SUPERCLOUD: GOING BEYOND FEDERATED CLOUDS
Should we migrate critical data to computation
or vice versa?

E.g. app needs to import or export data
Should we migrate critical data to computation

... or vice versa?

- E.g. app needs to import or export data

Challenges

- Limited bandwidth tactical networks
- Interoperability
- Secure sharing

How can we securely and efficiently migration computation across the cloud?
INFRASTRUCTURE AS A SERVICE

- Offer on-demand virtual machines
- Charge according to used hours
- Multiple data center locations
PROBLEM – SINGLE CLOUD

- Vendor lock-in
  - Latency limitation
  - No control of price
  - Availability limitation

Which one? Price? Migration? Availability?

- Amazon Web Services
- Rackspace
- HP Cloud
- Google Compute Engine
- Azure
BENEFITS OF MULTIPLE CLOUDS

Provide power, control and flexibility to user

- Lower latency
- Reduce Cost
- Higher availability
- Burst Relief
- Security Improvement
The Supercloud goes beyond a Federated Cloud in that it

- supports *user-level* migration between autonomous clouds;
- Permits environment to include “stubs” for resources that actually are tied to specific places, making them seem to be available from anywhere;
- supports a variety of underlying virtual machine monitors;
- supports a shared but decentralized storage system and a novel virtualized network that can migrate with the app;
- supports system-wide pub/sub for event notification.
Unshackle the Cloud: xClouds

- Bring **extensibility** into IaaS clouds

- Allow users to run or implement their own hypervisor-level services

- Avoid lock-in with **user-centric homogenization**
How to Build xClouds

- Users are isolated
- VMM composed of **modules**
How to Build xClouds

- Users are isolated
- VMM composed of modules
  - User / Provider
    (U / P)

```
    Hardware
     /           /
 User 2  User 1  User 3
     |           |
     |           |
 VM   VM
 U    P
 P    P
 P    U
 P    U
 P    P
 P    P
```
How to Build xClouds

- Users are isolated
- VMM composed of **modules**
  - User / Provider
    - (U / P)
  - Mutable / Immutable
    - (P / U)

![Diagram showing User 1, User 2, and User 3 with VMs and Hardware]
How to Build xClouds: Alternatives

Download VMM Extensions

e.g SPIN, VINO

Providers must adopt new VMM

Hardware
# How to Build xClouds: Alternatives

<table>
<thead>
<tr>
<th>Download VMM Extensions</th>
<th>Expose Hardware Through VMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>e.g. SPIN, VINO</td>
<td>e.g. Exokernel</td>
</tr>
<tr>
<td>Providers must adopt new VMM</td>
<td>Providers must adopt new VMM</td>
</tr>
</tbody>
</table>

![Diagram showing VMM extensions and hardware exposure]

**Hardware**
## How to Build xClouds: Alternatives

<table>
<thead>
<tr>
<th>Download VMM Extensions</th>
<th>Expose Hardware Through VMM</th>
<th>Add Another VMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>e.g. SPIN, VINO</td>
<td>e.g. Exokernel</td>
<td>e.g. Turtles Project</td>
</tr>
<tr>
<td>Providers must adopt new VMM</td>
<td>Providers must adopt new VMM</td>
<td>Turtles needs VMM support, but...</td>
</tr>
</tbody>
</table>

![Diagram showing the process of building xClouds with different VMM extensions and exposure methods.](image-url)
HOW TO BUILD XCLOUDS: ANOTHER LAYER
HOW TO BUILD XCLOUDS: ANOTHER LAYER

Linux Dom0

PCI-Driver

Backend

Dom0 for Xenblanket

Frontend

PV-DomU

PV-DomU

Xen - 4.1.1 Blanket

1st-Layer Xen

Hypercalls

Device

Baremetal
xClouds works Today!

- Nested paravirtual device drivers
- Xen on EC2
xCLOUDS WORKS TODAY!

- Nested paravirtual device drivers
- Xen on EC2

- Can create your own

Cloud-within-a-Cloud
- Event-Channel Drivers for virtual interrupts
  - One for Receive Event from 1st-Layer Xen: virtual interrupts
  - One for multiplexing to nested DomU

- Grant Page Table Drivers for shared memory
  - One for Shared Ring buffer with 1st-Layer Dom0
  - One for sharing memory with/or between nested DomUs

- Two Xenbus drivers
  - One for connect to 1st-Layer shared memory devices
  - One for nested DomUs to connect to nested shared memory devices

- Block and Net Frontend Drivers
  - For virtual disk and network devices of nested Dom0
Need Hypercall Passthrough

- Nested Dom0 must be able to get information about shared memory devices from 1st Layer-Xen

- Nested Dom0 can only issue hypercall to Nested Xen
  - So, nested Xen should help passthrough related hypercalls
KILLER APPS FOR THE SUPERCLOUD

- Sensitive Resource
- Follow the sun
- Geographic proximity
- Spot Instances
- Dynamic Resource Scaling
- Bursting
- Sensitive resource
  - Some provider has a special device, or special data
  - Supercloud allows you to add a “stub” for it to your environment. Looks like a normal local device or local data file / database

- If you access the resource, SuperCloud migrates your VM to where it resides
  - Moving computation to data on demand avoids locking app to the place where that resource resides
Uniform VM image?
- using nested virtualization

Cross-cloud migration?
- developed a new image storage facility

Transparent networking?
- based on SDN: Open vSwitch and VXLAN
SUPERCLOUD ARCHITECTURE

Amazon

- Dom0
- Open Stack
- User VMs
- XenServer
- Xen-Blanket
- Xen/PV-on-HVM

Supercloud Architecture Diagram

HP Cloud

- Dom0
- Open Stack
- User VMs
- XenServer
- Xen-Blanket
- KVM/virtio

Rackspace

SDN

Xen-Blanket

User VMs

XenServer

Dom0

Open Stack

Xen/PV-on-HVM

BENEFITS OF MULTIPLE CLOUDS

Provide power, control and flexibility to user

Multi-cloud

- Lower latency
- Reduce Cost
- Higher availability
- Burst Relief
- Security Improvement
Which data center is closer?
CONTENT DELIVERY NETWORK

![Graph showing minimum latency (ms) for RackSpace, Amazon, HP Cloud, and SuperCloud.]
SuperCloud is closer to the clients

Supercloud and Rackspakce has the lowest Latency

Supercloud and Amazon has the lowest Latency

CONTENT DELIVERY NETWORK
BENEFITS OF MULTIPLE CLOUDS

Provide power, control and flexibility to user

- Lower latency
- Reduce Cost
- Higher availability
- Burst Relief
- Security Improvement

Multi-cloud
Amazon Spot Instance
- Dramatic Price Change
- Usually Cheap
- Charges hourly
- Ready to be terminated

Can’t save money when prices goes high
Only stateless jobs!
SMART SPOT INSTANCE

SuperCloud on Spot Instances

- Migrate out to other places when instances being terminated
- Migrate to the cheapest place when starting new billing hours

Benefits

- No termination
- Lower budget
SPOT INSTANCE PRICE HISTORY

Spot price goes very high sometimes.

Regular instances have stable price.

Spot price is usually low.

Supercloud is migrated to the cheapest place every hour.

Migrate supercloud to regular instances.
Smart spot instances save money!
SuperCloud uses Xen on Xen virtualization

- Has some extra overheads, obviously
- But enables migration of the entire VM and its runtime environment

SuperCloud is able to...

- Moves computation to data if data is big, sensitive, special device on a special host, etc
- Can also migrate to chase lowest-priced resources

Transparent to the application: standard OS APIs