ERICSSON

Cloud Native
#1 - This thing called cloud

Johan Eker

Lars Larsson

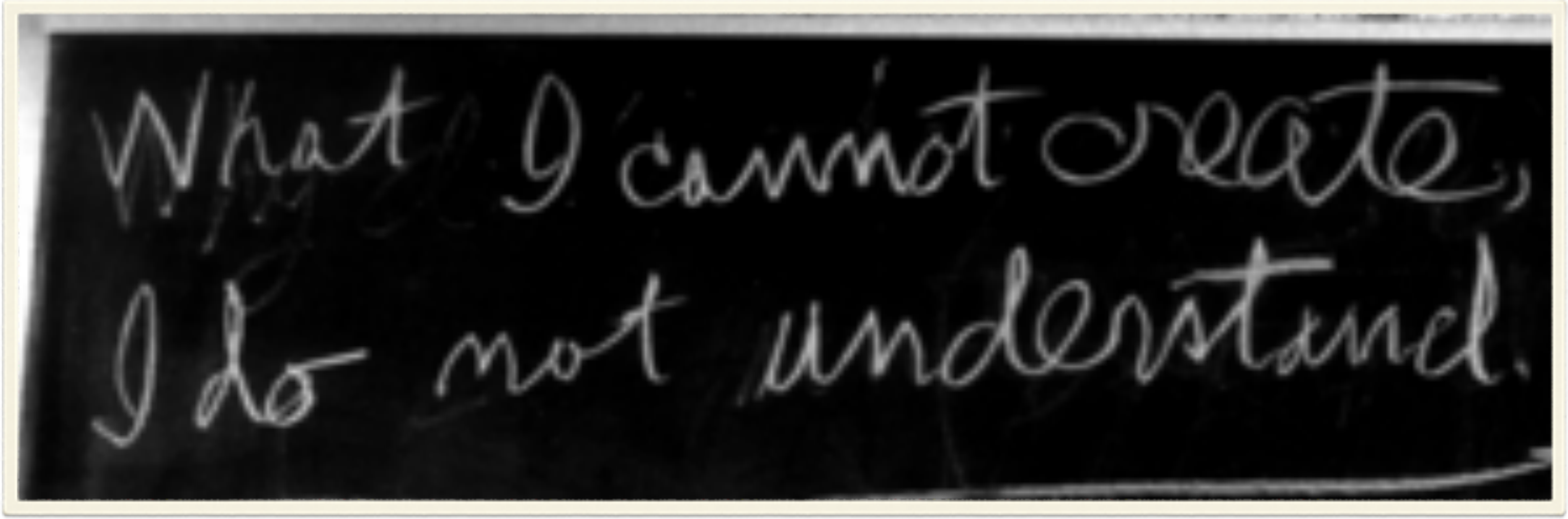
Master of all things cloud.

This Course

Learning goals

- Good understanding of the principles behind cloud services, e.g. virtual resource, storage, etc.
- Ability to manage infrastructure-as-a-service (IaaS) and design and implement robust and scalable cloud applications.
- Good understanding of the underlying theoretical challenges with distributed systems in a cloud context, i.e. consensus, consistency, time, etc.
- Ability to design, implement and deploy data and compute intense cloud native applications on standard cloud platforms.
- Good overview of technology trends and research topics.

No exam. Hand-in all assignment by the time of the final presentation (session #8). No later.


What I cannot create,
I do not understand.

Facilities

- Hands-on will mean a lot of work (and sometimes need for support) -- Help each other!
- Slack will be our means of communication between sessions
  - https://cloudnativecourse.slack.com

- You will get accounts on Ericsson Research Data Center (use them with care)
  - https://xerces.ericsson.net

- Store your code at the course's GitLab service
  - https://gitlab.datahub.erdc.ericsson.net

Prerequisite

- Python know-how
- Basic operating system skills
- Access to a computer where are root (or where you can install Docker)
- A lot of time...

patience

/ˈpeɪʃ(ə)ns/

noun

1. the capacity to accept or tolerate delay, problems, or suffering without becoming annoyed or anxious.
   "you can find bargains if you have the patience to sift through the rubbish"
   synonyms: forbearance, tolerance, restraint, self-restraint, resignation, stoicism, fortitude, sufferance, endurance; More
2. BRITISH
   any of various forms of card game for one player, the object of which is to use up all one's cards by forming particular arrangements and sequences.

TL;DR

Prepare for everything to fail sometime (a promise)

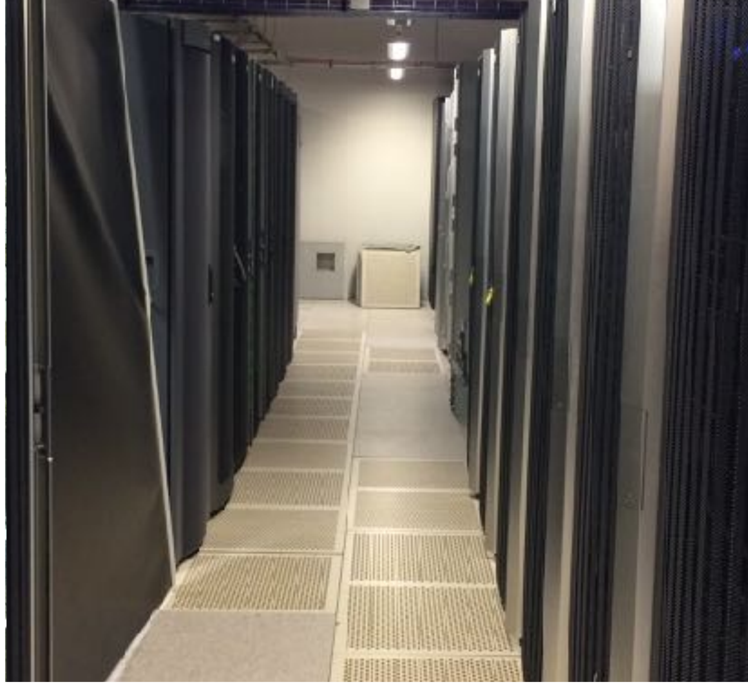
Avoid state whenever possible (or be quick to save)

Allow for inconsistency (you have no choice)

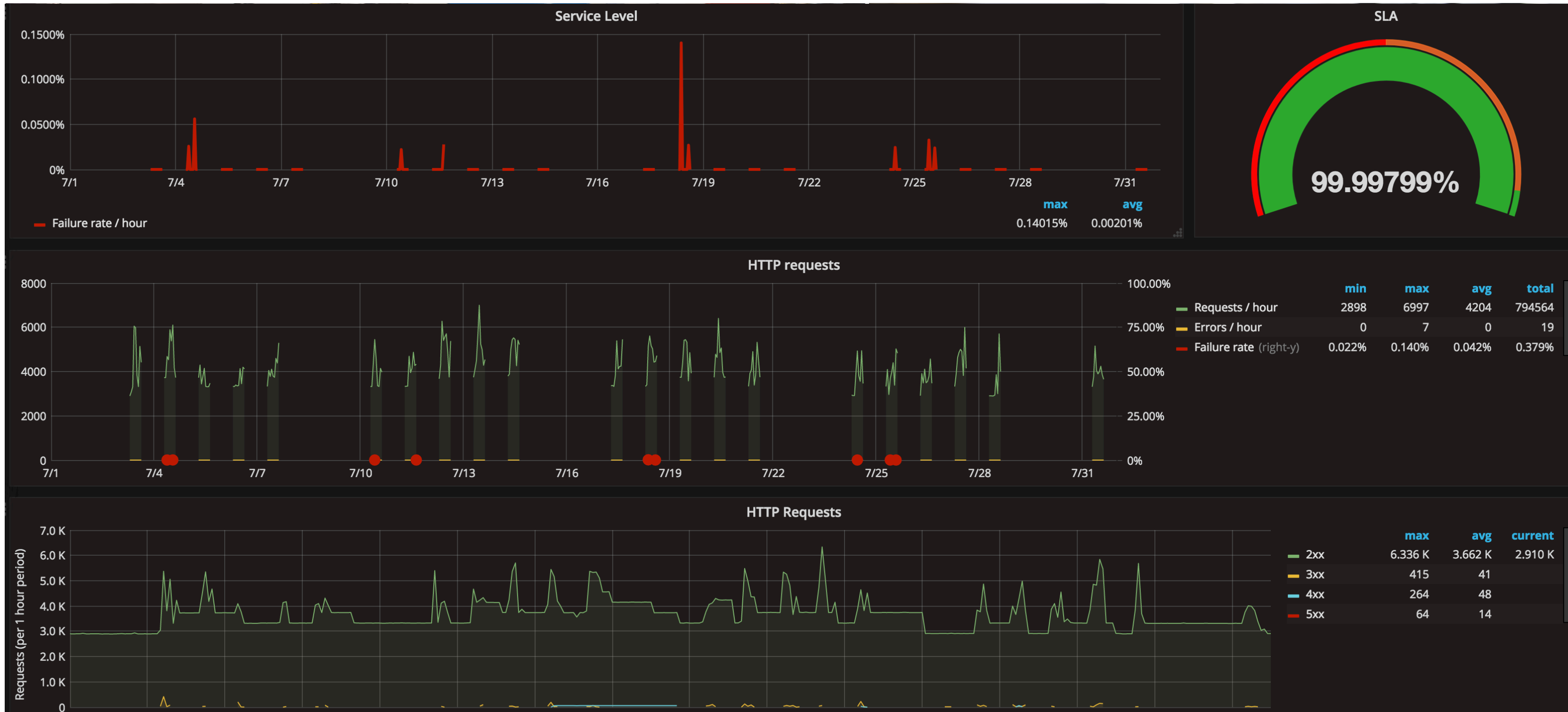
Keep things simple (things will get complex anyway)

Pick your tools well (and stick to them)

Cache is king



ER
DC


Service Level
0.1500%
0.1000%
0.0500%
0%
7/1 7/4 7/7 7/10 7/13 7/16 7/19 7/22 7/25 7/28 7/31
Failure rate / hour
max 0.14015%
avg 0.00201%
SLA
99.99799%
HTTP requests
8000 6000 4000 2000 0
100.00% 75.00% 50.00% 25.00% 0%
min max avg total
Requests / hour 2898 6997 4204 794564
Errors / hour 0 7 0 19
Failure rate (right-y) 0.022% 0.140% 0.042% 0.379%
HTTP Requests
Requests (per 1 hour period)
7.0 K 6.0 K 5.0 K 4.0 K 3.0 K 2.0 K 1.0 K 0
max avg current
2xx 6.336 K 3.662 K 2.910 K
3xx 415 41
4xx 264 48
5xx 64 14

ER DC Team

Ericsson and SEB make banking personal again
By: Ericsson | 24 February, 2017 | Business | banking, SEB
"I opened my company account via video call. Fast and convenient."
seb.ee/eng/remote-advisory
SEB
When was the last time you visited your bank? The advent of internet banking – now accessible via phones as well as PCs – has made the idea of actually traveling to a physical bank branch seem antiquated.
And yet there are still times when, despite the amazing convenience of internet banking, there is no substitute for being able to talk to another human being. So you find some time in your busy schedule, make an appointment and fight the traffic on the way to the bank.
If only there were a better way.
5GEM
SKF
VINNOVA
SIGMA
Autonomous Vessels
AXIS COMMUNICATIONS
SAAB
WASP WALLENBERG AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM
Connected Transport
KING'S LAUNDRY
WASP WALLENBERG AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM
ASTAZERO
MicroWeather
Example data
SMHI
Connected Drone
Telia
Vejdirektoratet
NordicWay
NordicWay is a pilot project that seeks to enable vehicles to communicate safety hazards through cellular networks on a road corridor through Finland, Norway, Sweden and Denmark.
The project is a collaboration between public and private partners in the four countries, and is co-financed by the European Union within the Connecting Europe Facility programme 2015-2017.
Nordic WAY
Cellular C-ITS
3G and 4G/LTE
interoperable
ecosystem
2000
Smarta Offentliga Miljöer II
Schneider Electric
SENSATIVE
VINNOVA

Chris Nig
@Chris_Nig

Follow

hey @EricssonLabs , your cocoapods server pods.cct.ericsson.net seems to be down

12:59 PM - 8 Apr 2017

1

Ericsson Research @EricssonLabs · Apr 10
Replying to @Chris_Nig
Up and running again. Sorry for the inconvenience :(

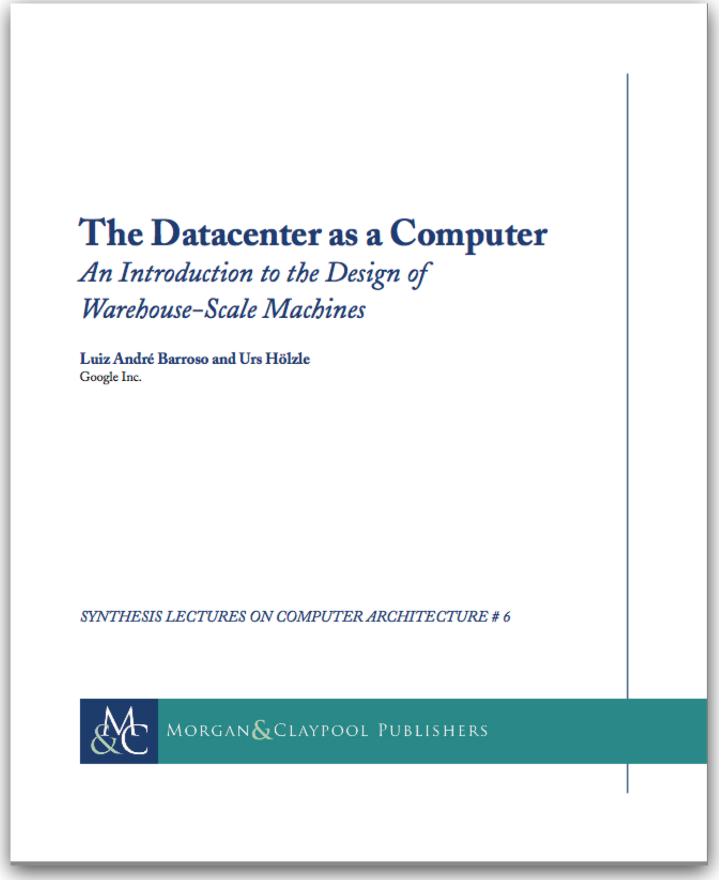
From: [redacted]
Sent: den 10 april 2017 13:52
To: [redacted]
Subject: RE: Är det nån av er som vet nåt om det här?

The Research Cloud in Lund was down during the weekend due to a rat eating our ISP’s fibre cables.

Regards,
Magnus Wester

This session

The not so cloud native way of doing things



The Datacenter as a Computer

An Introduction to the Design of Warehouse-Scale Machines

Luiz André Barroso and Urs Hölzle
Google Inc.

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE # 6

MORGAN&CLAYPOOL PUBLISHERS

practice

DOI:10.1145/1721654.1721672

Clearing the clouds away from the true potential and obstacles posed by this computing capability.

BY MICHAEL ARMBRUST, ARMANDO FOX, REAN GRIFFITH, ANTHONY D. JOSEPH, RANDY KATZ, ANDY KONWINSKI, GUNHO LEE, DAVID PATTERSON, ARIEL RABKIN, ION STOICA, AND MATEI ZAHARIA

A View of Cloud Computing

CLOUD COMPUTING, THE long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about overprovisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or underprovisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get results as quickly as their programs can scale, since using 1,000 servers for one hour costs no more than using one server for 1,000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT.

As a result, cloud computing is a popular topic for blogging and white papers and has been featured in the title of workshops, conferences, and even magazines. Nevertheless, confusion remains about exactly what it is and when it's useful, causing Oracle's CEO Larry Ellison to vent his frustration: "The interesting thing about cloud computing is that we've redefined cloud computing to include everything that we already do.... I don't understand what we would do differently in the light of cloud computing other than change the wording of some of our ads."

Our goal in this article is to reduce that confusion by clarifying terms, providing simple figures to quantify comparisons between of cloud and conventional computing, and identifying the top technical and non-technical obstacles and opportunities of cloud computing. (Armbrust et al[4] is a more detailed version of this article.)

Defining Cloud Computing

Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. The services themselves have long been referred to as Software as a Service (SaaS).[a] Some vendors use terms such as IaaS (Infrastructure as a Service) and PaaS (Platform as a Service) to describe their products, but we eschew these because accepted definitions for them still vary widely. The line between "low-level" infrastructure and a higher-level "platform" is not crisp. We believe the two are more alike than different, and we consider them together. Similarly, the

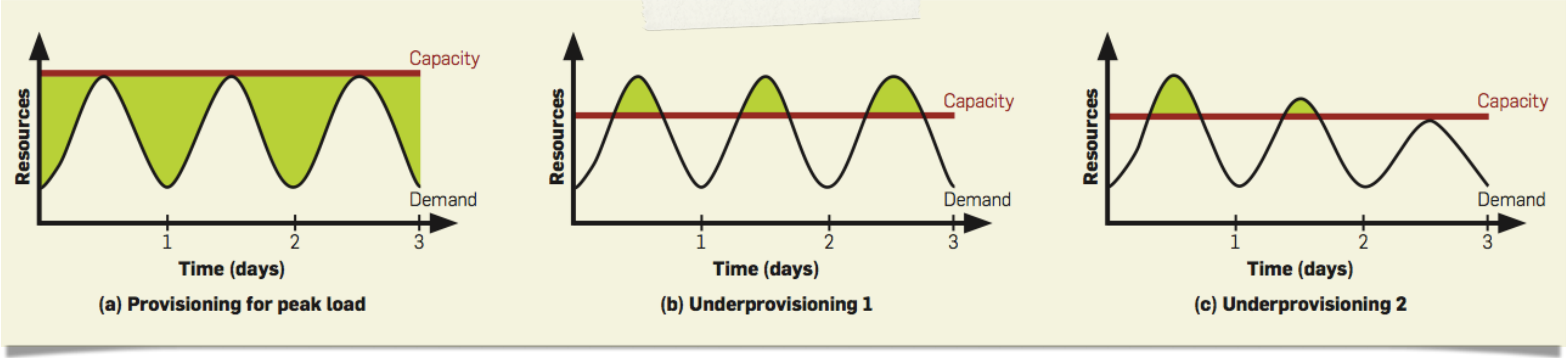
a For the purposes of this article, we use the term Software as a Service to mean applications delivered over the Internet. The broadest definition would encompass any on demand software, including those that run software locally but control use via remote software licensing.

50 COMMUNICATIONS OF THE ACM | APRIL 2010 | VOL. 53 | NO. 4

Cloud is a business model

Buy compute power by the meter

Allocate just the right amount

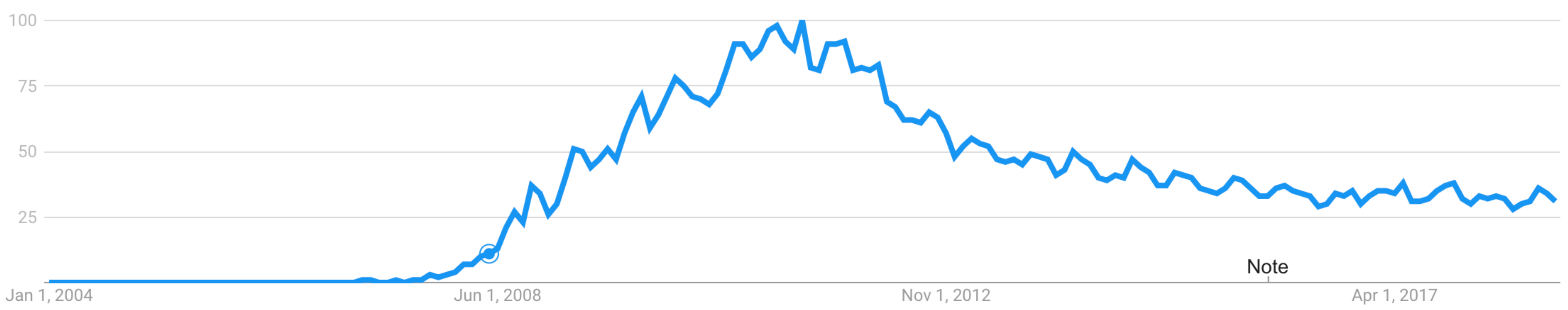


cloud computing
Search term

+ Compare

Worldwide · 2004 - present · All categories · Web Search

Interest over time



The next utility

> If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry.
>
> —John McCarthy, speaking at the MIT Centennial in 1961[2]

The illusion of infinite compute power and storage at your fingertips

Let’s try to define cloud computing

NIST

- On-demand self-service.
- Broad network access.
- Resource pooling.
- Rapid elasticity.
- Measured service

Quiz

(Excerpt from Intel Developer Forum Keynote 2000)

ANDREW GROVE: is there a role for more powerful computers?
GUEST: **More memory especially is very useful for us**,
you can start to think about having, like, the whole Web in RAM.

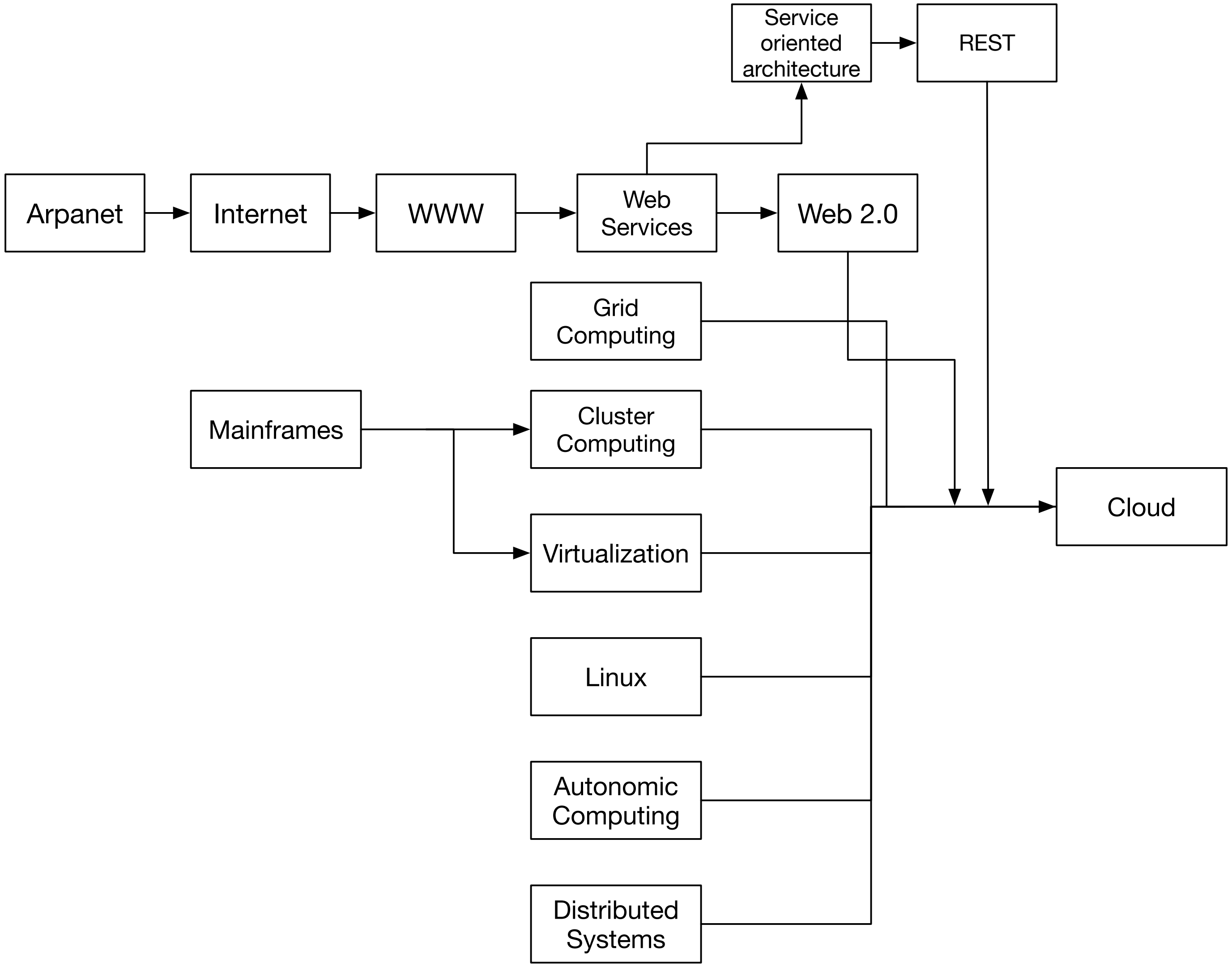
ANDREW GROVE: Say that again.
GUEST: We'd like to have **the whole Web in memory,**
in random access memory.

