,w_n,d_{sn})\}$ where $r_i$ and $w_i$ are the number of read and write requests and $d_{si}$ is its location.
  - For the $i$th front-end, the expected read latency is $R_i = \text{avg} L(d_{si},p_j) \forall j=1...m$. 
Scheduling Framework

- **Distributed applications with Replicated State:**
  - Write requests are processed in three steps:
    - Write request from front-end goes randomly to one of the nodes, average latency is $w_i(1) = \text{avg } L(d_{si},p_{j}) j=1...m$.
    - The request forwarded to the leader, the average latency is: $w_i(2) = \text{avg } L(p_{j},p_{l}) j=1...m$.
    - The leader broadcasts the request twice, for each it must wait for at least half of the ensemble to replies. The average latency is: $w_i(3) = 2x \text{ median } L(p_{l},p_{j}) j=1...m, j\neq l$.
    - The expected reply network latency for write request is $w_i = w_i(1) + w_i(2) + w_i(3)$.
  - The evaluation function is calculating the weighted average for all requests, assuming a weights $\alpha$ and $\beta$ for read and write: $f(P, S, L) = - \sum (\alpha \cdot R_i \cdot r_i + \beta \cdot W_i \cdot w_i)$.

- Note: this current protocol for read and write is as an example for the ZooKeeper distributed application.
Supercloud Architecture

- Two layers of hardware virtualization:
  - First layer is IaaS that managed by the cloud provider.
  - Second layer is also IaaS that managed by the Supercloud and can be completely controlled by the users.
Supercloud Architecture

- The second layer:
  - Uses Xen-Blanket para-virtualized interface.
  - One VM is called Dom0 and it manages the other VMs, called DomUs.
  - Uses same storage to serve VM images, VM migration needs to transfer only the memory.
  - Communicate in a consistent manner no matter where the end-point VMs reside.
  - A migrated user VM IP address remains unchanged, using SDN (software defined network).

**Xen-Blanket** - service that homogenizes diverse cloud infrastructures, users are able to run their unmodified VMs on any cloud without any special provider support.

**Para-virtualization** - the guest system is aware that it is a guest and has drivers issuing commands to the host and not to the hardware.
Supercloud Architecture - Storage

A running application typically does not require all data in the image. We only need to get the used blocks that have been changed for the current migrated VM and its application.

- **Data view layer:**
  - The global meta-data includes a version number for each block.
  - When VM is reading a block, the version number is compared to the version number in the local meta-data in the local data store.
  - If the latest version is available locally then it is returned, else checks the location and fetches remotely. Increases its version number by one.

- **Data store layer:**
  - Stores and propagates data, images are divided into blocks with a constant 4KB.
Data Propagation Policies:

- Fetching each block on-demand significantly degrades read performance.
- A block that is read frequently and updated rarely should be aggressively propagated, and propagating a block that is updated frequently is a waste of network resources.
- The data store layer monitors the access pattern of the VMs and selectively propagates those blocks that are most likely to be accessed after migration and least likely to be updated after propagation.
Supercloud Architecture - Storage

Data Propagation Policies - Algorithm:

- Maintain a priority queue for updated blocks for each image file, the priority is updated on each read or write (use the read/write ratio $F = \frac{f_r}{f_w}$ to calculate).
- When two blocks have the same value of $F$, we first propagate the block with a higher read frequency.
- The propagation priority $P_b$ for a block $b$ is:

$$P_b = \begin{cases} 
K \cdot \frac{f_r^b}{f_w^b} + f_r^b & \text{if } \frac{f_r^b}{f_w^b} \geq S \\
-1 & \text{if } f_w^b = 0 \text{ or } \frac{f_r^b}{f_w^b} < S
\end{cases}$$

- The $K$ determines how much we want to favor the read/write ratio, we use $k=1000$. 
Supercloud Architecture - Storage

**Data Propagation Policies - algorithm:**

- The constant $S$ determines the aggressiveness of the propagation, the lower $S$ means more data will be propagated.
- When $f_w^b = 0$, which means the block has never been updated, or when $\frac{f_r^b}{f_w^b} < S$, we set the priority to -1 to indicate that this block is not to be propagated proactively.
- At last we need to determine where to propagate the block, a transition probability table is added into the image file’s meta-data, indicating the probability that a VM is moved from one place to another (the table holds the number and place for all previous blocks migrations).
Supercloud Architecture - Networking