ANDREW GROVE: That requires a fair amount of memory
(Laughter)
GUEST: The Web, **a good part of the Web,**
is a few terabytes. So it's not unreasonable....
(Laughter)

WHO WAS THE GUEST?

Slide from Paolo Faraboschi

How it all came together



Deployment Models (NIST)

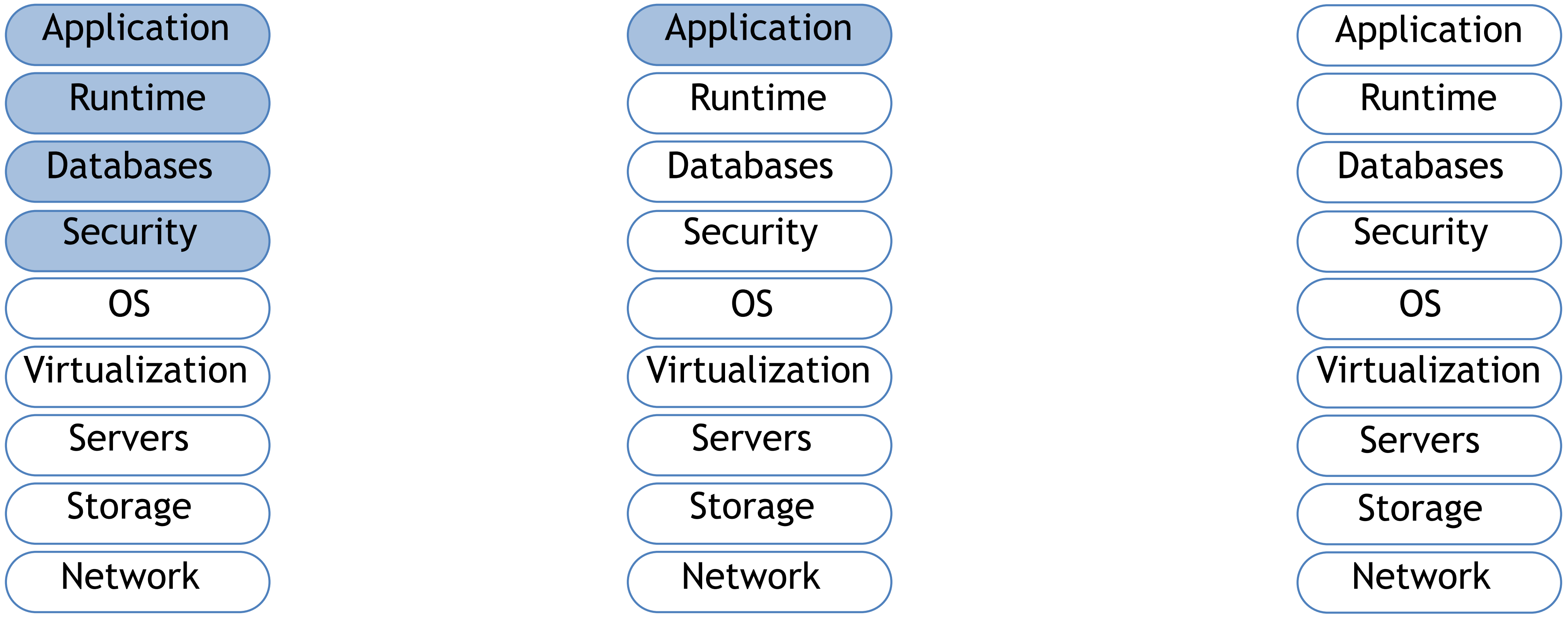
- **Private cloud.** The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

- **Community cloud.** The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

- **Public cloud.** The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.

- **Hybrid cloud.** The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Service Models



IaaS
Infrastructure-as-a-Service

PaaS
Platform-as-a-Service

SaaS
Software-as-a-Service

CNCF Cloud Native Definition v1.0

Approved by TOC: 2018-06-11

中文版本 | 日本語版 | 한국어 | Deutsch| Español | Français | Português Brasileiro | Русский(in Chinese, Japanese, Korean, Brazilian, Portuguese, German, French and Spanish below)

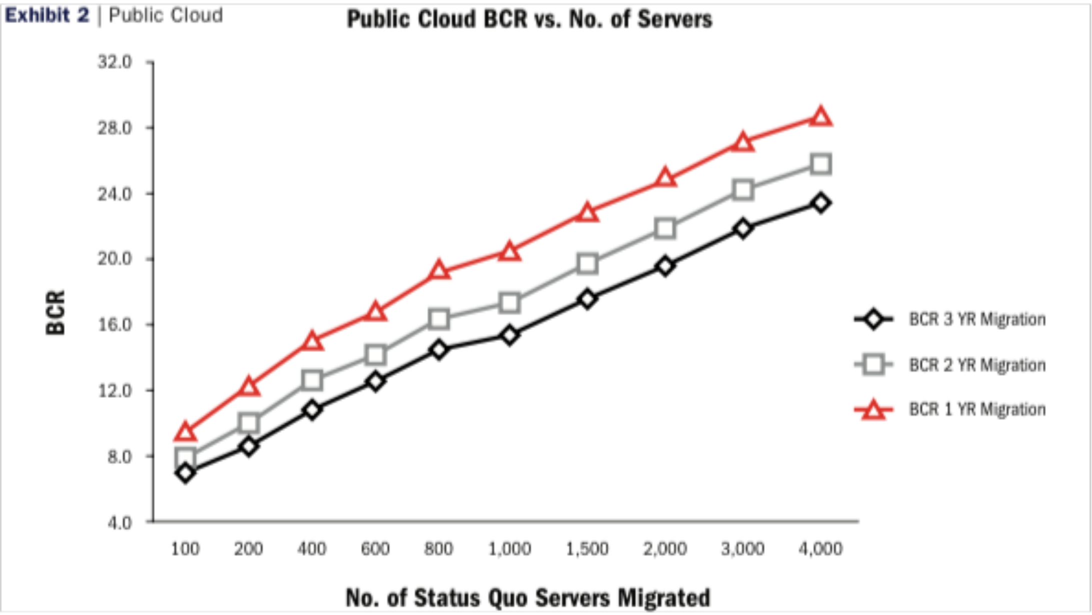
Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

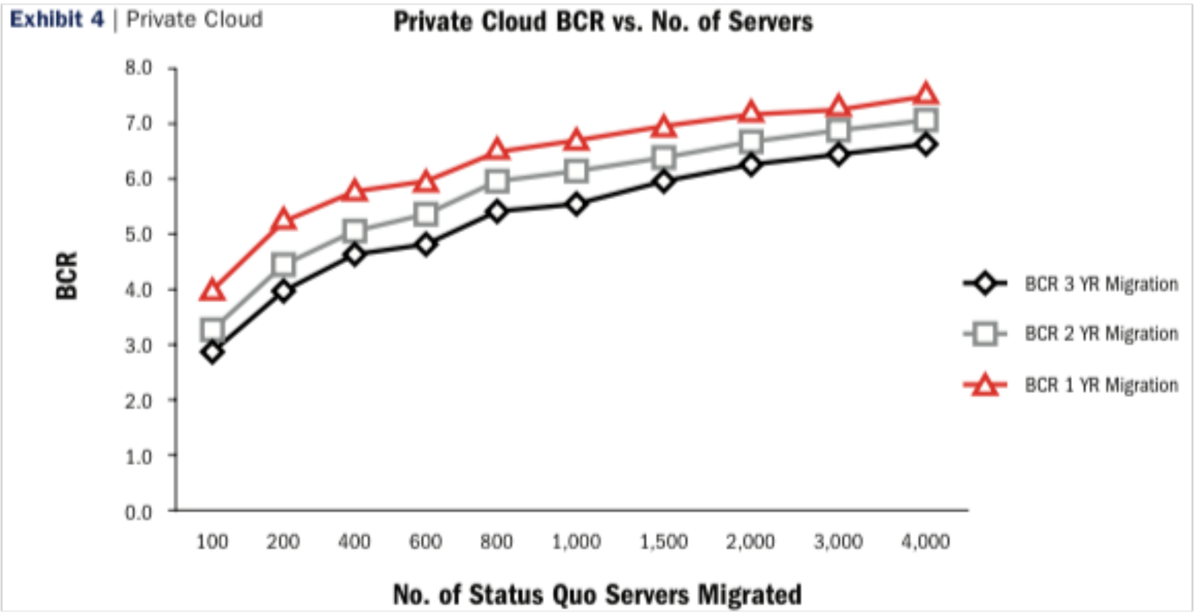
These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

Cost Reduction

US administration moving to cloud saves 7-28 times



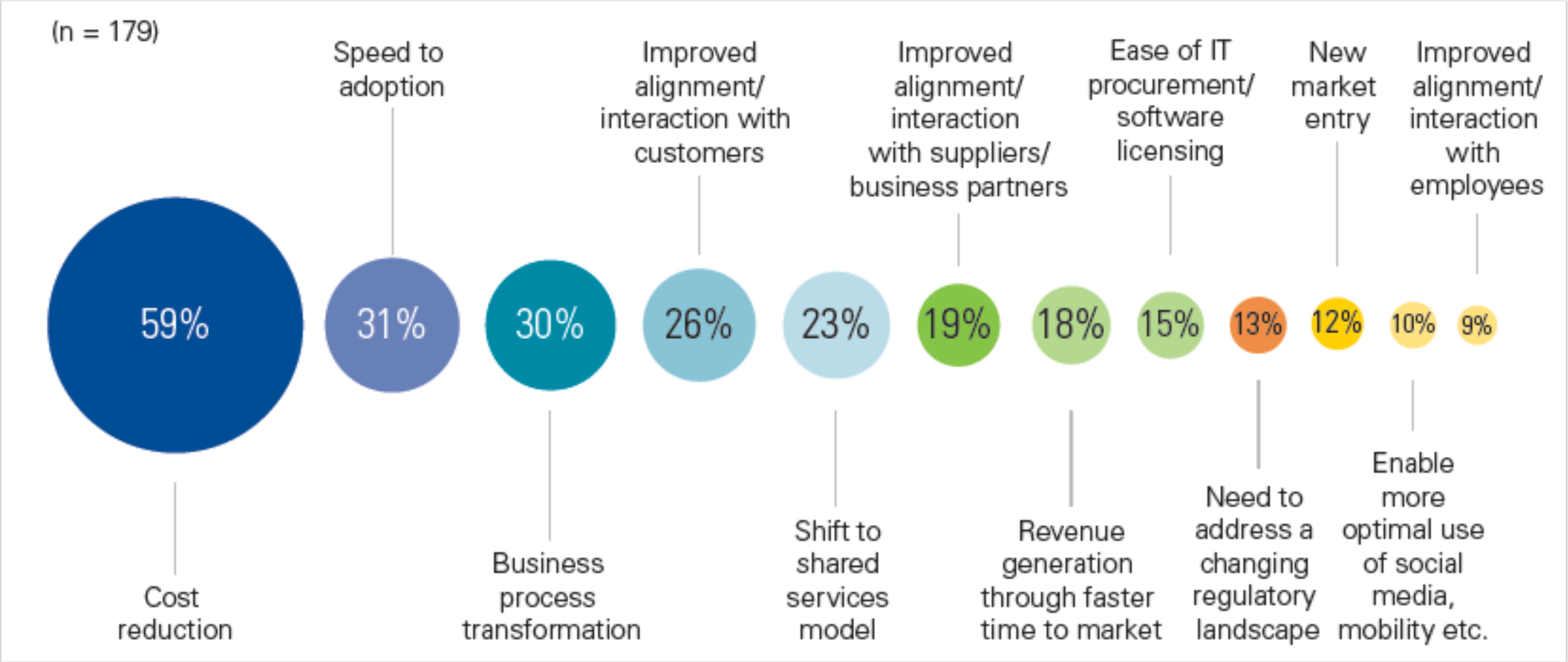


BCR=Benefit-to-cost ratios

Calculated over a 13-year life cycle

Booz-Allen-Hamiliton report "The Economics of cloud computing"

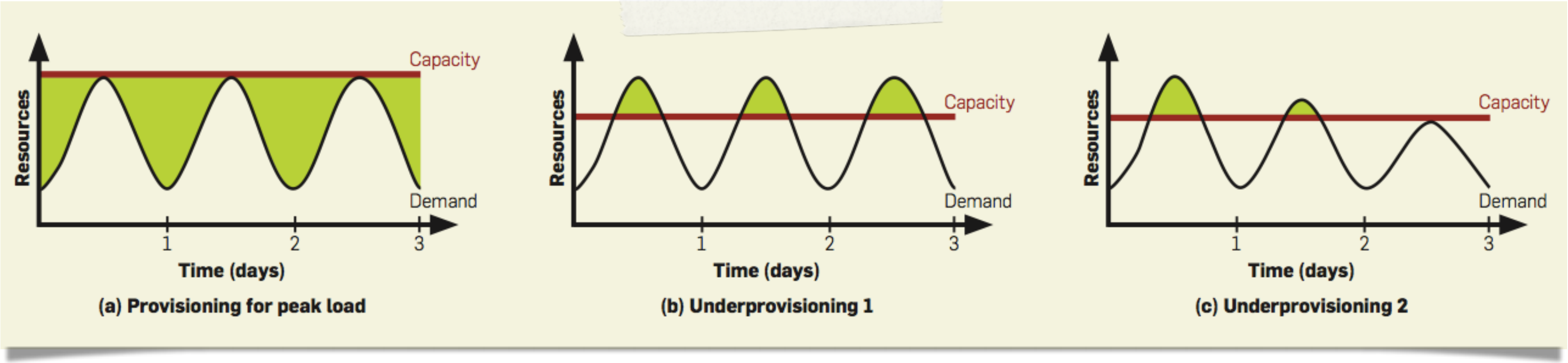
Migration motives



KPMG International's 2012 Global Cloud Provider Survey (n=179)

Difficult to dimension

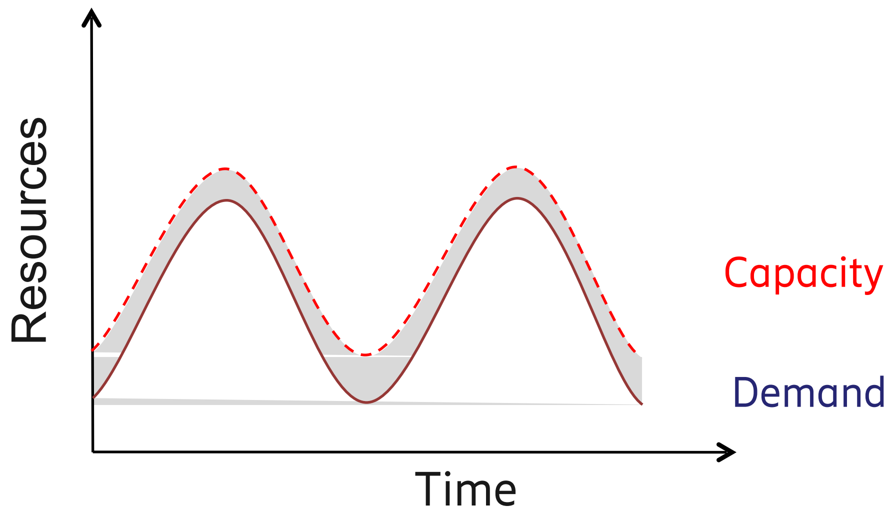
- Workload varies much:
- Death of Michael Jackson: 22% of tweets, 20% of Wikipedia traffic, Google thought they are under attack
- Obama inauguration day: 5x increase in tweets
- Over-provisioning is expensive, under-provisioning may be worse



Rent a Datacenter

Pay by use - Rent a VM!

	vCPU	ECU	Memory (GiB)	Instance Storage (GB)	Linux/UNIX Usage
Ge amazon webservices - Current Generation					
t2.micro	1	Variable	1	EBS Only	$0.013 per Hour
t2.small	1	Variable	2	EBS Only	$0.026 per Hour
t2.medium	2	Variable	4	EBS Only	$0.052 per Hour
m3.medium	1	3	3.75	1 x 4 SSD	$0.070 per Hour
m3.large	2	6.5	7.5	1 x 32 SSD	$0.140 per Hour
m3.xlarge	4	13	15	2 x 40 SSD	$0.280 per Hour
m3.2xlarge	8	26	30	2 x 80 SSD	$0.560 per Hour

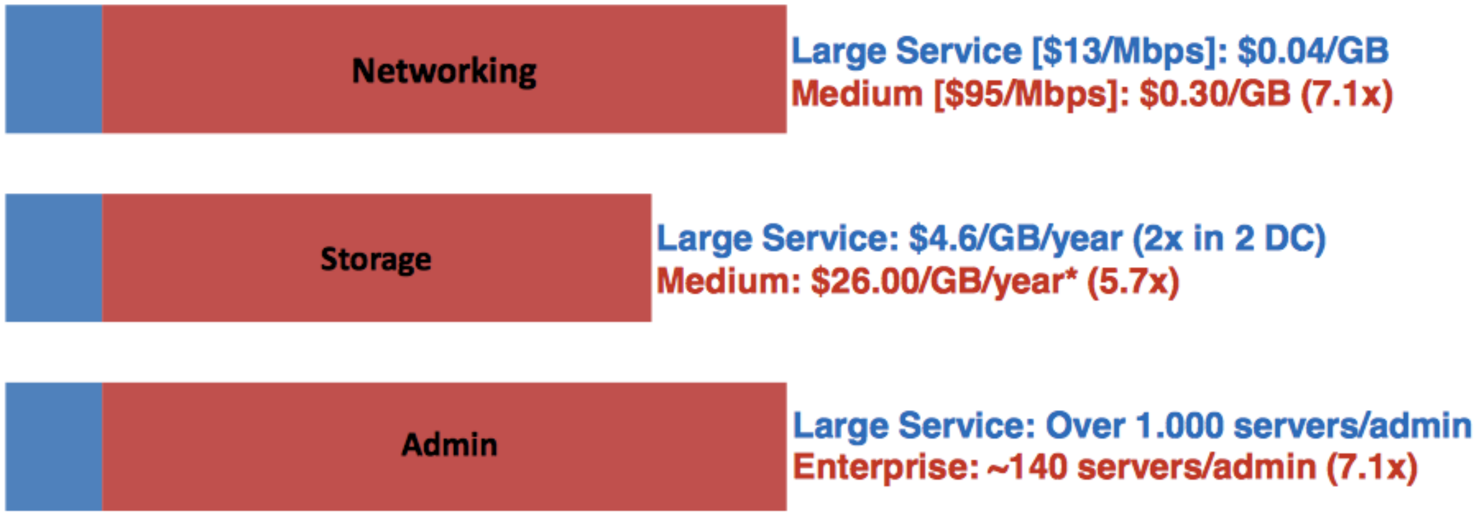


Computing resources in the cloud

1000 machines for 1 hour ⇔ 1 machine for 1000 hours

Bigger is Better

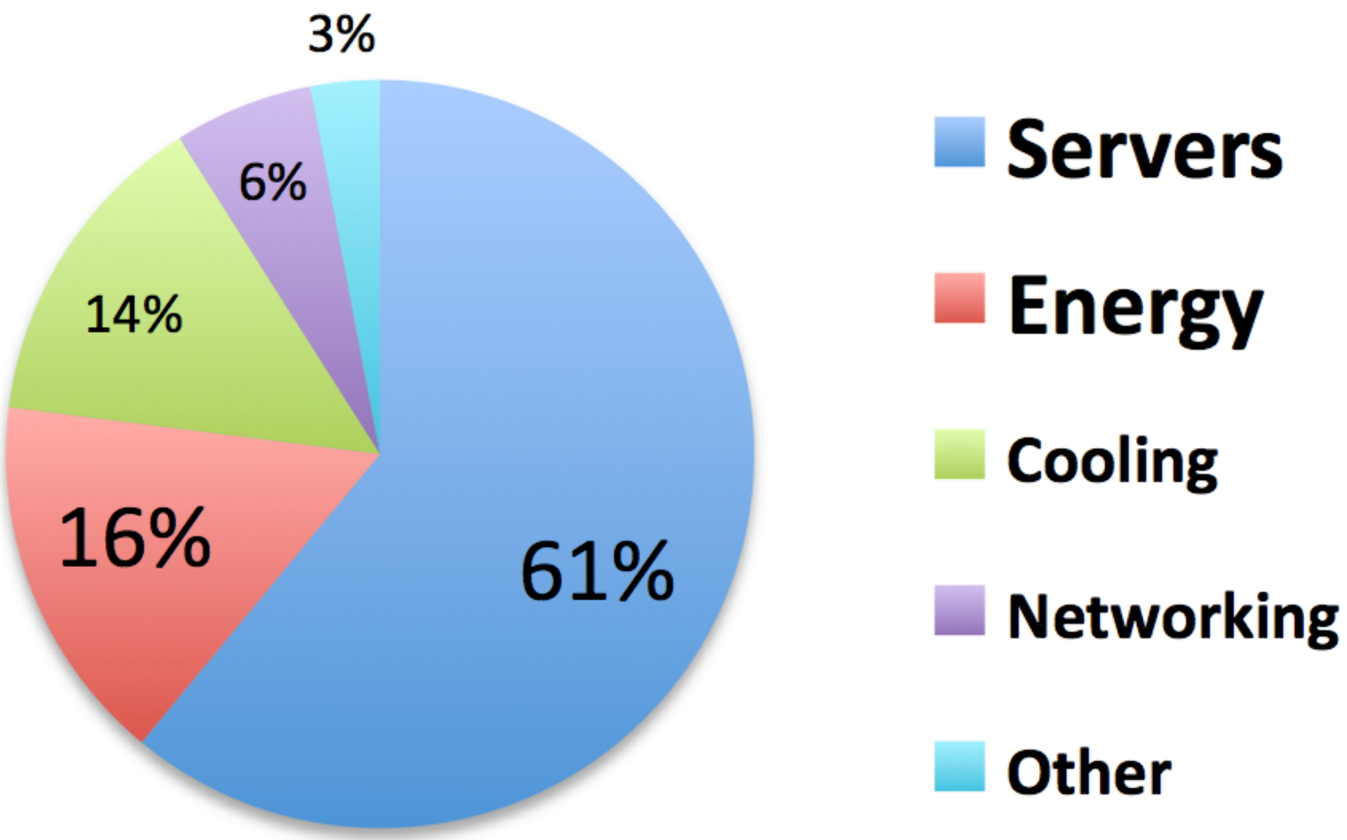
- Substantial economies of scale possible
- Compare a very large service with a small/mid-sized: (~1000 servers):



- High cost of entry
  - Physical plant expensive: 15MW roughly $200M
- Summary: significant economies of scale but at very high cost of entry
  - Small number of large players likely outcome

James Hamilton, *Internet Scale Service Efficiency*, Large-Scale Distributed Systems and Middleware (LADIS) Workshop Sept'08. http:/mvdirona.com

Total Cost of Ownership



[J. Hamilton, http://mvdirona.com]

Obstacles/Opportunities for transitioning to the cloud

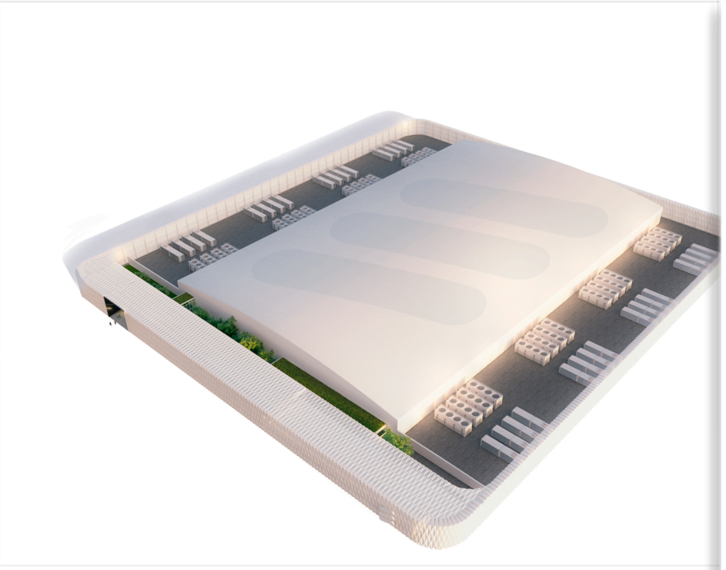
1. Availability
2. Data lock-in
3. Data confidentiality/auditability
4. Data transfer bottlenecks
5. Performance unpredictability
6. Scalable storage
7. Bugs in large-scale distributed systems
8. Scaling quickly
9. Reputation fate sharing
10. Software licensing

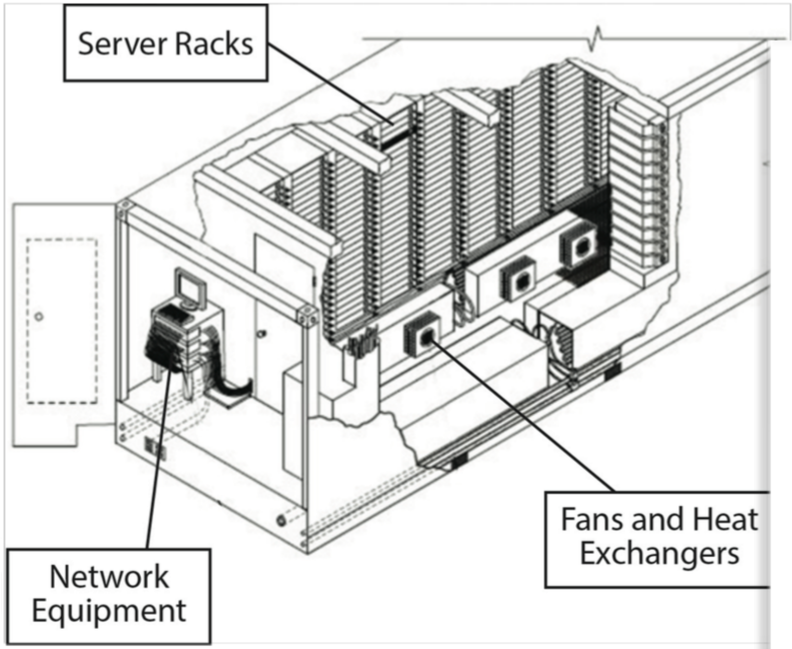
Source: “A View of Cloud Computing”, Armbrust et al

But cloud is more than just a cost saving business model.