VPN (virtual private network):

- VPN is the extension of a private network that includes links across shared or public networks.
- VPN connections enable to send data between two computers across the Internet in a manner that emulate the properties of a point-to-point private link.
- VPN tunneling enables the encapsulation of one type of protocol packet within the datagram of a different protocol. (PPTP - Point-to-Point Tunneling Protocol to TCP/IP).
Supercloud Architecture - Networking

High-Performance VPN (virtual private network)

- Existing VPN solutions use a centralized server to forward traffic, which causes high latency and poor throughput.
- Uses Open vSwitch, VXLAN tunnels and Frenetic SDN controller to build a high performance VPN.
- Packets always forwarded directly to their destinations.

- **Open vSwitch** - distributed virtual multilayer network switch, provides switching stack for hardware virtualization.
- **VXLAN tunnels** - virtual extensible LAN is a virtual network that is build on top of existing network layer 2 and layer 3.
- **Frenetic SDN controller** - domain specific language for programming software-defined networks, allows network operators to program the network as a whole.
Supercloud Architecture - Networking

High-Performance VPN (virtual private network)
We investigate three questions to evaluate Supercloud performance:

1. How effective is the scheduler in enabling applications to follow-the-sun?
2. Is VM migration a viable approach to follow-the-sun?
3. What is the efficacy of Supercloud storage and networking in supporting live VM migration?

- A distributed application, ZooKeeper is used to investigate application performance.
- ZooKeeper - hierarchical key-value store, used to provide a distributed configuration service, synchronization service for large distributed systems.
Supercloud - Evaluation

**Experiment setup - Follow the Sun:**

- Measure ZooKeeper ability to respond to clients in two different regions, Virginia and Taiwan.
- ZooKeeper clients randomly connected to one ZooKeeper server and submitted blocking read and write requests in a ratio of 9:1.
- The clients ran of first-layer VMs in Virginia on Amazon VMs and in Taiwan on Google VMs.
- Three ZooKeeper nodes (zk1, zk2, zk3) were allocated in each ensemble.
- Zookeeper server VMs reported VM load to the scheduler, the scheduler placement evaluation was triggered every minute.
- Migrations were performed in parallel.
- Tested 3 different set-ups:
  - **US Ensemble**: all three ZooKeeper nodes, zk1, zk2 and zk3, were in Amazon Virginia.
  - **Global Ensemble**: zk1 and zk2 were in Amazon Virginia, and zk3 was in Google Taiwan.
  - **Dynamic Ensemble**: the Supercloud scheduler automatically placed VMs according to the workload.
Supercloud - Evaluation

Experiment results:

Figure 4. ZooKeeper throughput (vertical dashed lines indicate the end of the migrations).

Figure 5. ZooKeeper latency CDF.
Supercloud - Evaluation

Experiment results:

- The figure shows the throughput for the three scenarios.
- "Observed US" - is the throughput measured for US clients.
- "Observed Asia" - is the throughput measured for Asian clients.
- "Expected Aggregate" - is the throughput if the latency were negligible.

- Figure 4.a - US clients were able to reach maximum throughput, but Asian clients suffered from high latency and thus low throughput.
- Figure 4.b - US clients still had good throughput because two of the servers were in US, but Asian clients still suffered from high latency.
- Figure 4.c - US clients and Asian clients reached maximum throughput when the VM could migrate for US to Asia and vice versa.

- Figure 5.a - 90% of US clients observed less than 15ms latency, were all Asian clients observed close to 200ms latency.
- Figure 5.b - Still, most of the US clients observed low latency, were half of Asian clients observed close to 200ms latency.
- Figure 5.c - 70% of all clients observed less than 15ms latency, and only 20%-30% observed 150-200ms.
Experiment setup - Comparing migration approaches (Application VS VM):

- We used Cassandra and ZooKeeper, to investigate the viability of relying on live VM migration via the Supercloud versus regular application migration.

- Application migration (Cassandra):
  - Started a 3-node Cassandra cluster in the Amazon Virginia region, then migrated the whole cluster to the Google Taiwan region (followed the process specified in the Cassandra documentation).
  - Allocated three new nodes in Google Taiwan region and the data was copied from Amazon Virginia 3 nodes to Google Taiwan 3 nodes.