It’s a new way of working and designing applications.

The Datacenter



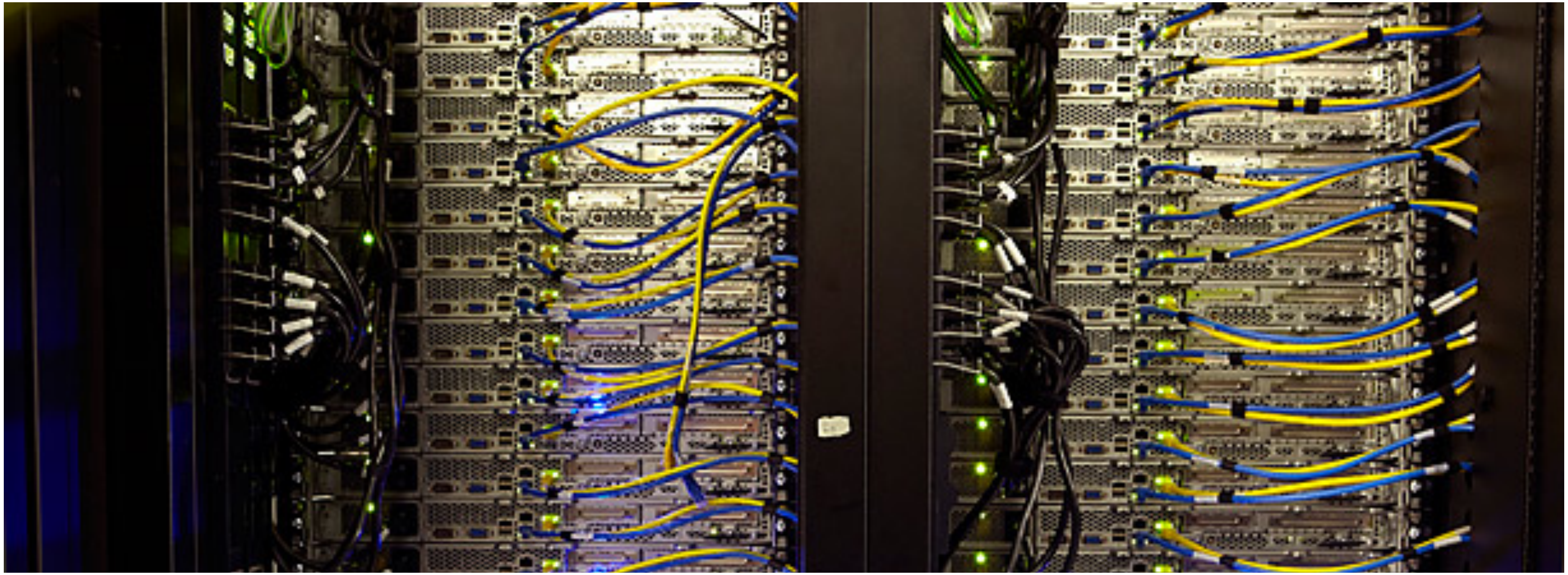


What's inside?

Racks

What's inside?

Networking

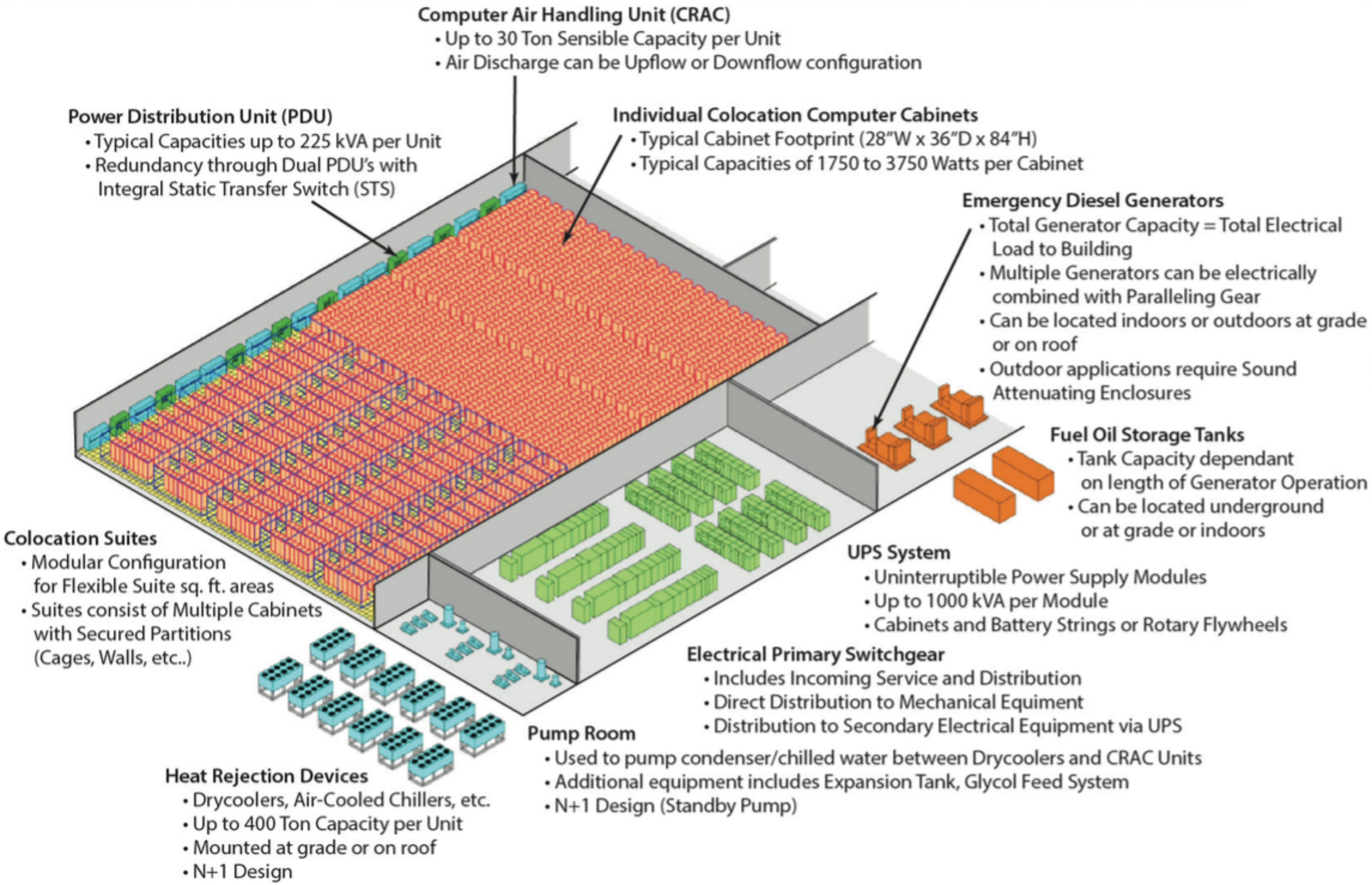


What's inside?

Power supplies

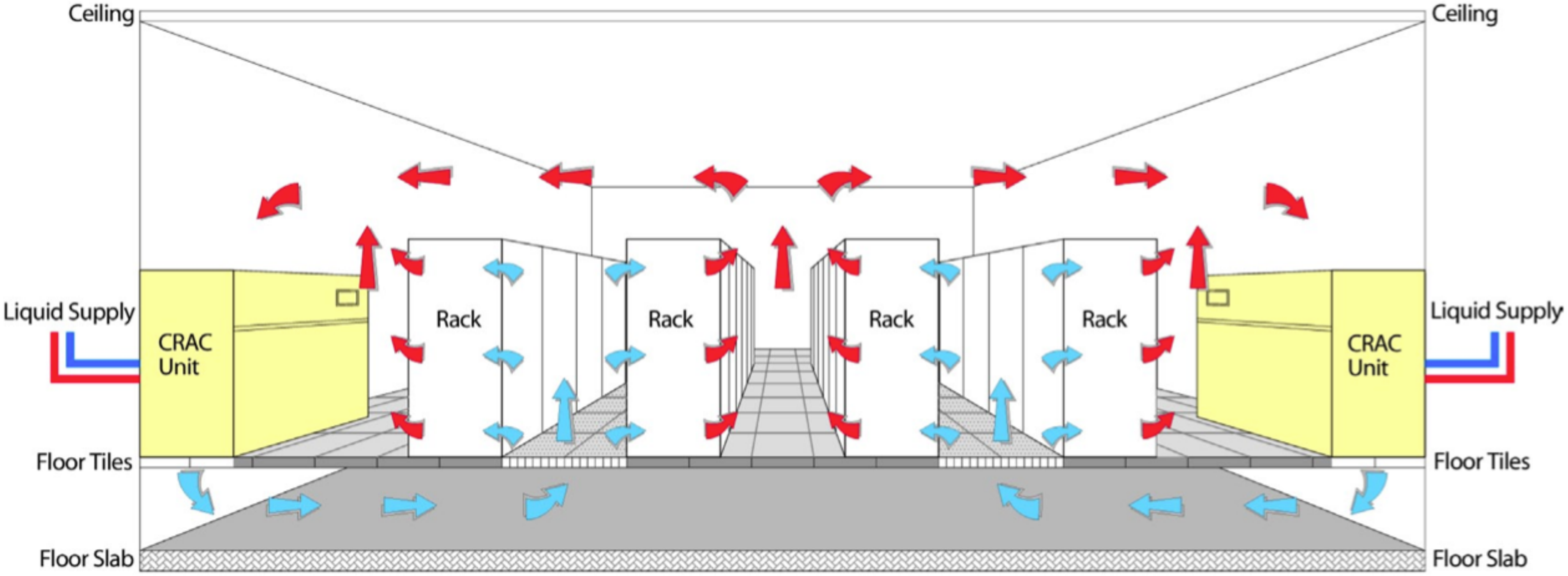
What's inside?

Cooling



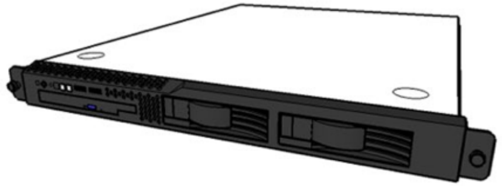
Source: "The Datacenter as a computer", Barroso et al

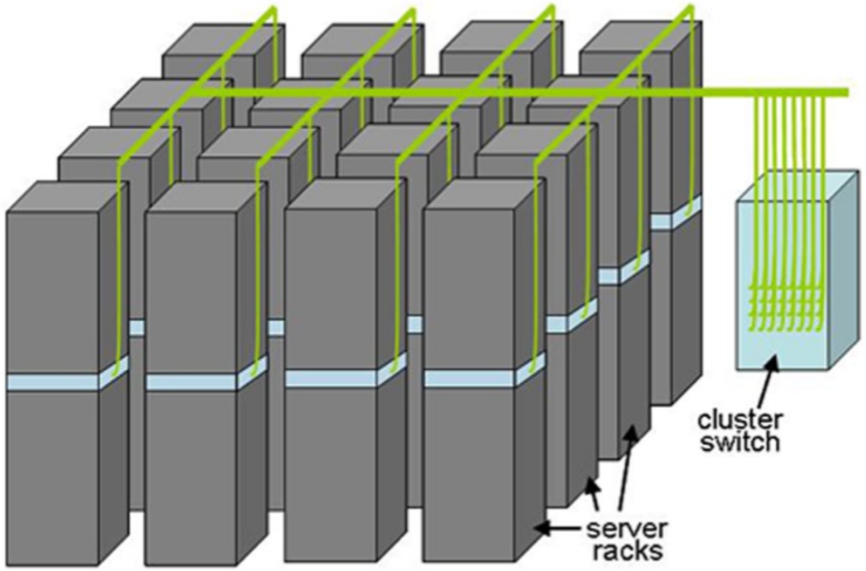
Computer Architecture



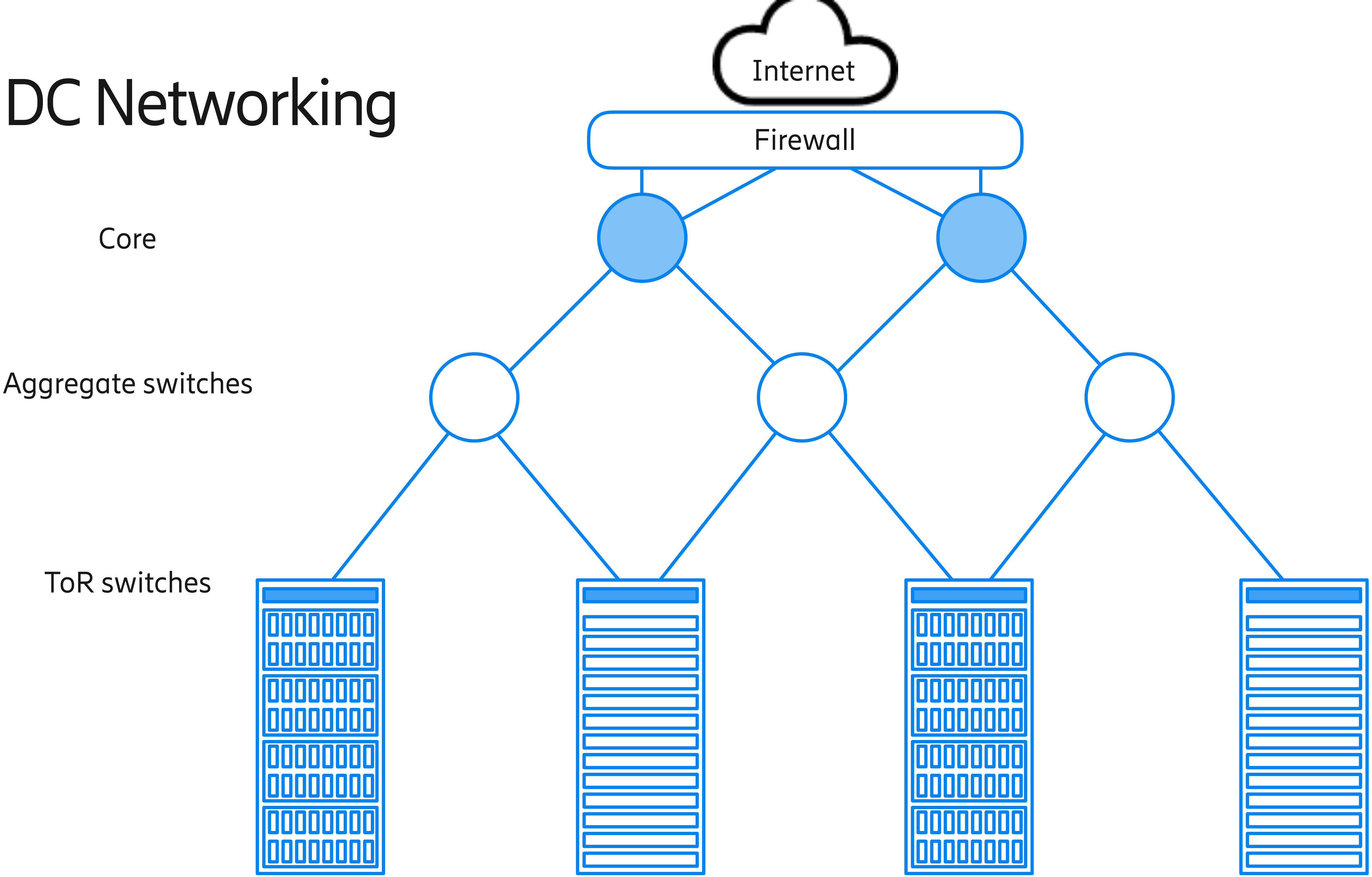
Source: "The Datacenter as a computer", Barroso et al

Datacenter Elements

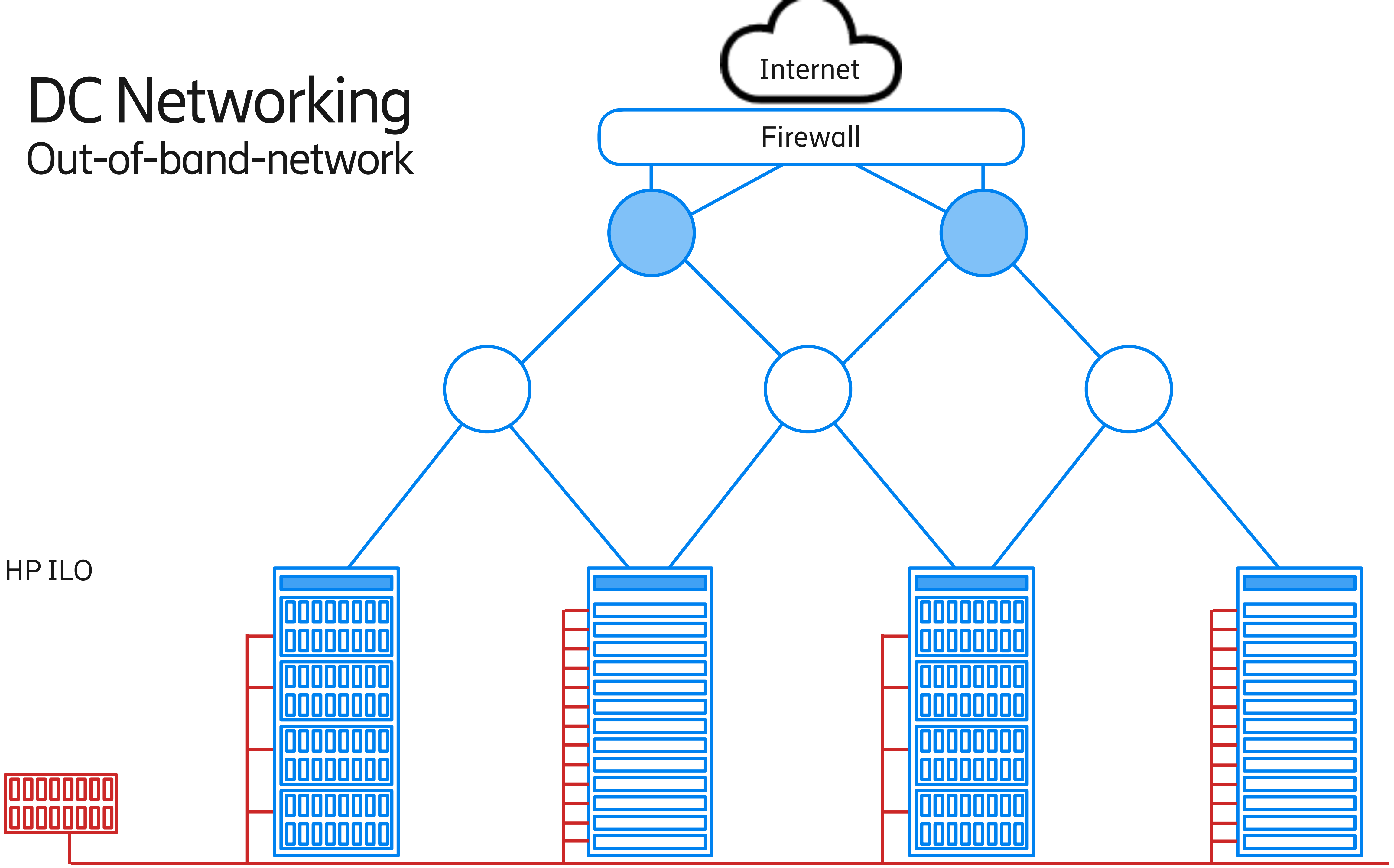




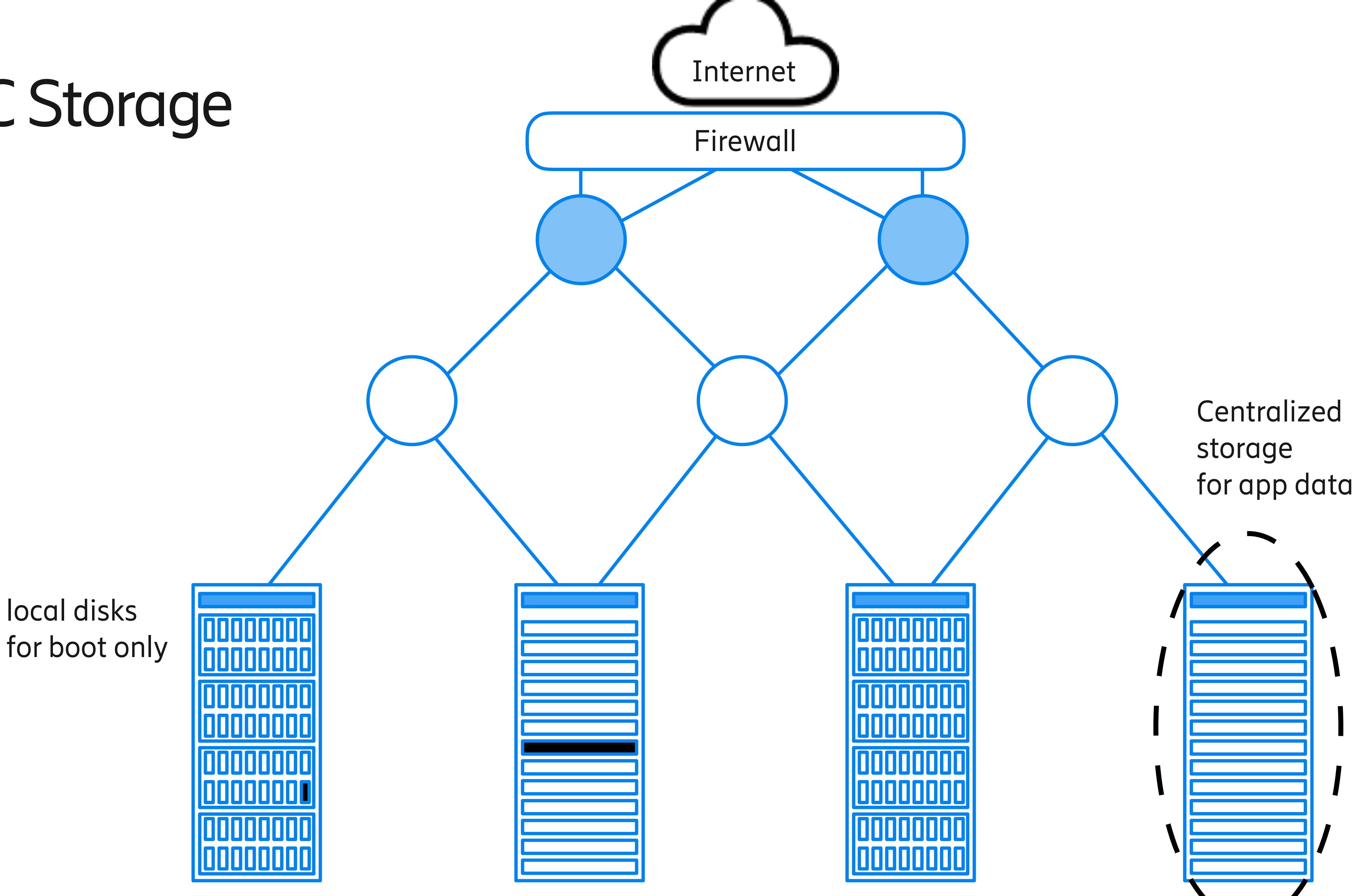
DC Networking



DC Networking
Out-of-band-network

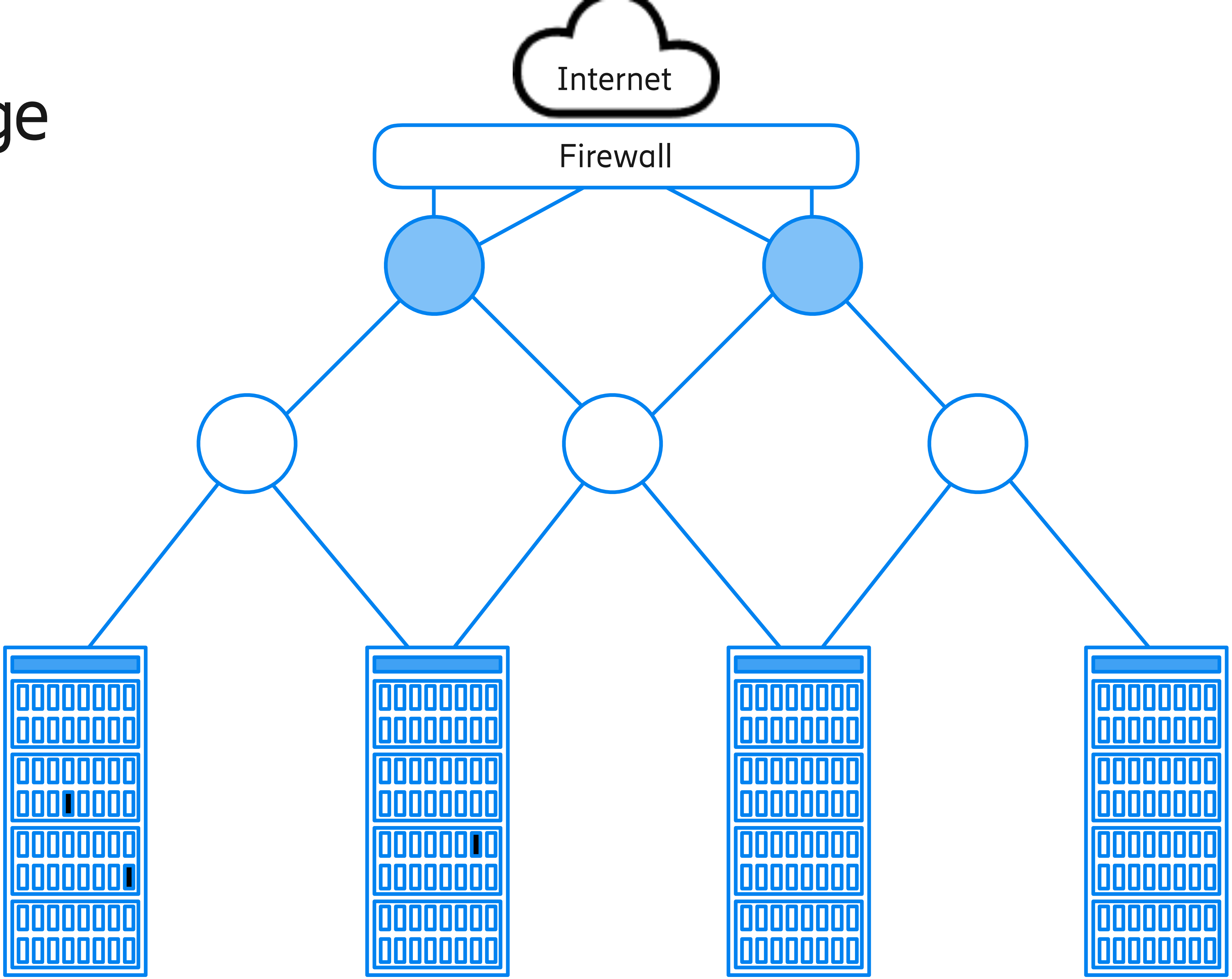


DC Storage



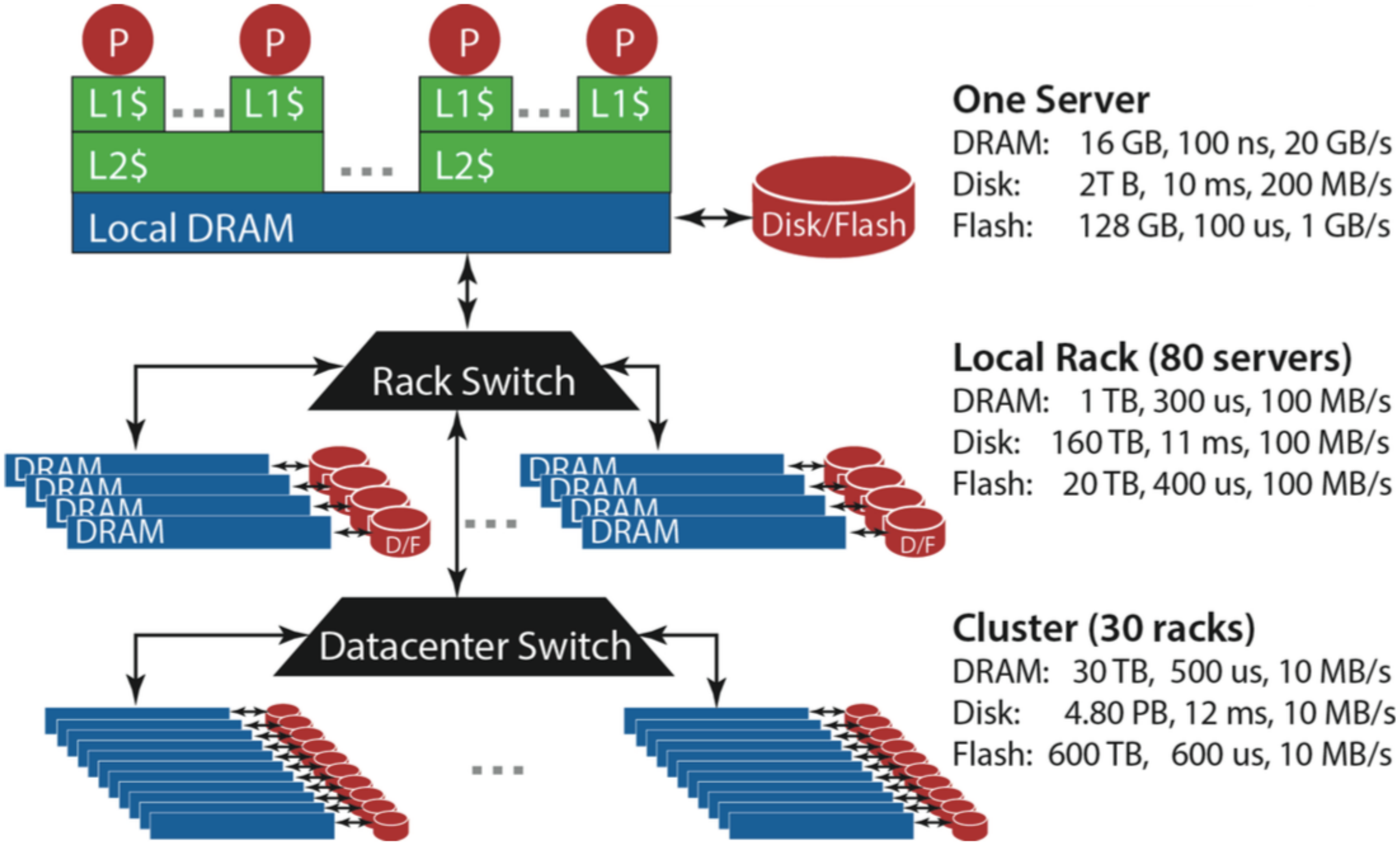
Internet
Firewall
Centralized
storage
for app data
local disks
for boot only

DC Storage



Distributed file system
Multiple copies
Varying latency

Storage at Google

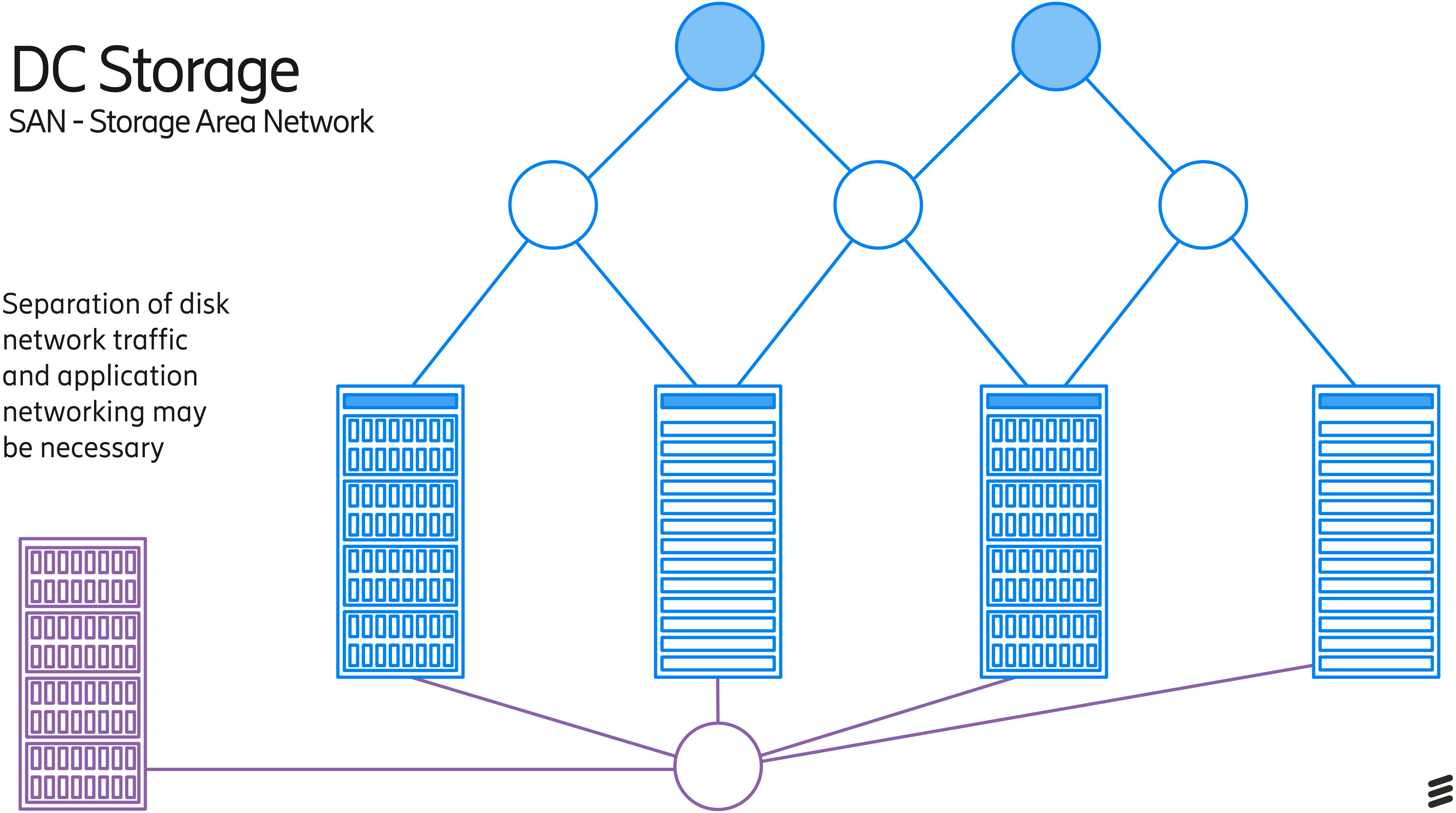


Source: "The Datacenter as a computer", Barroso et al

DC Storage

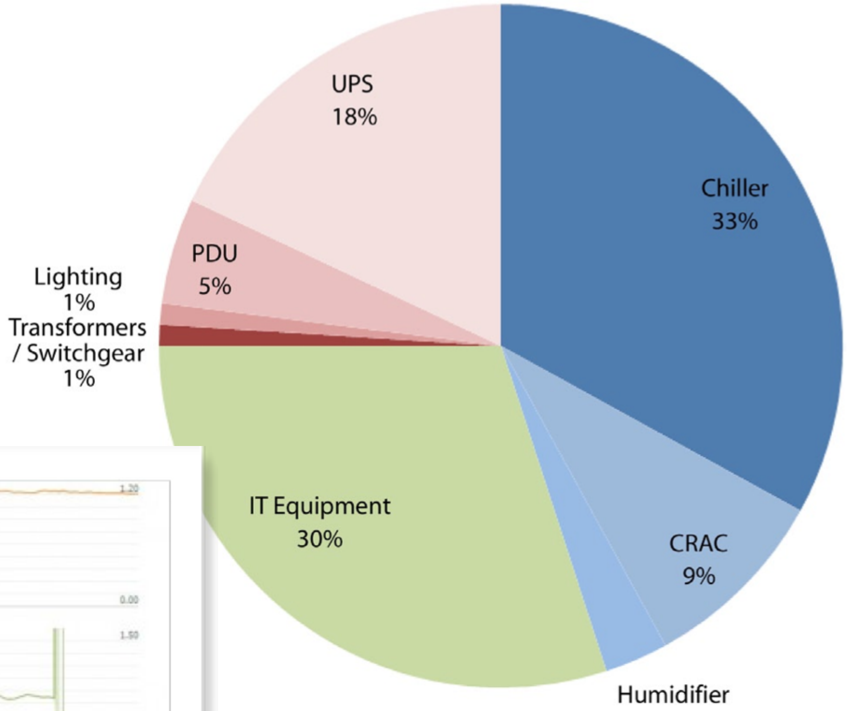
SAN - Storage Area Network

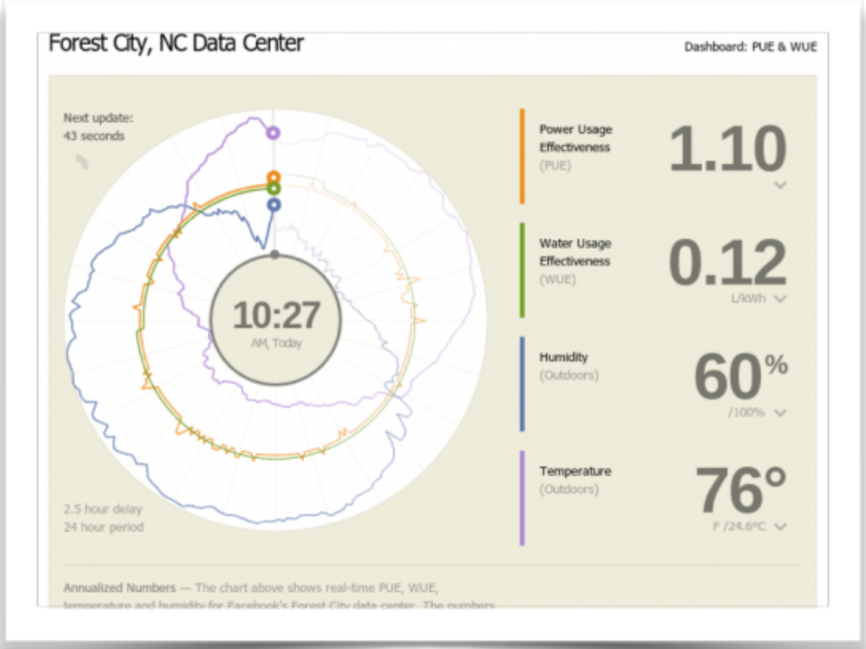
Separation of disk network traffic and application networking may be necessary



DC Efficiency

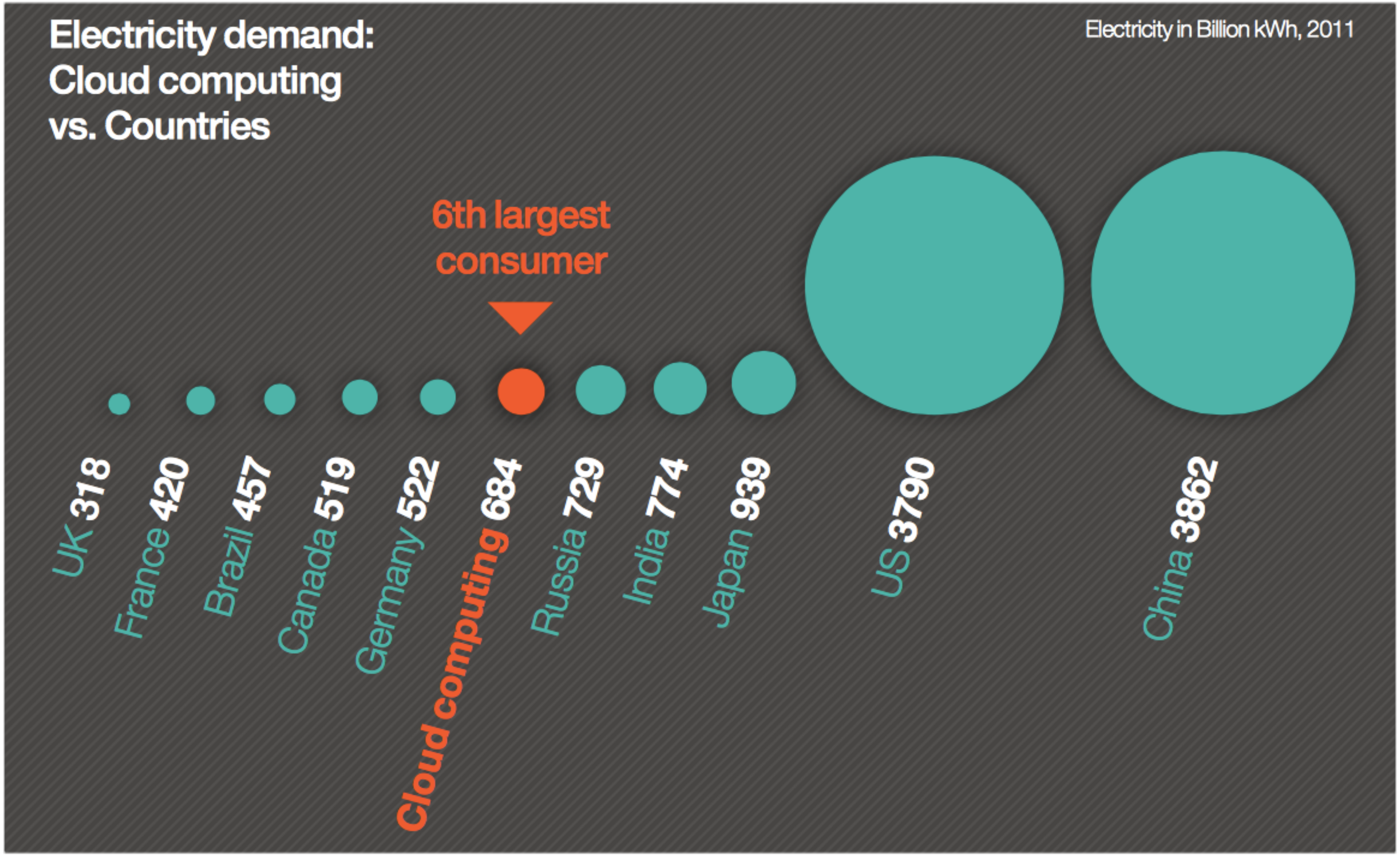
https://www.facebook.com/PrinevilleDataCenter/app_399244020173259





$$WUE = \frac{\text{Annual Water Usage}}{\text{IT Equipment Energy}}$$

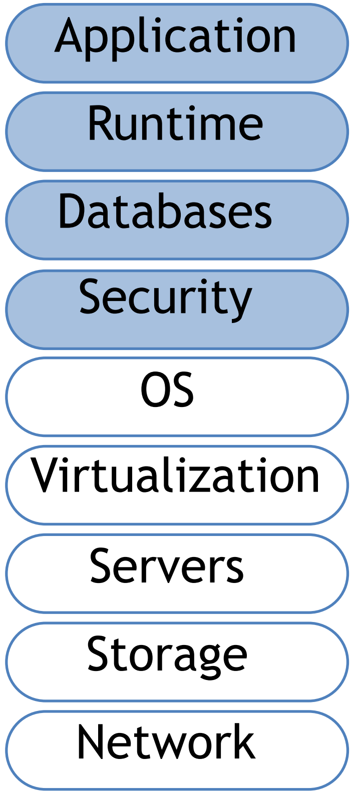
$$PUE = \frac{\text{Total Facility Energy}}{\text{IT Equipment Energy}}$$



source: Greenpeace - Clicking Clean April 2014

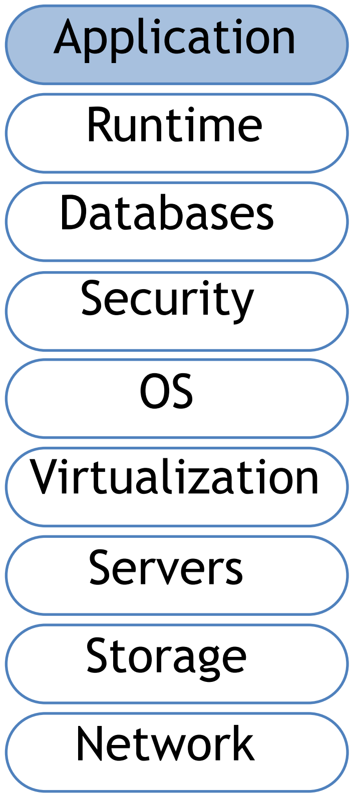
Infrastructure-as-a-Service

- Provide a virtual datacenter
- Compute/storage/network
- Often some basic services also
- The user is responsible for making the application run correctly, i.e. fault tolerance, timing, handling crashes, scaling, authentication, redundancy, etc.
- Amazon Web Services, Google Compute, Rackspace