- Supercloud migration (Cassandra):
  - Setup up VMs in Amazon Virginia and Google Taiwan as before.
  - The database was populated with 30,000 key/value pairs, each had a size of 1KB.
  - One client was started in each region, the clients read and wrote to the database continuously with a ration of 4:1.
Supercloud - Evaluation

Experiment results:

- The top results are for application migration for both Amazon Virginia and Google Taiwan clients.
- During migration, the throughput dropped dramatically and remained low for about 200 seconds (as seen in first graph).
Supercloud - Evaluation

Experiment setup - Comparing migration approaches (Application VS VM):

- **Application migration (ZooKeeper):**
  - “2-step reconfiguration” - moved a majority of the servers from one region to another: first added new node in Tokyo, then removed the original leader in Virginia.
  - “3-step reconfiguration” - added two nodes in Tokyo first, then removed both nodes from Virginia. After the new leader was elected in Tokyo, added a new node in Virginia and removed one of the nodes from Tokyo.

- **Supercloud migration (ZooKeeper):**
  - Setup up VMs in Amazon Virginia and Google Taiwan as before.

- One client was started in each region, generating a constant workload with read/write ratio 9:1.

- Each read operation obtained a 1 KB ZooKeeper znode and each write operation overwrote a 1 KB znode.
**Supercloud - Evaluation**

**Experiment results:**

- (a) In “2-step reconfiguration” the leader was in Virginia and two followers were in Tokyo, which was inefficient and, as a result, the throughput dropped significantly for both clients.
- (b) In “3-step reconfiguration” the leader was successfully moved to Tokyo so good throughput were achieved. It took 20 during which performance was inconsistent and low.
- (c) For the super cloud, the drop in performance was less than a second and transparent to the ZooKeeper application and clients.
Supercloud - Evaluation

Experiment setup - Storage Evaluation:

- The purpose was to evaluate benchmark performance during live migration.
- Deployed two Supercloud setups instances in Amazon Virginia and Northern California regions.
- Ping latency between two VMs was 75ms.
- We started a user VM with 1 virtual CPU core and 512MB memory and ran a benchmark in it.
- Without migration, a VM using local storage finished the benchmark in two minutes.
- In the experiments, immediately after starting the benchmark, we triggered a VM migration from Virginia to Northern California.
- The full migration took 40 to 50 seconds.
- NFS: a traditional NFS server deployed in Amazon Virginia.
- Sync-on-Write: consistent geo-replicated store that synchronously propagates each write.
- On-Demand: Supercloud storage with proactive propagation turned off.
- Supercloud: Supercloud storage with proactive propagation in the background.
Supercloud - Evaluation

Experiment results:
**Experiment results:**

- (a) NFS shows that throughput dropped significantly after the clients were migrated.
- (b) Sync-on-write shows that it incurs high performance overhead both before and after migration because each write needs to be propagated through the internet.
- (c) Supercloud on demand, eventually achieved good average throughput after migration.
- (d) Supercloud proactive, shows that most read and write requests could be served locally.

- Figure 9.a shows the average throughput for each case, Supercloud proactive achieved the highest throughput.
- Figure 9.b shows WAN traffic, On-demand achieved the lowest WAN since it fetched data remotely only when necessary. The Supercloud incurred more incoming traffic because it pushed some data that was not needed.
Supercloud - Evaluation

Experiment setup - Network Evaluation:

- Deployed the Supercloud in two instances in Amazon Virginia cloud.
- The purpose was to measure UDP and TCP throughput between them.
- Measured UDP and TCP throughput, between second-layer Dom0 VMs and DomU VMs (the Dom0 is the hypervisor for DomU VMs).
- For the latency test, we ran 50 UDP pings.
- **Non-nested**: a setup where we ran the benchmark directly in the first-layer VMs.
- **OpenVPN**: a VPN solution using a centralized controller.
- **Tinc**: a P2P VPN solution, implements full-mesh routing.
- **Supercloud**: Supercloud implementation based on Open vSwitch.

UDP - user datagram (basic transfer unit) protocol for data transfer.
Experiment results:

- Although Open vSwitch based virtual network slightly increases the UDP latency, the overhead was much smaller than either openVPN or tinc.
- Dom0 latency was smaller than DomU because network packets to DomU go through Dom0 first.
● **The Supercloud** presents a complete cloud software stack under the user’s full control that can seamlessly span multiple zones cloud providers.

● It features live migration, as well as shared storage, virtual networking, and automated scheduling of workloads, placing and migrating VM resources as needed.

● [Supercloud demo](#)
Thanks!