Platform-as-a-Service

- Cloud provides runtime/middleware
  - Java VM, Python VM, JS VM
  - Databases, communication, etc.
- User does not manage/control application infrastructure (network, servers, OS, etc.)
- PaaS handles scale-out
- Customer pays SaaS provider for the service; SaaS provider pays the cloud for the infrastructure
- Example: Windows Azure, Google App Engine, Examples: Google App Engine, Node.js, Map Reduce



Google App Engine

Google App Engine

- App Engine invokes your app's servlet classes to handle requests and prepare responses in this environment.
- Add
  - Servlet classes, (*.java)
  - JavaServer Pages (*.jsp),
  - Your static files and data files,
  - A deployment descriptor (the web.xml file)
- Auto scale to many billion requests per day

Python Java PHP Go

```java
public class GuestbookServlet extends HttpServlet {
  @Override
  public void doGet(HttpServletRequest req, HttpServletResponse resp)
      throws IOException {

    UserService userService = UserServiceFactory.getUserService();
    User currentUser = userService.getCurrentUser();

    if (currentUser != null) {
      resp.setContentType("text/plain");
      resp.getWriter().println("Hello, " + currentUser.getNickname());
    } else {
      resp.sendRedirect(userService.createLoginURL(req.getRequestURI()));
    }
  }
}
```

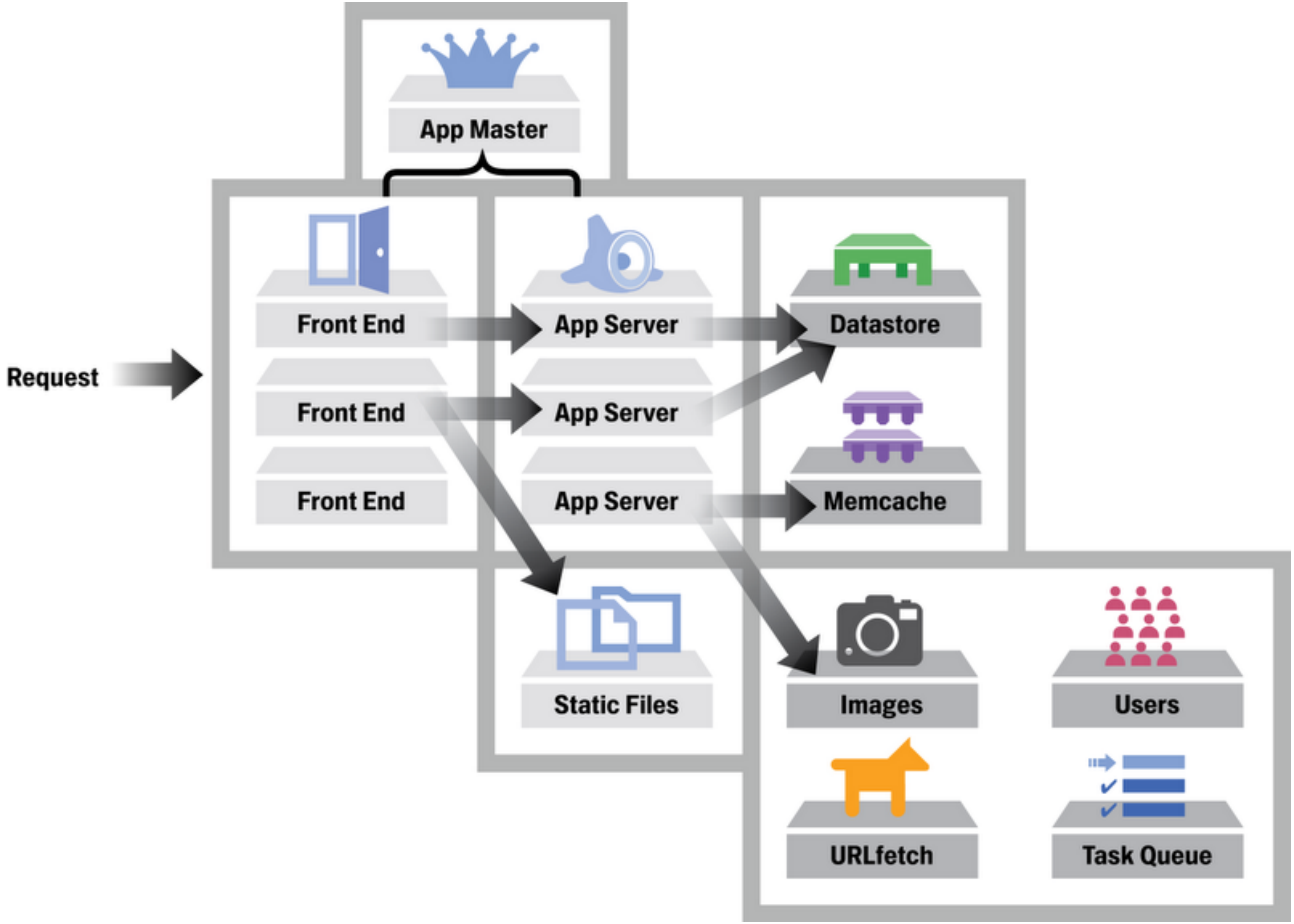
```python
import webapp2

class MainPage(webapp2.RequestHandler):
    def get(self):
        self.response.headers['Content-Type'] = 'text/plain'
        self.response.write('Hello, World!')

application = webapp2.WSGIApplication([('/', MainPage),], debug=True)
```

```yaml
application: your-app-id
version: 1
runtime: python27
api_version: 1
threadsafe: true

handlers:
- url: /.*
  script: helloworld.application
```



App Master
Request
Front End
Front End
Front End
App Server
App Server
App Server
Datastore
Memcache
Static Files
Images
Users
URLfetch
Task Queue
Google App Engine

Software-as-a-Service

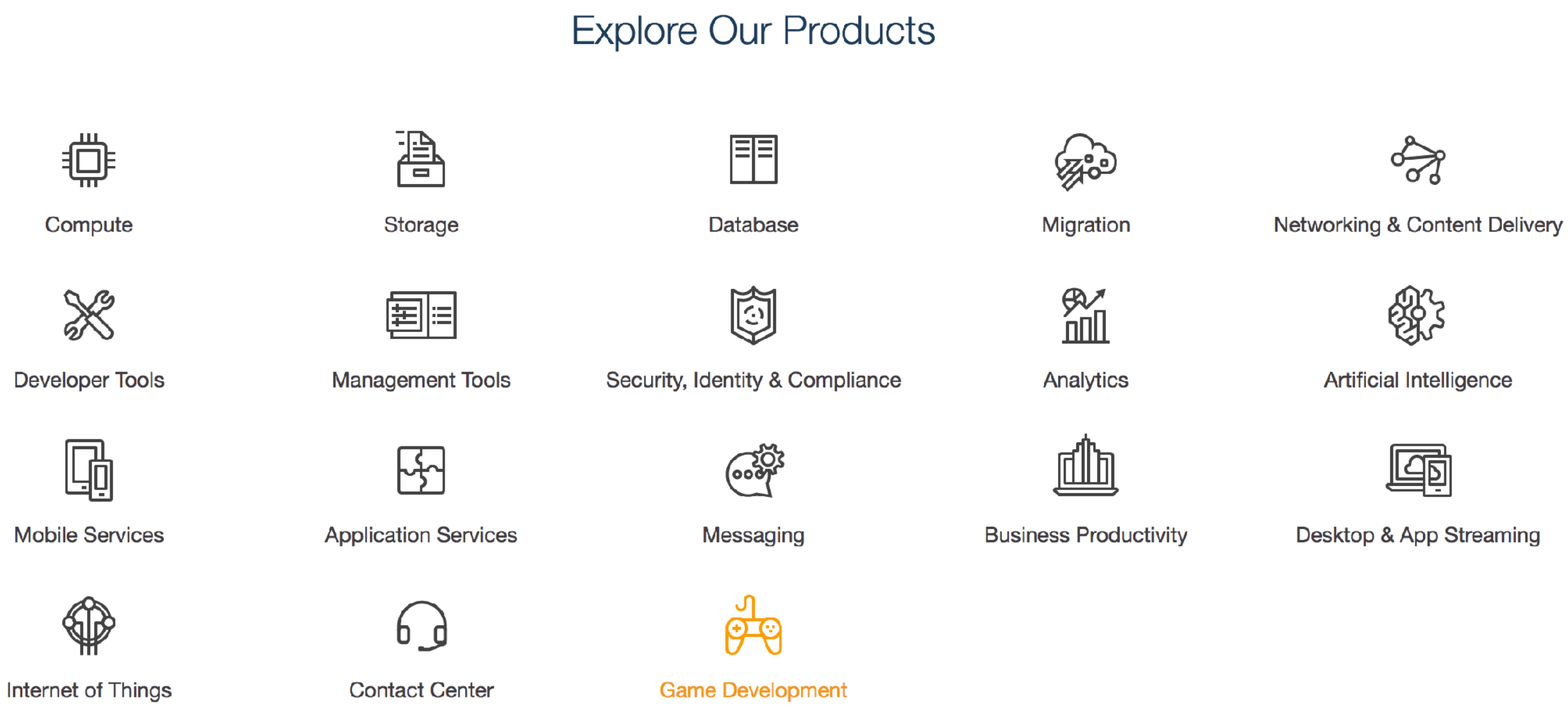
- Cloud provides an entire application
  - Often running in the browser
- Application and data hosted centrally
  - No installation, zero maintenance
  - No control (no need for a sysop)
- Example:
  - Google Apps, Word processor, presentation, spreadsheet, calendar, photo, CRM, Dropbox Paper

Application
Runtime
Databases
Security
OS
Virtualization
Servers
Storage
Network

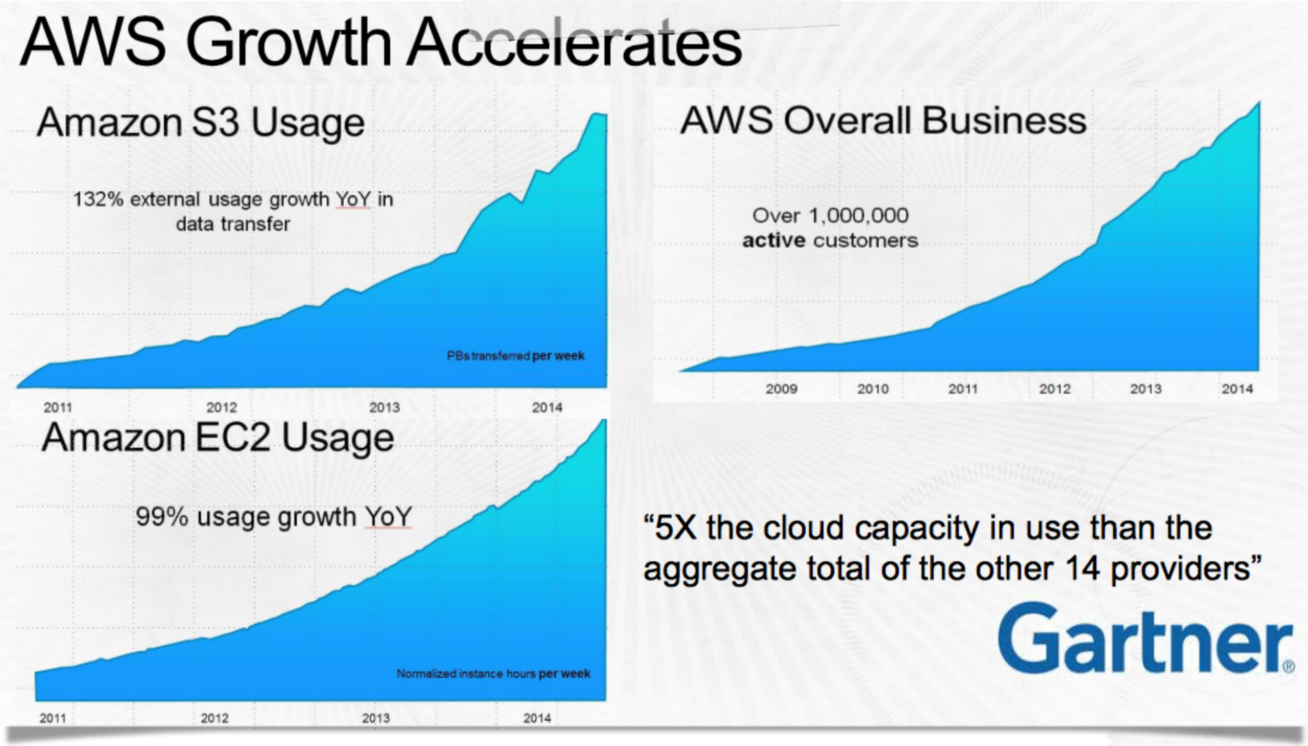
Differences between SaaS and traditional models

	Traditional	SaaS
price	one time (upfront)	subscription
accessibility	local machine	internet
upgrades	manual	managed
deployment	local IT	managed
security	local IT	managed
data storage	on-prem	managed
vendor incentive	initial sale & upgrades	high-subscription rate

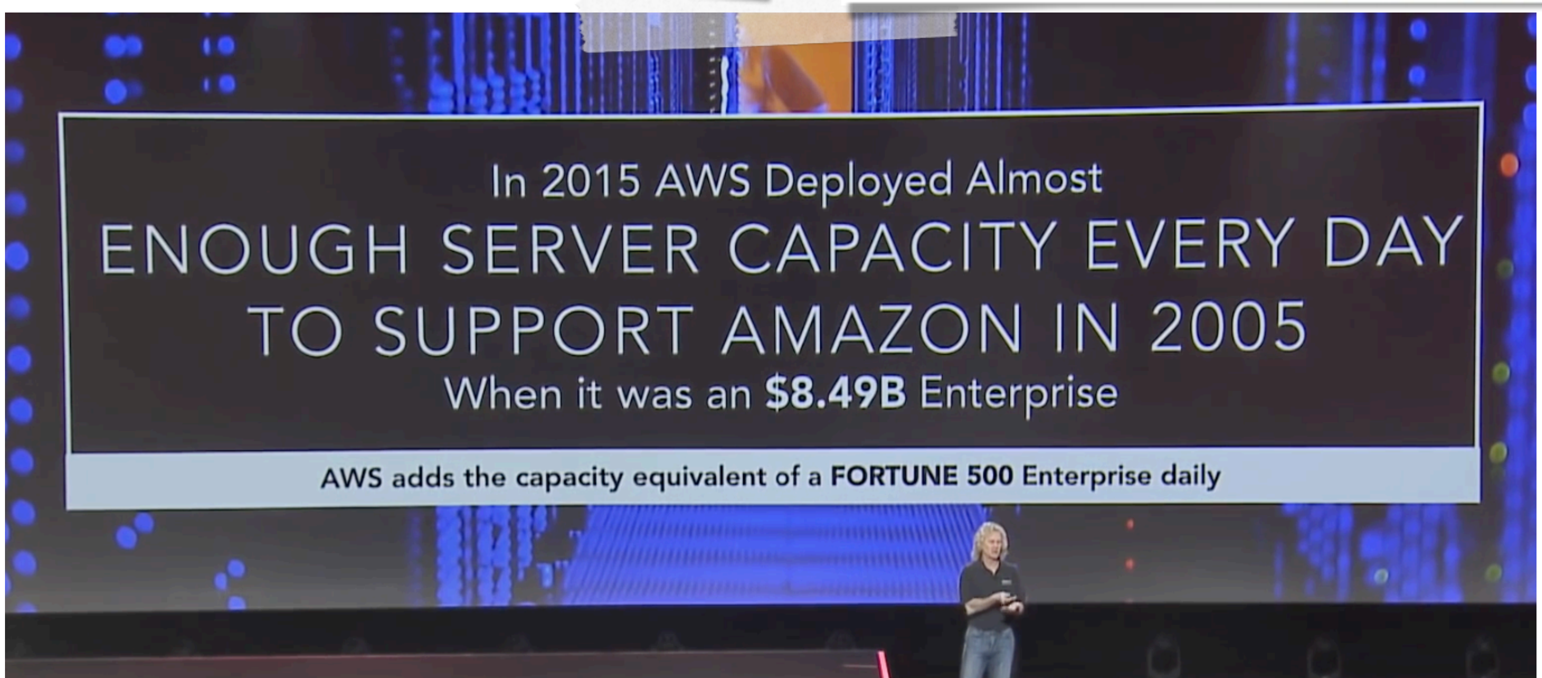
Amazon AWS today



Source: Gartner (July 2019)

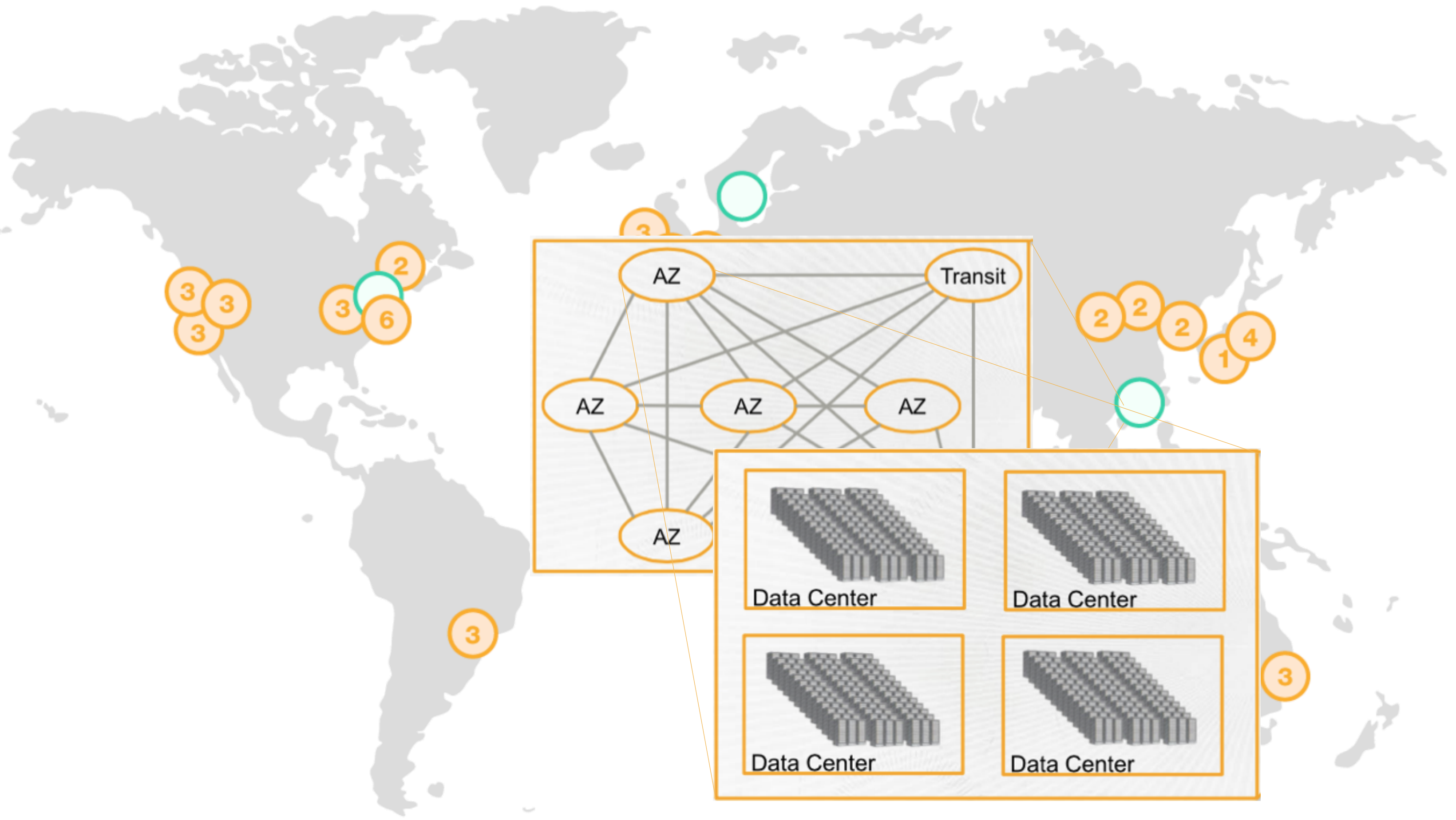

AWS Growth Accelerates
Amazon S3 Usage
132% external usage growth YoY in data transfer
PBs transferred per week
2011
2012
2013
2014
AWS Overall Business
Over 1,000,000 active customers
2009
2010
2011
2012
2013
2014
Amazon EC2 Usage
99% usage growth YoY
Normalized instance hours per week
2011
2012
2013
2014
"5X the cloud capacity in use than the aggregate total of the other 14 providers"
Gartner


Amazon.com's revenue & profit 2004-2014
Revenue
Net profit
Millions of Dollars
$90,000
80,000
70,000
60,000
50,000
40,000
30,000
20,000
10,000
0
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
Note: 2013 and 2014 are estimates.


In 2015 AWS Deployed Almost
ENOUGH SERVER CAPACITY EVERY DAY
TO SUPPORT AMAZON IN 2005
When it was an $8.49B Enterprise
AWS adds the capacity equivalent of a FORTUNE 500 Enterprise daily

Amazon Infrastructure

- 18+ Regions, connected by private fiber
- Regions consists of 2 or more availability zones (AZ)
- 54+ Az
- AZ < 2 ms apart and usually <1ms
- AZ is one or more DCs
- DC consists of 50-80.000 machines
- Inter AZ DCs < 1/4 ms apart



source: Jame Hamilton, ReInvent 20XX

AWS Compute

- EC2 (Elastic Compute Cloud)
  - Linux or Windows VM
  - Several types:
    - On-demand instances
    - Reserved Instances - long term, low price
    - Spot Instances" - you bid on a VM
    - Dedicates Instances - single tenant HW

	Linux/UNIX Usage
Standard On-Demand Instances	
m1.small	$0.048 per Hour
m1.medium	$0.096 per Hour
m1.large	$0.193 per Hour
m1.xlarge	$0.385 per Hour
Second Generation Standard On-Demand Instances	
m3.medium	$0.077 per Hour
m3.large	$0.154 per Hour
m3.xlarge	$0.308 per Hour
m3.2xlarge	$0.616 per Hour

Service Commitment

AWS will use commercially reasonable efforts to make Amazon EC2 and Amazon EBS each available with a Monthly Uptime Percentage (defined below) of at least 99.95%, in each case during any monthly billing cycle (the "Service Commitment"). In the event Amazon EC2 or Amazon EBS does not meet the Service Commitment, you will be eligible to receive a Service Credit as described below.

4.3 hours downtime/year

AWS Storage

Storage Pricing

Region: US Standard

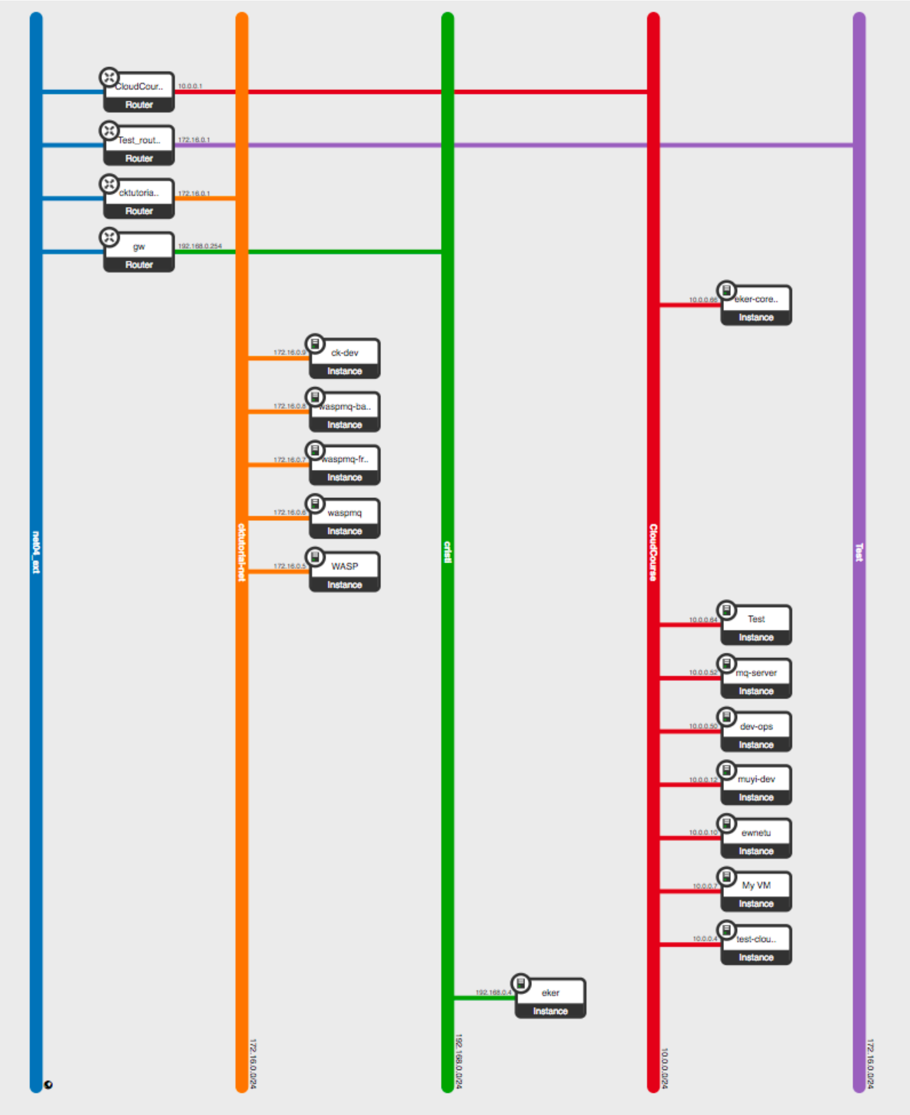
	Standard Storage	Reduced Redundancy Storage	Glacier Storage
First 1 TB / month	$0.0300 per GB	$0.0240 per GB	$0.0100 per GB
Next 49 TB / month	$0.0295 per GB	$0.0236 per GB	$0.0100 per GB

Dropbox price: 1TB $9.99/month
Dropbox used one AWS S3 bucket for all its customers!

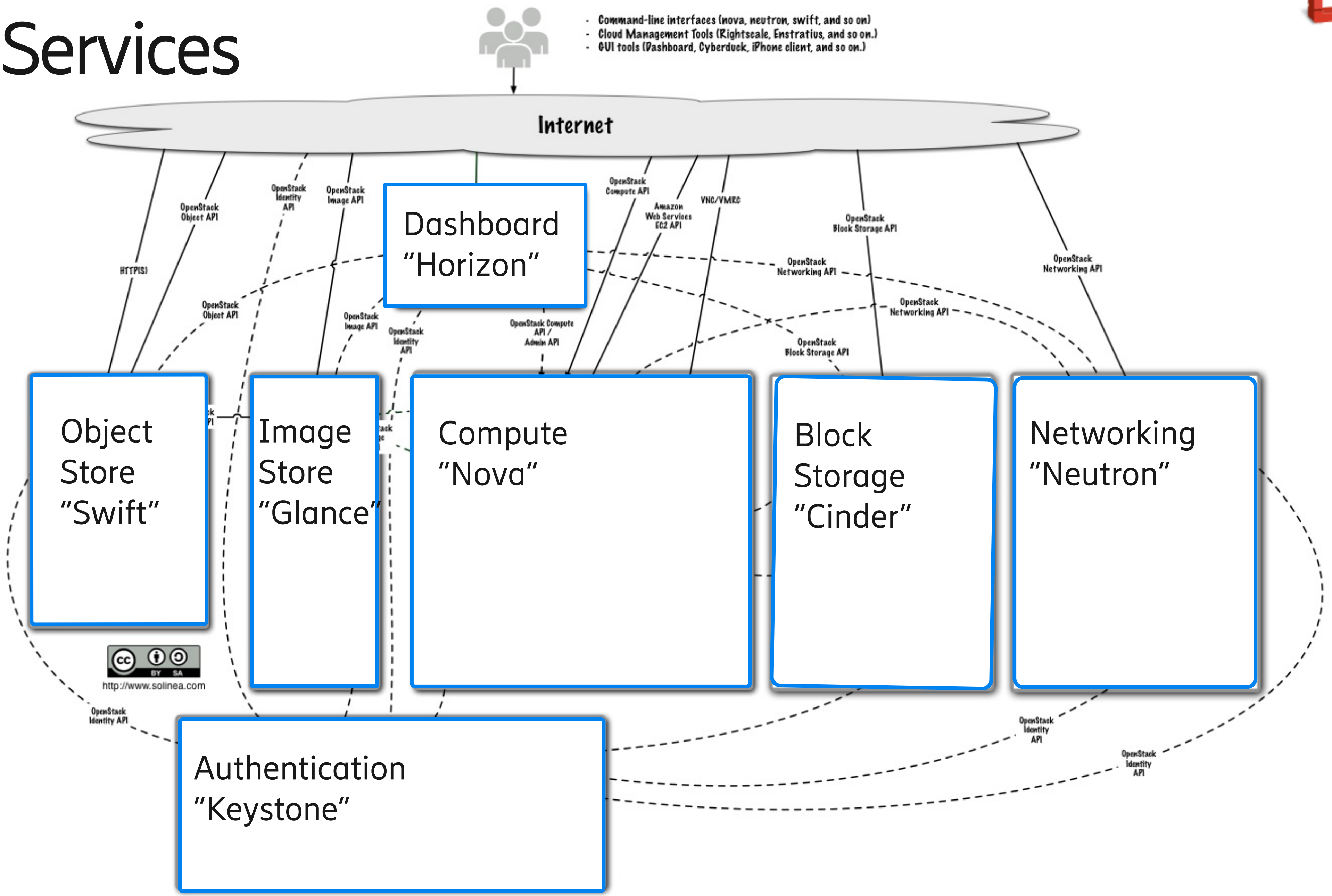
OpenStack

— Open source cloud management project
— Launched in 2010 by Rackspace and NASA with in
— Open source under Apache license
— A number of distributions
— IaaS - Infrastructure-as-a-Service

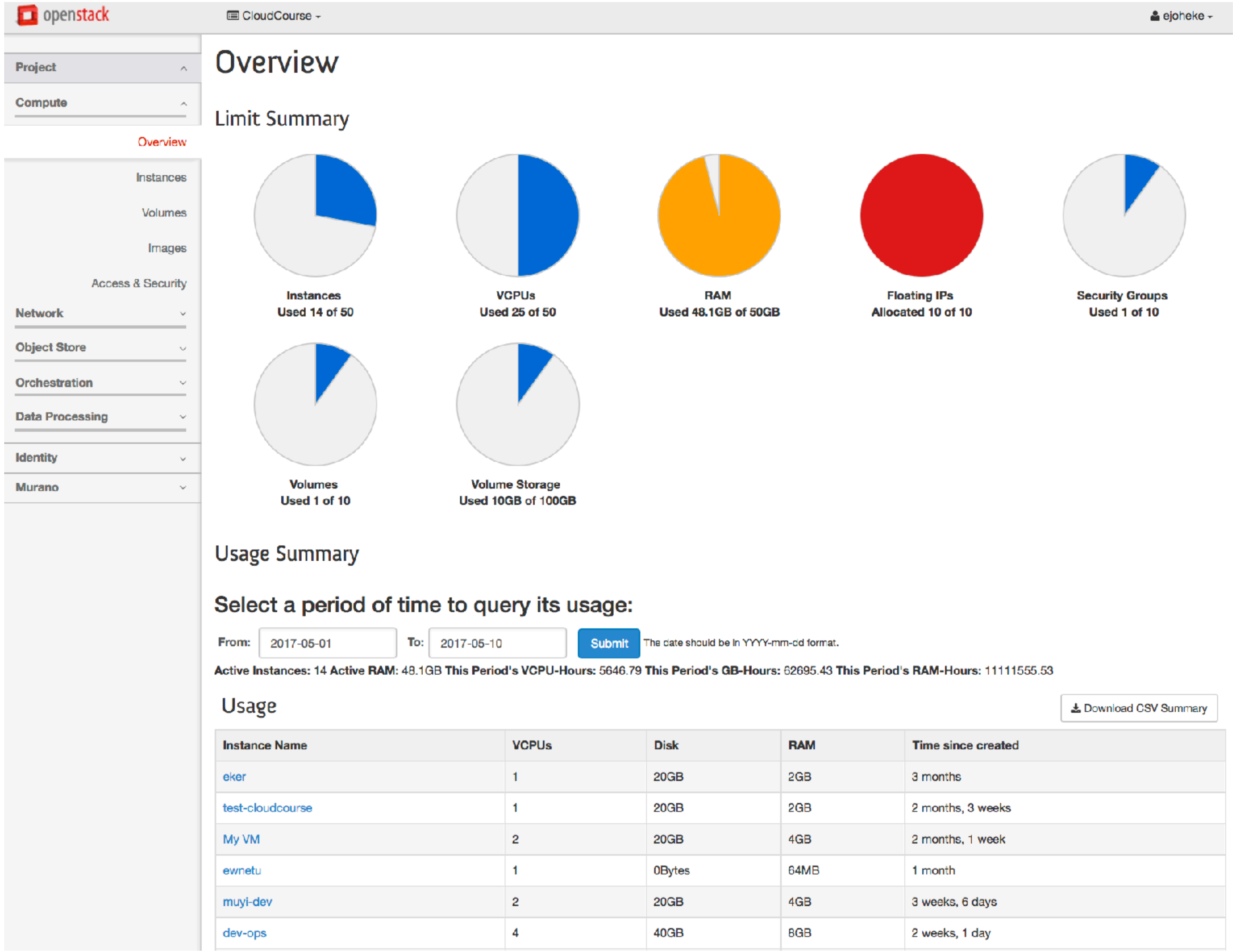
— Tenants/projects & users



Core Services

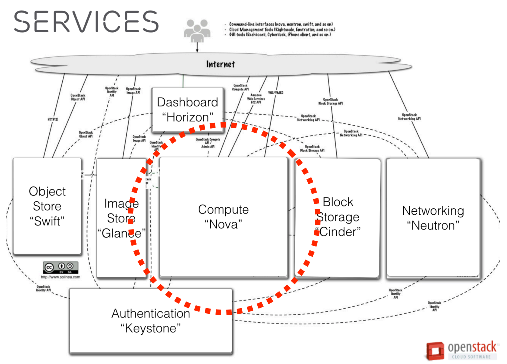


Dashboard: "Horizon"



https://xerces.ericsson.net

Compute: "Nova"



- Manage and automate pools of computer resources
  - Life cycle of VM instances
  - Keeps track of resources (virtual & real)

- Nova does not provide any virtualization capabilities,
  - Uses libvirt API to interact with supported hypervisors.
  - Hypervisor agnostic (Xen, KVM, VMware, Hyper-V, etc.)

- Decides where to allocate instances (Nova-Schedule) openstack. xerces • Datahub ▾ ejoheke ▾

- Project
  - API Access
  - Compute
    - Overview
    - Instances
    - Images
    - Key Pairs
    - Server Groups
  - Volumes
  - Network
  - Orchestration
  - Object Store
- Identity

Project / Compute / Instances

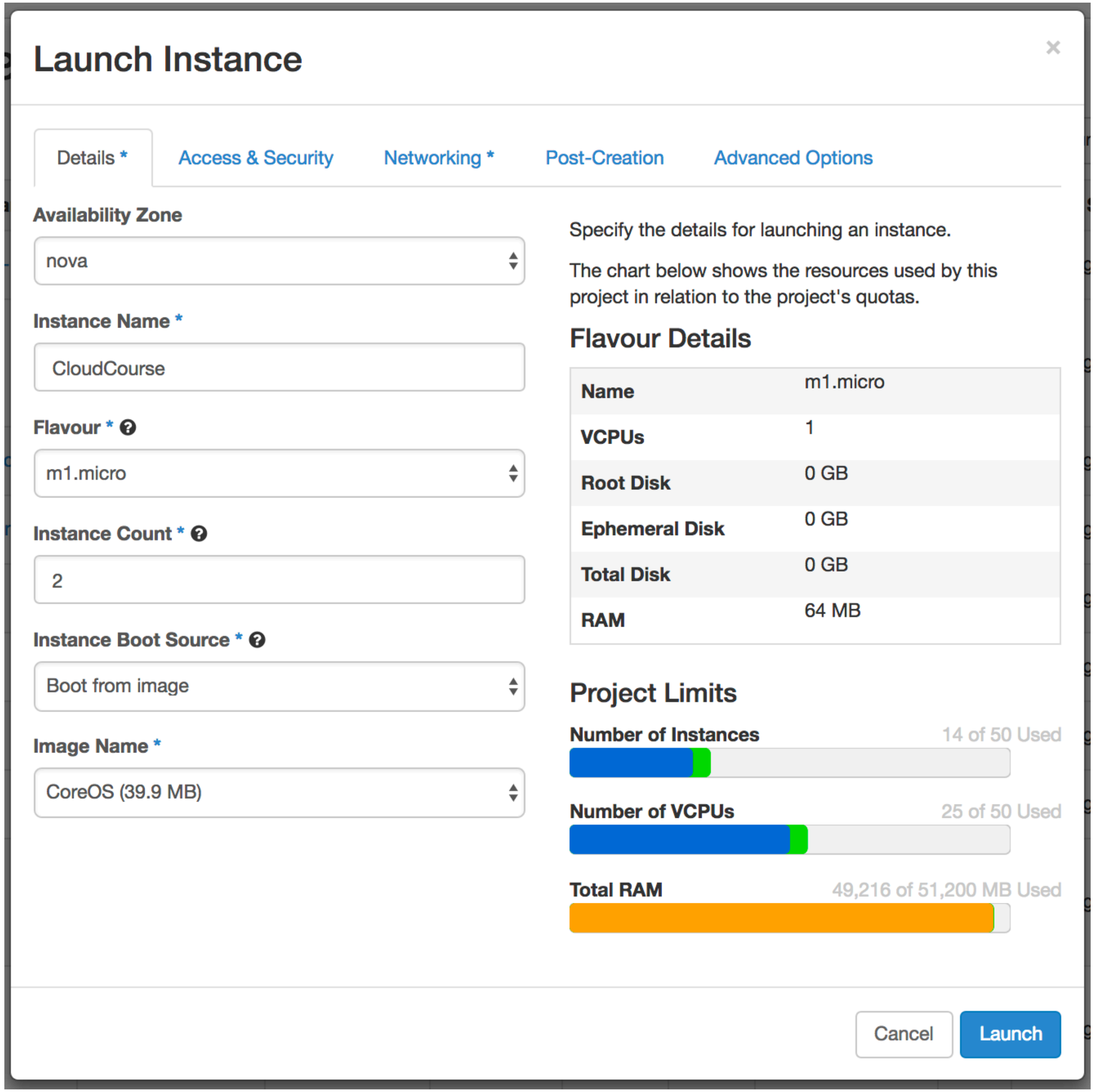
Instances

Instance ID = ▾ | Filter | Launch Instance | Delete Instances | More Actions ▾

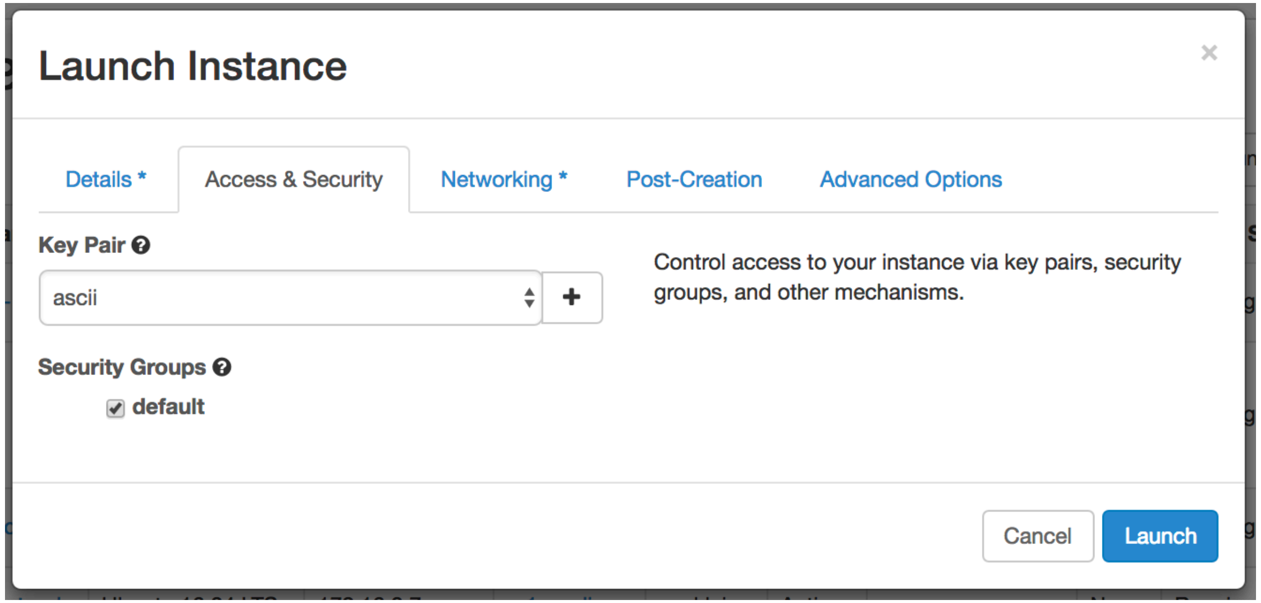
Displaying 13 items

Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
gitlab-runner	-	172.16.0.16	c4m8	gitlabxerces	Active	nova	None	Running	1 week, 3 days	Create Snapshot
gitlab	-	172.16.0.14 **Floating IPs:** 129.192.70.164	c4m16	gitlabxerces	Active	nova	None	Running	1 week, 3 days	Create Snapshot
ejoheke-sandbox-k8s-master-1	ubuntu-16.04	10.0.0.14 **Floating IPs:** 129.192.68.152	c2m4	kubernetes-ejoheke-sandbox	Active	nova	None	Running	1 month, 1 week	Create Snapshot
ejoheke-sandbox-k8s-node-2	ubuntu-16.04	10.0.0.19 **Floating IPs:** 129.192.70.79	c2m4	kubernetes-ejoheke-sandbox	Active	nova	None	Running	1 month, 1 week	Create Snapshot
ejoheke-sandbox-k8s-node-1	ubuntu-16.04	10.0.0.12 **Floating IPs:** 129.192.70.84	c2m4	kubernetes-ejoheke-sandbox	Active	nova	None	Running	1 month, 1 week	Create Snapshot
ejoheke-sandbox-k8s-node-nf-2	ubuntu-16.04	10.0.0.4	c2m4	kubernetes-ejoheke-sandbox	Active	nova	None	Running	1 month, 1 week	Create Snapshot
ejoheke-sandbox-k8s-node-nf-1	ubuntu-16.04	10.0.0.16	c2m4	kubernetes-ejoheke-sandbox	Active	nova	None	Running	1 month, 1 week	Create Snapshot
Jakob	-	10.0.0.17 **Floating IPs:** 129.192.71.35	c3m4	pubkey	Active	nova	None	Running	7 months	Create Snapshot
kubecluster-certbot	-	10.0.0.15	c1m05	ascii	Active	nova	None	Running	7 months, 1 week	Create Snapshot
kube-cluster-server-03	ubuntu-16.04	10.0.0.10 **Floating IPs:** 129.192.70.237	c4m16	ascii	Active	nova	None	Running	11 months	Create Snapshot

DEMO

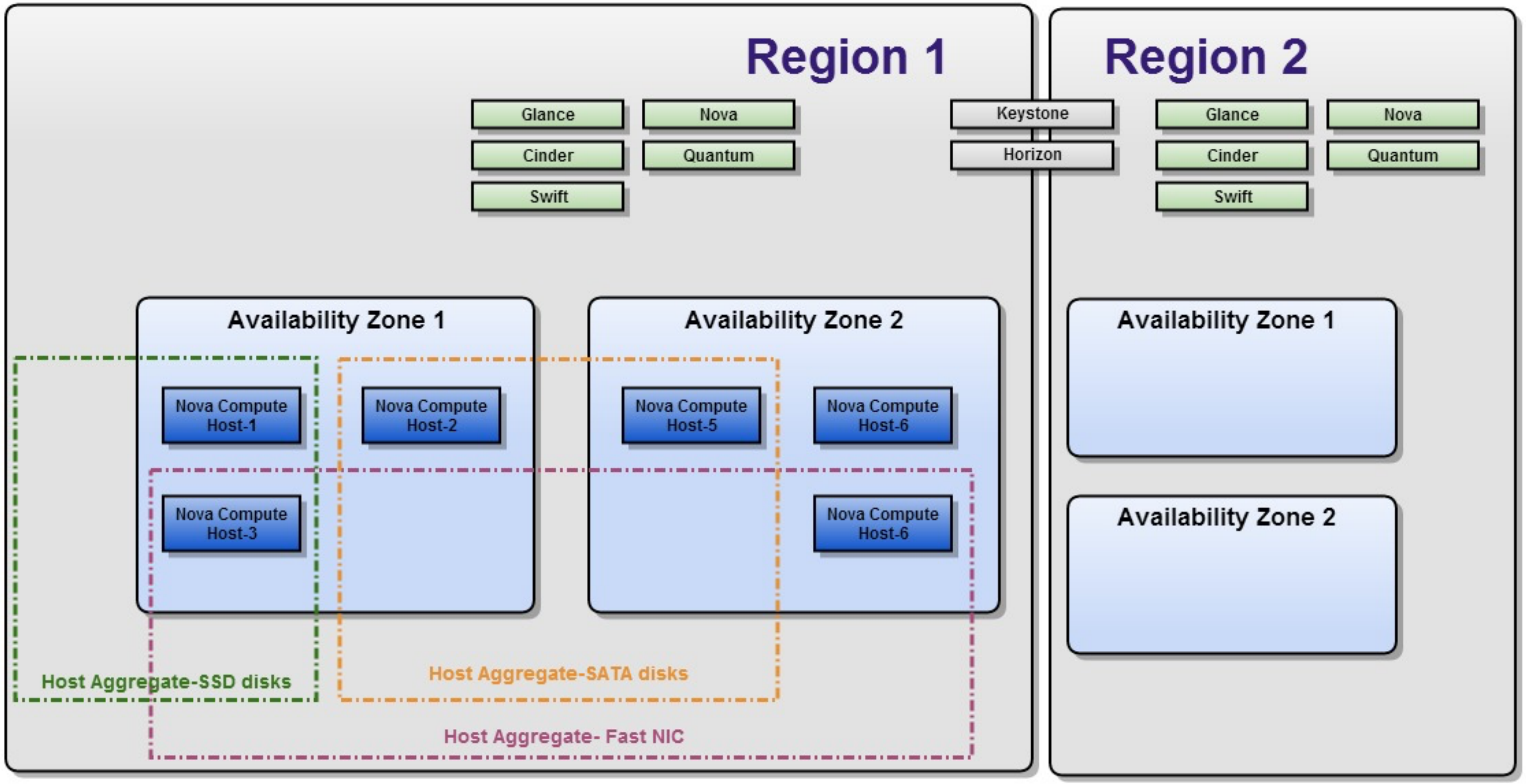

Launch Instance
Details *
Access & Security
Networking *
Post-Creation
Advanced Options
Availability Zone
nova
Instance Name *
CloudCourse
Flavour *
m1.micro
Instance Count *
2
Instance Boot Source *
Boot from image
Image Name *
CoreOS (39.9 MB)
Specify the details for launching an instance.
The chart below shows the resources used by this project in relation to the project's quotas.
Flavour Details
Name m1.micro
VCPUs 1
Root Disk 0 GB
Ephemeral Disk 0 GB
Total Disk 0 GB
RAM 64 MB
Project Limits
Number of Instances 14 of 50 Used
Number of VCPUs 25 of 50 Used
Total RAM 49,216 of 51,200 MB Used
Cancel
Launch

DEMO


Launch Instance
×
Details *
Access & Security
Networking *
Post-Creation
Advanced Options
Key Pair
ascii
+
Control access to your instance via key pairs, security groups, and other mechanisms.
Security Groups
default
Cancel
Launch

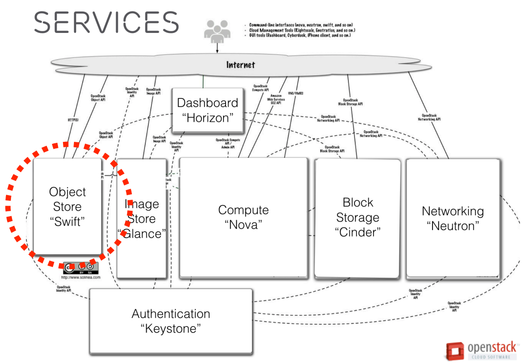
DEMO

AVAILABILITY ZONEs



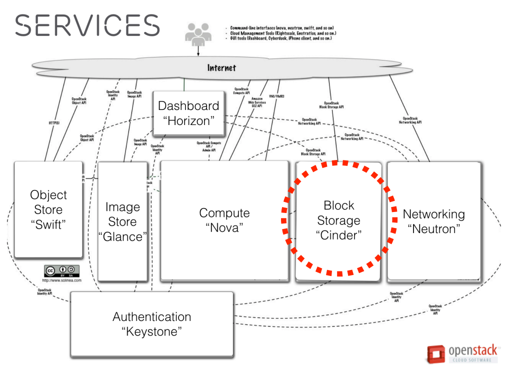
Specify AZ on VM creation

Object Store: “Swift”



- Swift is a highly available, distributed, eventually consistent object/blob store
- <key> & <object>
  - Unstructured data store. Swift simply stores bits. (not a database, not block-device)
  - Swift stores blobs of data.
- Organised in containers
- Swift provides a REST API over HTTP
  - A swift storage URL looks like
  - swift.example.com/v1/account/container/object

Block Storage: "Cinder"



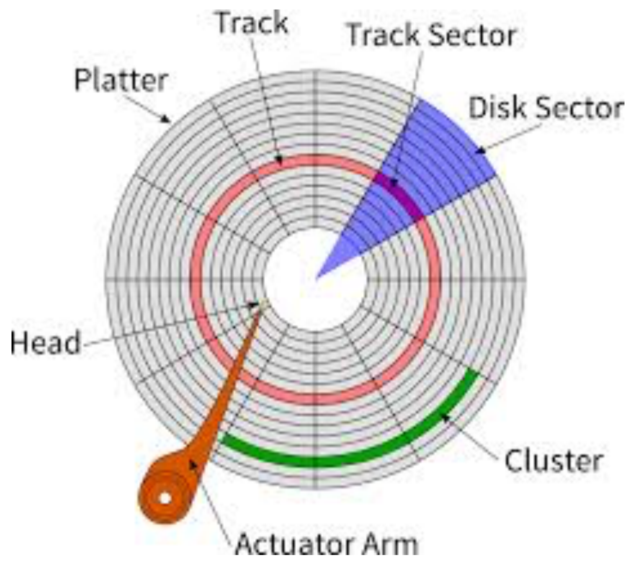
Persistent block storage for VMs

NB: There is nothing persistent with your regular VMs…

Volumes (virtual raw block devices)

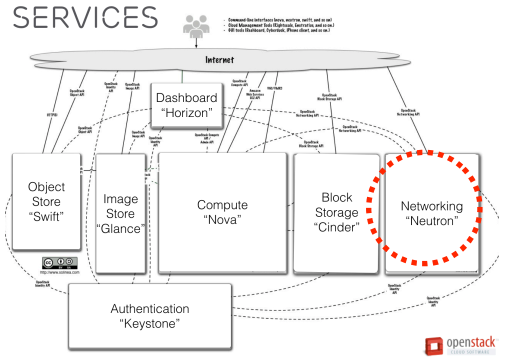
Be mounted inside a VM:

- Will show up as /dev/hhx
- Treat like a regular drive; mount, partition, format



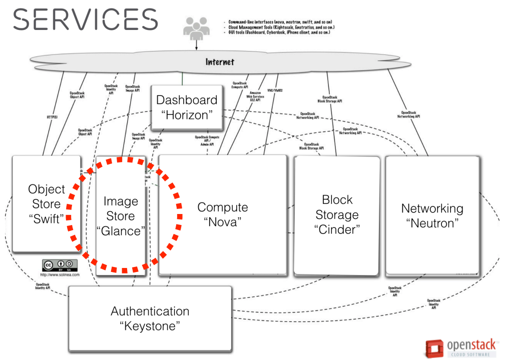
openstack CLOUD SOFTWARE

Networking: “Neutron”



- Networking as a service
- Manages IP addresses, (static/dynamic/floating)
- Users can create their own networks, control traffic, and connect servers and devices
- Configure firewalls

Image Store: “Glance”



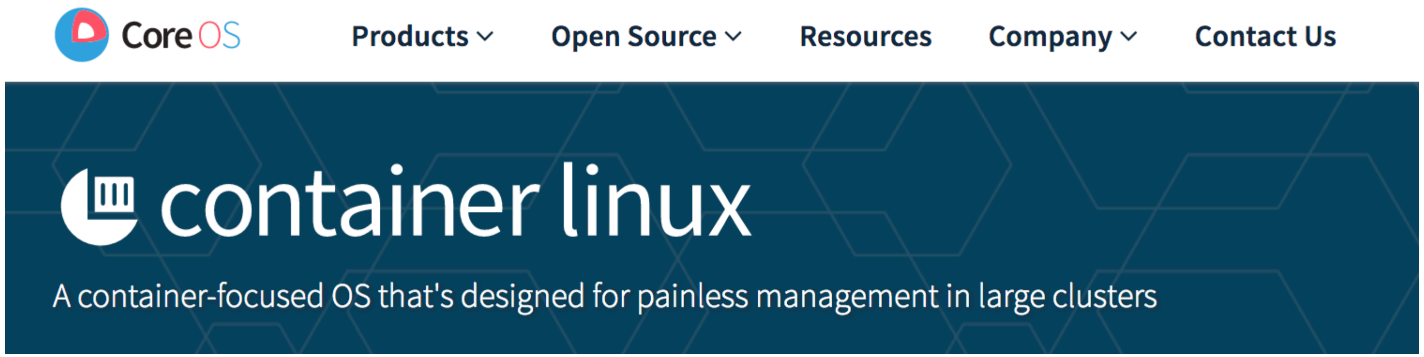
Catalog of VM boot images

Easy to upload your own favourite OS

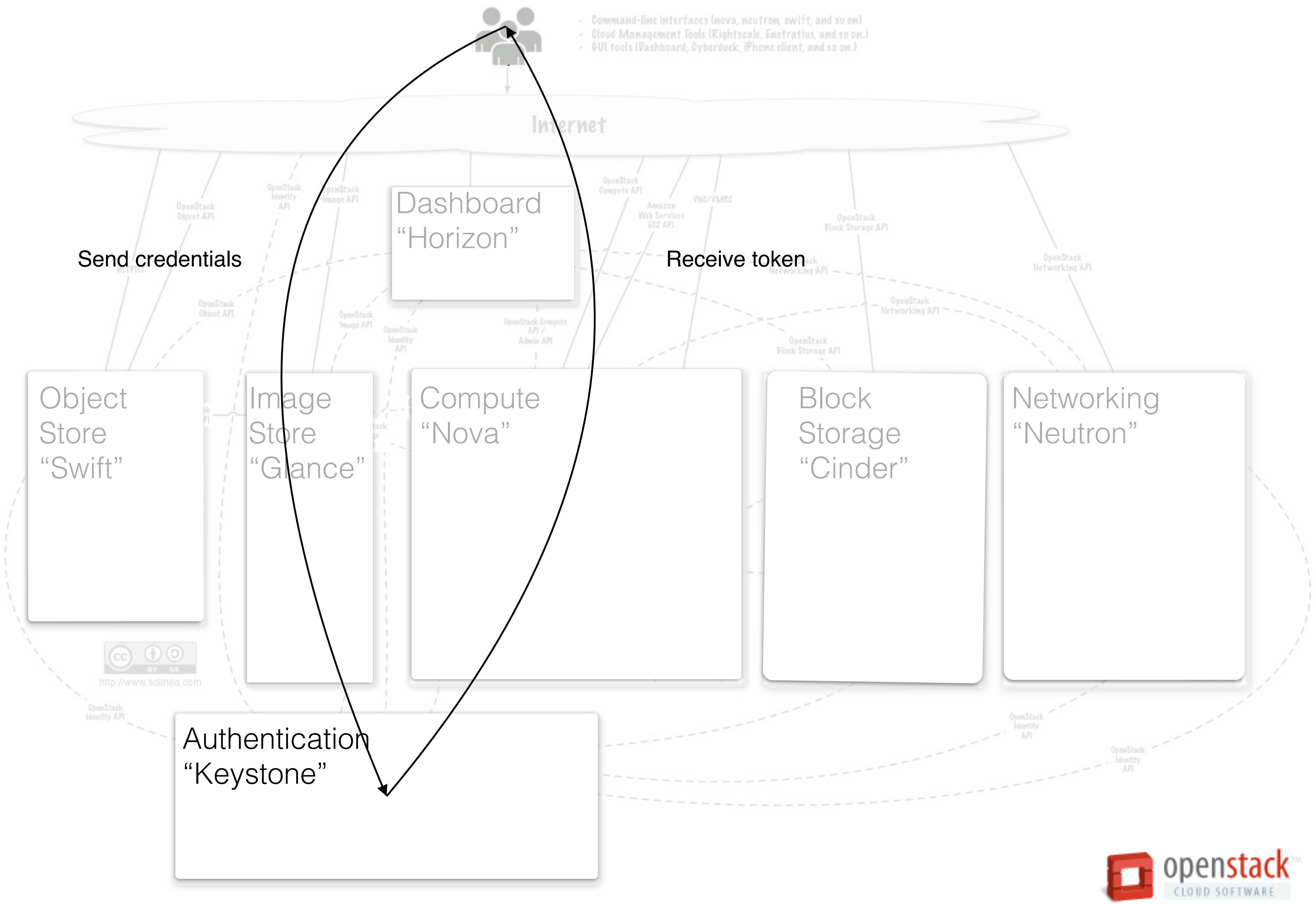
Save a running VM and use it as an image

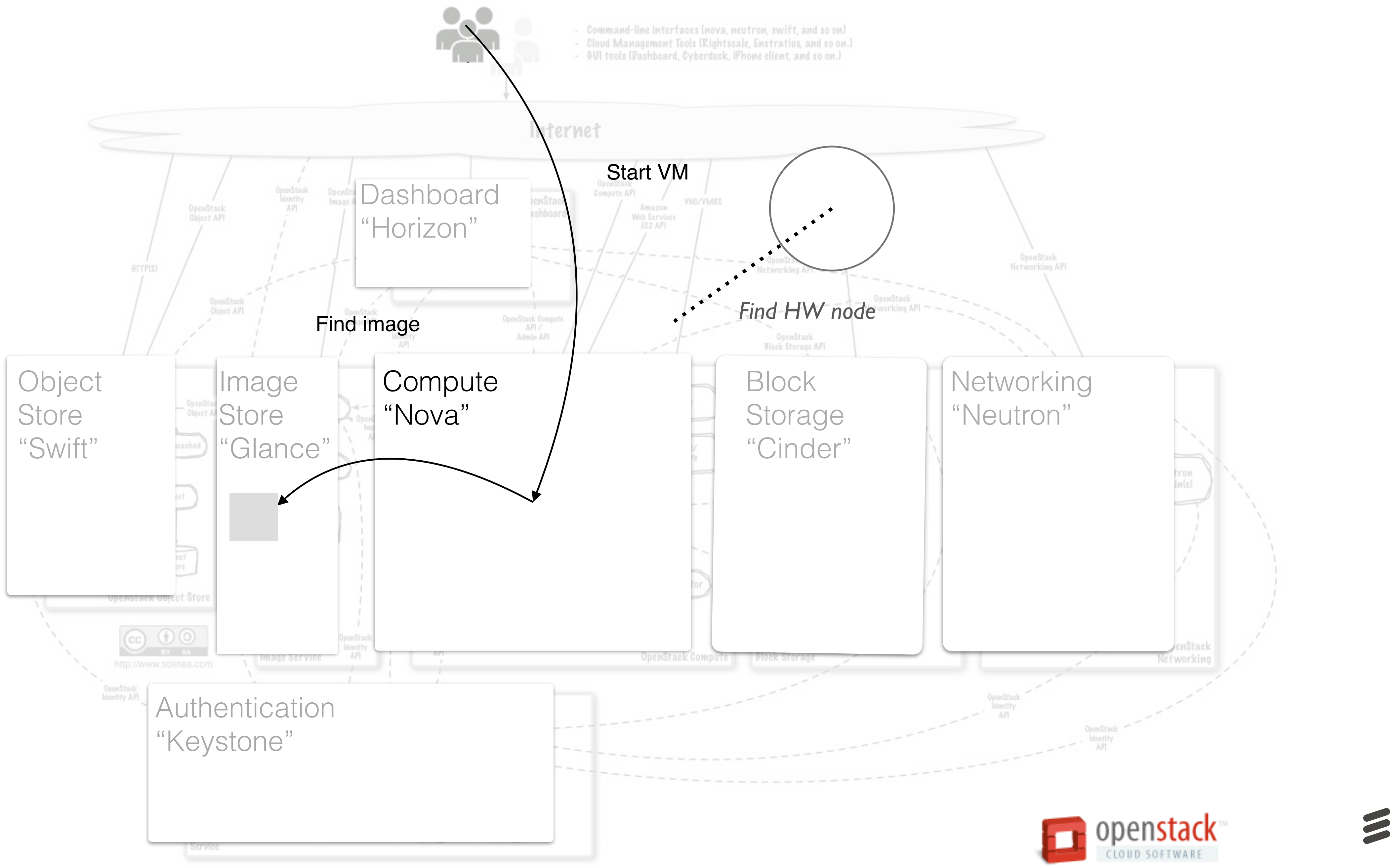
Import & export

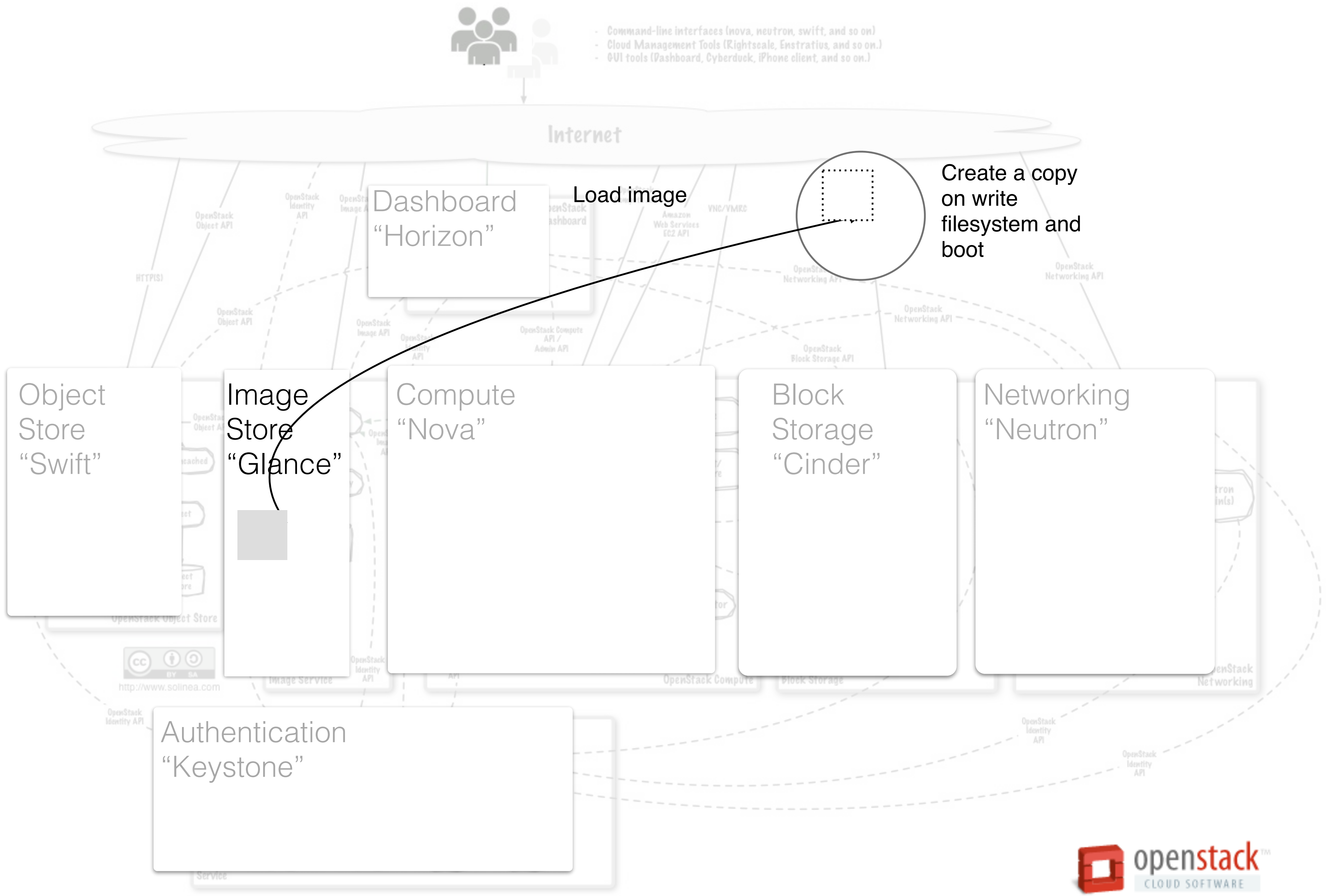
Note: do not use qcow

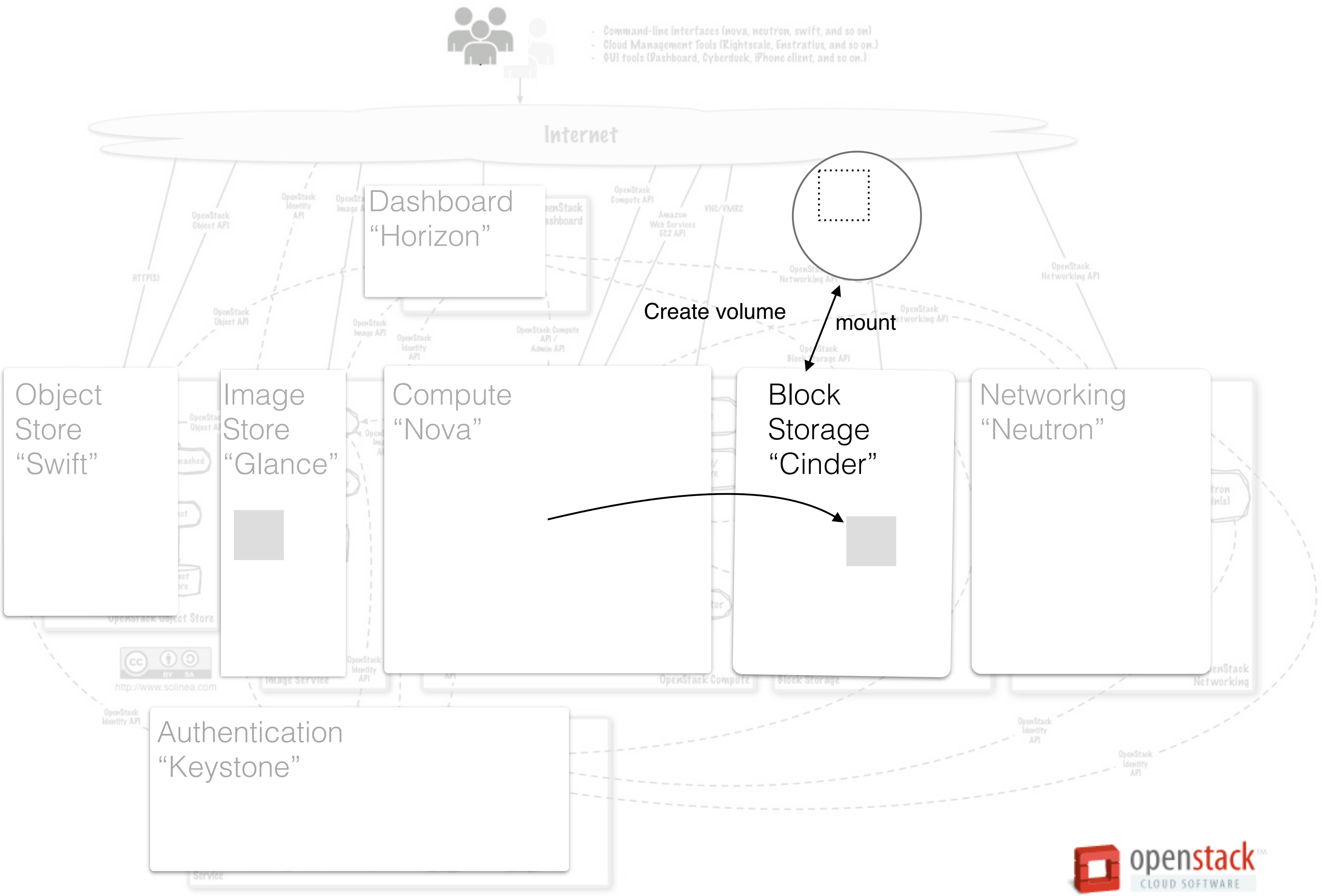


will reboot and update automatically… (for good and bad)

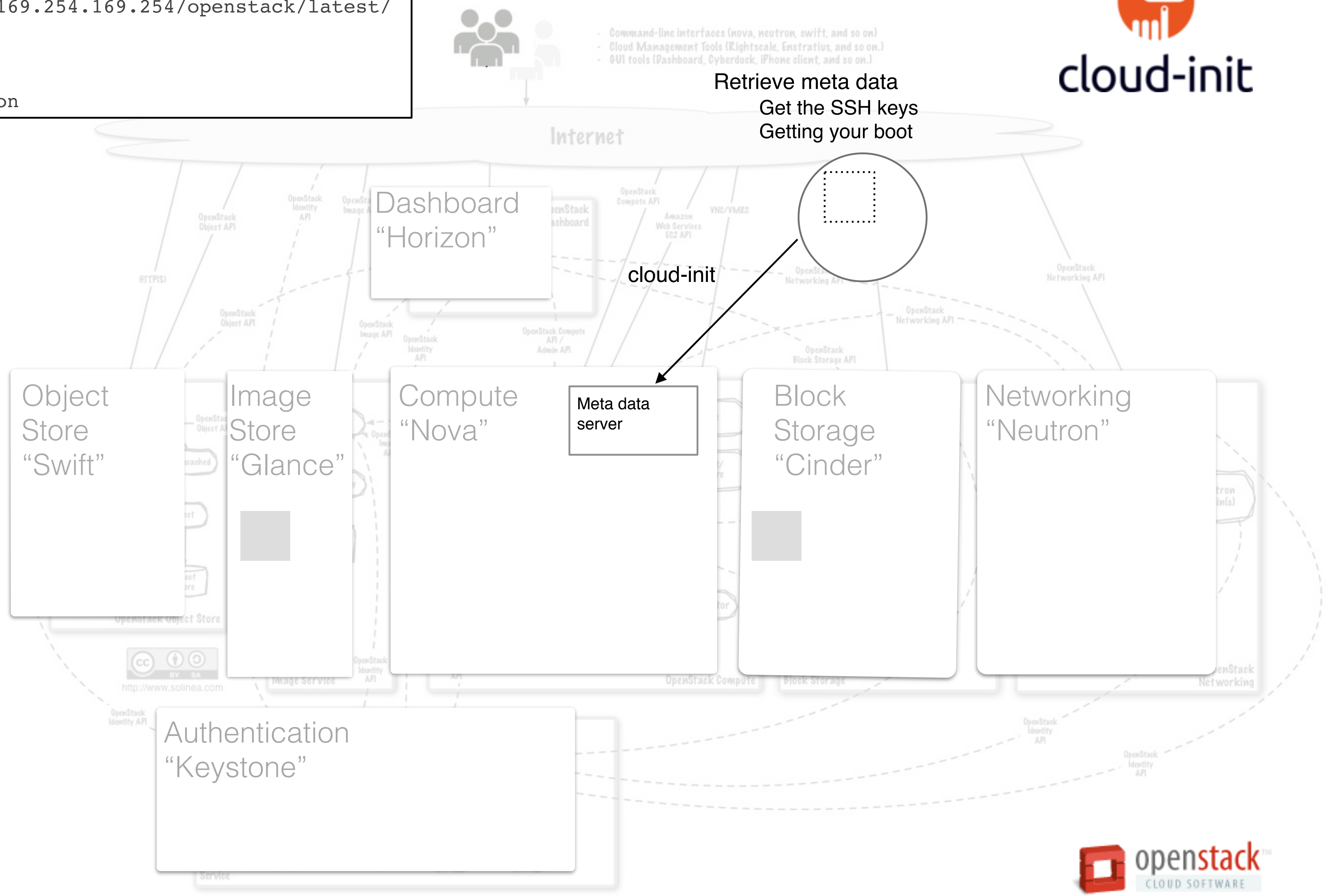

Send credentials
Receive token
Dashboard
"Horizon"
Object
Store
"Swift"
Image
Store
"Glance"
Compute
"Nova"
Block
Storage
"Cinder"
Networking
"Neutron"
Authentication
"Keystone"
Internet
openstack
CLOUD SOFTWARE

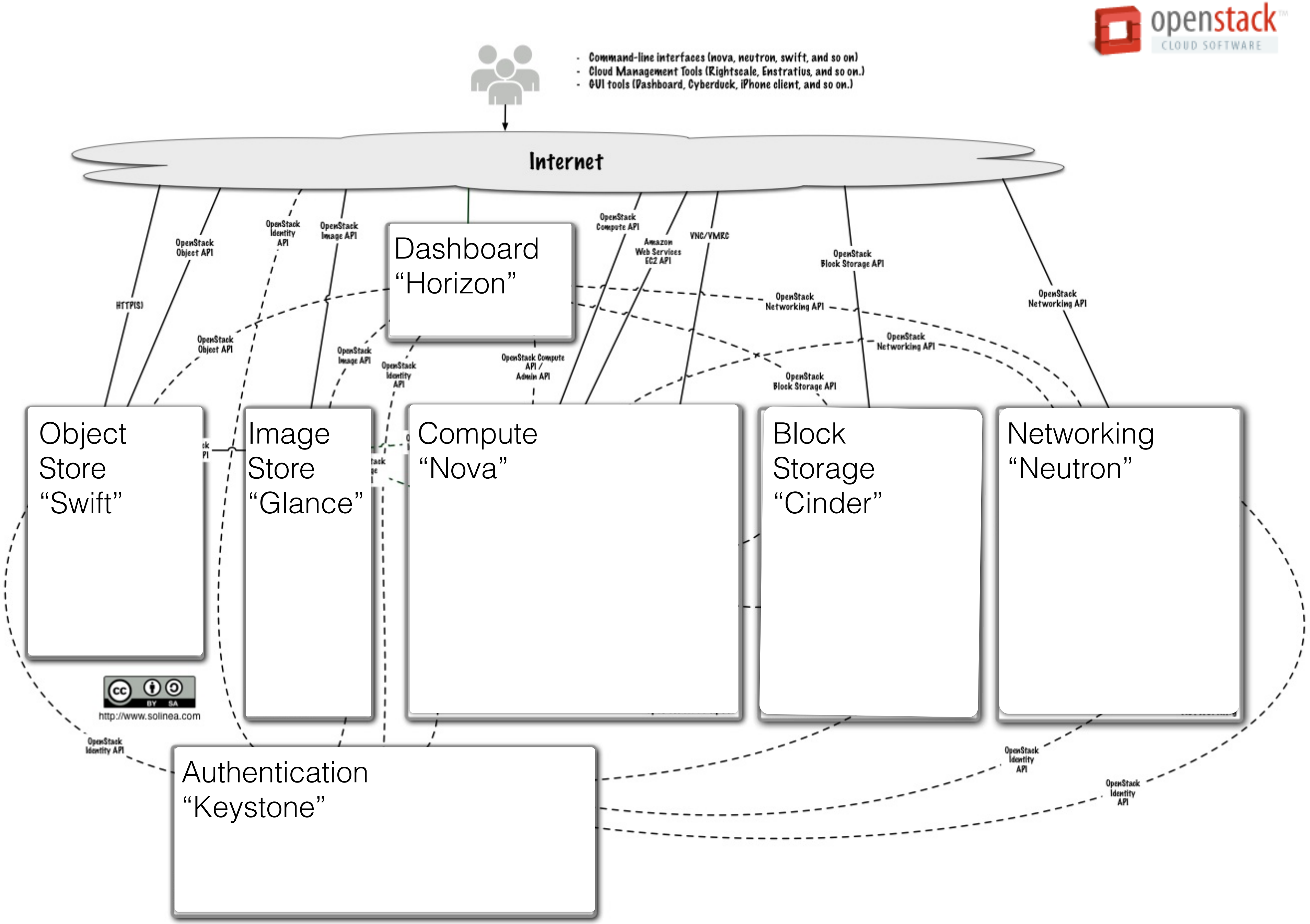

Start VM
Dashboard
"Horizon"
Find image
Find HW node
Object
Store
"Swift"
Image
Store
"Glance"
Compute
"Nova"
Block
Storage
"Cinder"
Networking
"Neutron"
Authentication
"Keystone"
openstack
CLOUD SOFTWARE

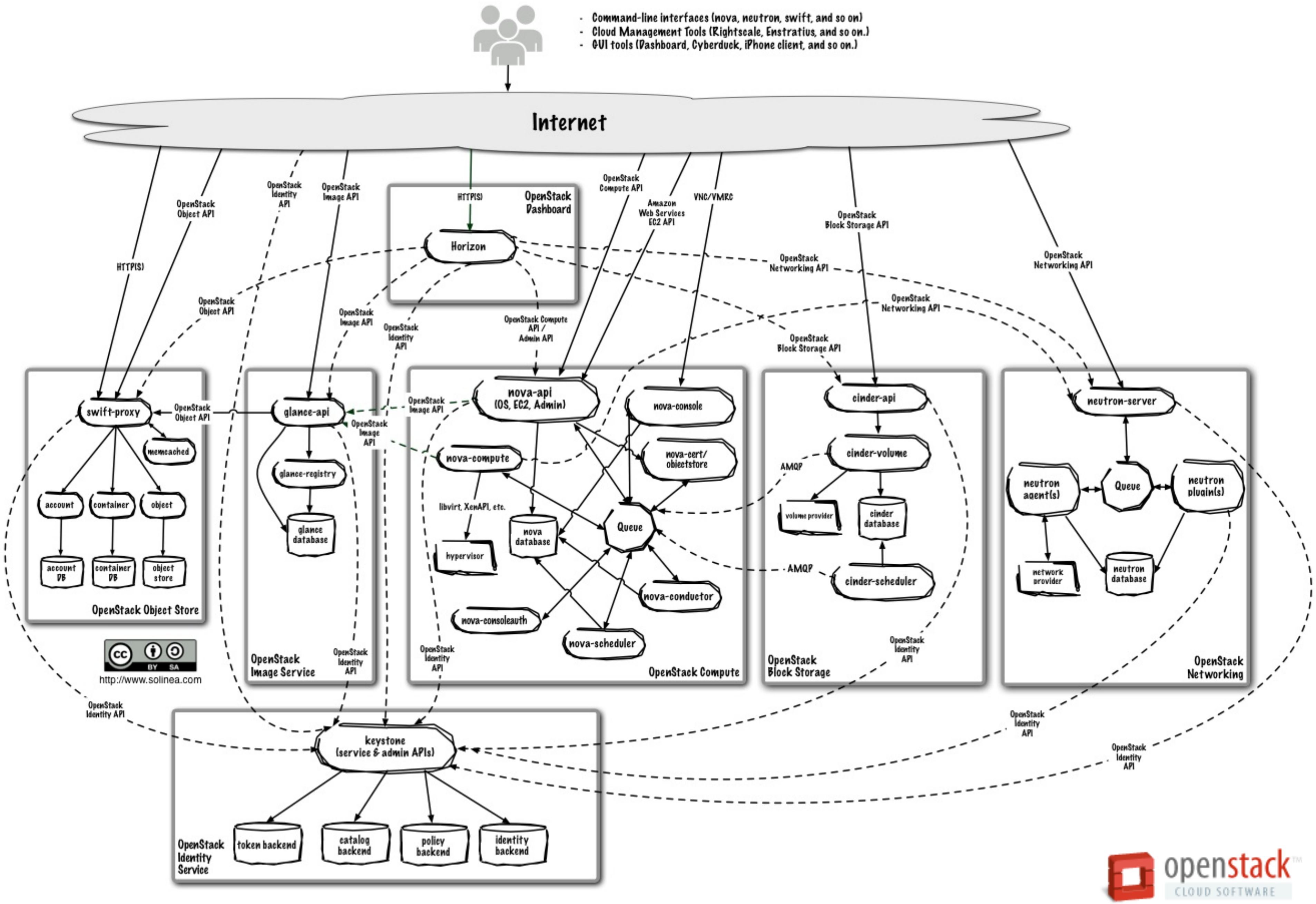

Internet
Dashboard
“Horizon”
Load image
Create a copy
on write
filesystem and
boot
Object
Store
“Swift”
Image
Store
“Glance”
Compute
“Nova”
Block
Storage
“Cinder”
Networking
“Neutron”
Authentication
“Keystone”
openstack
CLOUD SOFTWARE


Internet
Dashboard
"Horizon"
Create volume
mount
Object
Store
"Swift"
Image
Store
"Glance"
Compute
"Nova"
Block
Storage
"Cinder"
Networking
"Neutron"
Authentication
"Keystone"

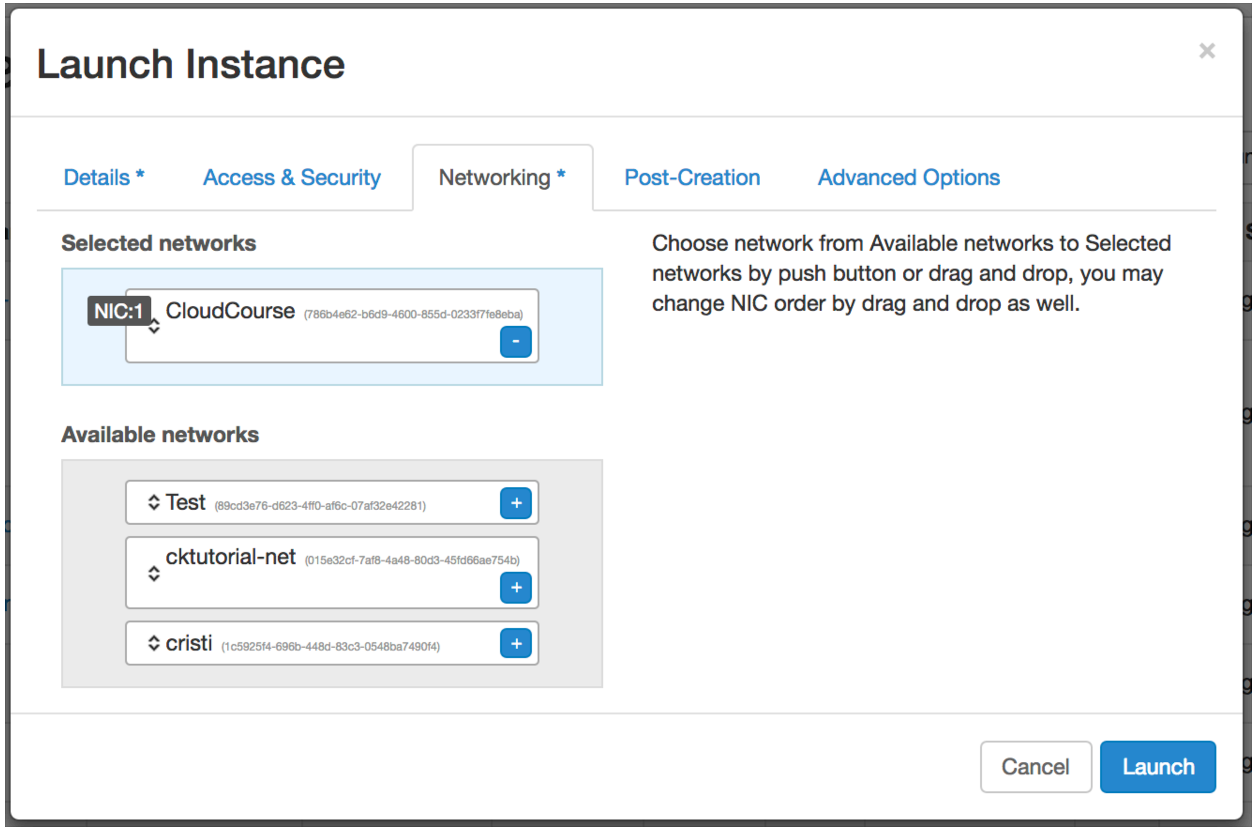
```
# curl http://169.254.169.254/openstack/latest/
meta_data.json
user_data
password
vendor_data.json
```




openstack
CLOUD SOFTWARE
- Command-line interfaces (nova, neutron, swift, and so on)
- Cloud Management Tools (Rightscale, Enstratius, and so on.)
- GUI tools (Dashboard, Cyberduck, iPhone client, and so on.)
Internet
HTTP(S)
OpenStack Object API
OpenStack Identity API
OpenStack Image API
Dashboard
"Horizon"
OpenStack Compute API
Amazon Web Services EC2 API
VNC/VMRC
OpenStack Block Storage API
OpenStack Networking API
OpenStack Networking API
OpenStack Networking API
OpenStack Object API
OpenStack Image API
OpenStack Identity API
OpenStack Compute API / Admin API
OpenStack Block Storage API
Object
Store
"Swift"
Image
Store
"Glance"
Compute
"Nova"
Block
Storage
"Cinder"
Networking
"Neutron"
http://www.solinea.com
OpenStack Identity API
OpenStack Identity API
OpenStack Identity API
Authentication
"Keystone"



- Command-line interfaces (nova, neutron, swift, and so on)
- Cloud Management Tools (Rightscale, Enstratius, and so on.)
- GUI tools (Dashboard, Cyberduck, iPhone client, and so on.)
Internet
HTTP(S)
OpenStack Object API
OpenStack Identity API
OpenStack Image API
OpenStack Dashboard
Horizon
OpenStack Compute API
Amazon Web Services EC2 API
VNC/VMRC
OpenStack Block Storage API
OpenStack Networking API
OpenStack Compute API / Admin API
swift-proxy
memcached
account
container
object
account DB
container DB
object store
OpenStack Object Store
glance-api
glance-registry
glance database
OpenStack Image Service
nova-api (OS, EC2, Admin)
nova-console
nova-cert/ objectstore
nova-compute
libvirt, XenAPI, etc.
hypervisor
nova database
Queue
nova-conductor
nova-consoleauth
nova-scheduler
OpenStack Compute
cinder-api
cinder-volume
AMQP
volume provider
cinder database
cinder-scheduler
OpenStack Block Storage
neutron-server
neutron agent(s)
neutron plugin(s)
network provider
neutron database
OpenStack Networking
http://www.solinea.com
keystone (service & admin APIs)
token backend
catalog backend
policy backend
identity backend
OpenStack Identity Service
openstack
CLOUD SOFTWARE


Launch Instance
Details *
Access & Security
Networking *
Post-Creation
Advanced Options
Selected networks
NIC:1
CloudCourse
Available networks
Test
cktutorial-net
cristi
Choose network from Available networks to Selected networks by push button or drag and drop, you may change NIC order by drag and drop as well.
Cancel
Launch

DEMO

Openstack CLI

```
$ . Datahub-openrc.sh

$ openstack server list

$ openstack network list

$ openstack help

$ openstack server help
```

Openstack CLI

```
$ openstack server create --image "Ubuntu 18.04" --network "internet" --flavor "c1m1" --key-name
"ascii" cli-test
-----------------------------+-------------------------------------------------------------------+
| Field                      | Value                                                             |
+----------------------------+-------------------------------------------------------------------+
| created                    | 2019-08-21T12:12:46Z                                              |
| flavor                     | c1m1 (dbd3b206-0783-4243-b6fe-1e43d765d633)                       |
| hostId                     |                                                                   |
| id                         | 2bdeebae-5c4a-4d35-b42e-fec06c196351                              |
| image                      | Ubuntu 18.04 (18a5fc04-39b0-49b8-ac52-3a572ed1d5c3)               |
| key_name                   | ascii                                                             |
| name                       | cli-test                                                          |
| progress                   | 0                                                                 |
| project_id                 | 9147fbfe224c4bd4864eec589413c095                                  |
| properties                 |                                                                   |
| security_groups            | name='default'                                                    |
| status                     | BUILD                                                             |
| updated                    | 2019-08-21T12:12:46Z                                              |
| user_id                    | 205a3ad1266927448414a68de327c12f35e38bf740d05d4e76353d0d45af4b96 |
| volumes_attached           |                                                                   |
+----------------------------+-------------------------------------------------------------------+
```

OPENSTACK CLI ClientS

- Python clients
- Tip: use VirtualEnv

```
# pip install python-PROJECTclient
```

- `barbican` - Key Manager Service API
- `ceilometer` - Telemetry API
- `cinder` - Block Storage API and extensions
- `cloudkitty` - Rating service API
- `designate` - DNS service API
- `fuel` - Deployment service API
- `glance` - Image service API
- `gnocchi` - Telemetry API v3
- `heat` - Orchestration API
- `magnum` - Container Infrastructure Management service API
- `manila` - Shared file systems API
- `mistral` - Workflow service API
- `monasca` - Monitoring API
- `murano` - Application catalog API
- `neutron` - Networking API
- `nova` - Compute API and extensions
- `senlin` - Clustering service API
- `swift` - Object Storage API
- `trove` - Database service API

openstack CLOUD SOFTWARE

Python SDK

```
from credentials import get_session
from novaclient.client import Client

session = get_session()
nova_client = Client("2.1", session=session)

worker_name = "dummy"
image = nova_client.images.find(name="Ubuntu 18.04")
flavor = nova_client.flavors.find(name="c1m1")
net = nova_client.networks.find(label="internet")
nics = [{'net-id': net.id}]
instance = nova_client.servers.create(name=worker_name, image=image, flavor=flavor,
key_name="ascii", nics=nics)
```

REST from the commandline

```
$ curl -X POST http://128.136.179.2:5000/v2.0/tokens \
  -H "Content-Type: application/json" \
  -d '{"auth": {"tenantName": "'"$OS_TENANT_NAME"'", \
   ”passwordCredentials": {"username": “'"$OS_USERNAME"'", \
   "password": "'"$OS_PASSWORD"'"}}}' \
   | python -m json.tool
```

```
$ curl -H "X-Auth-Token: 558f82170b9b46a8a088019774d382d1" \
  http://94.246.116.200:5000/v2.0/tenants\
  | python -m json.tool
```

But cloud is more than just a cost saving business model.

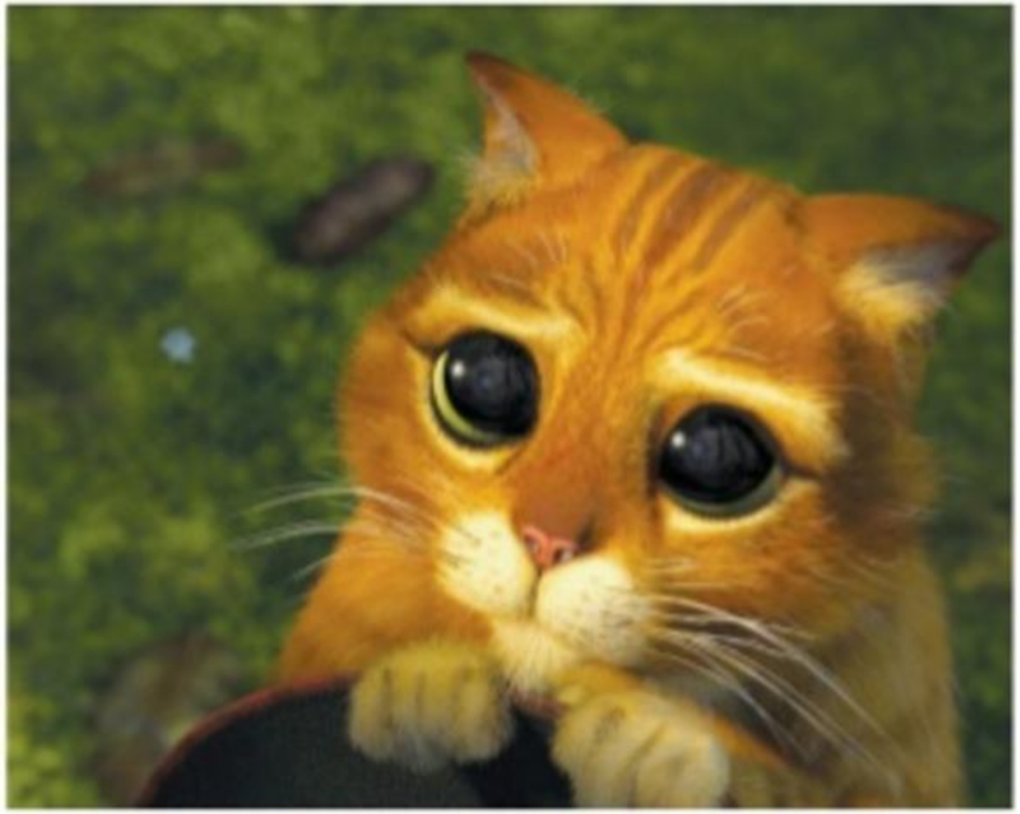
It's a new way of operating and designing services.

Automation

Leverage the fact that the infrastructure
is virtual and thus can be programmed.

Infrastructure-as-code

Pets vs Cattle



bob-the-mailserver
sperry-the-fileserver

cluster-server-151
cluster-server-152

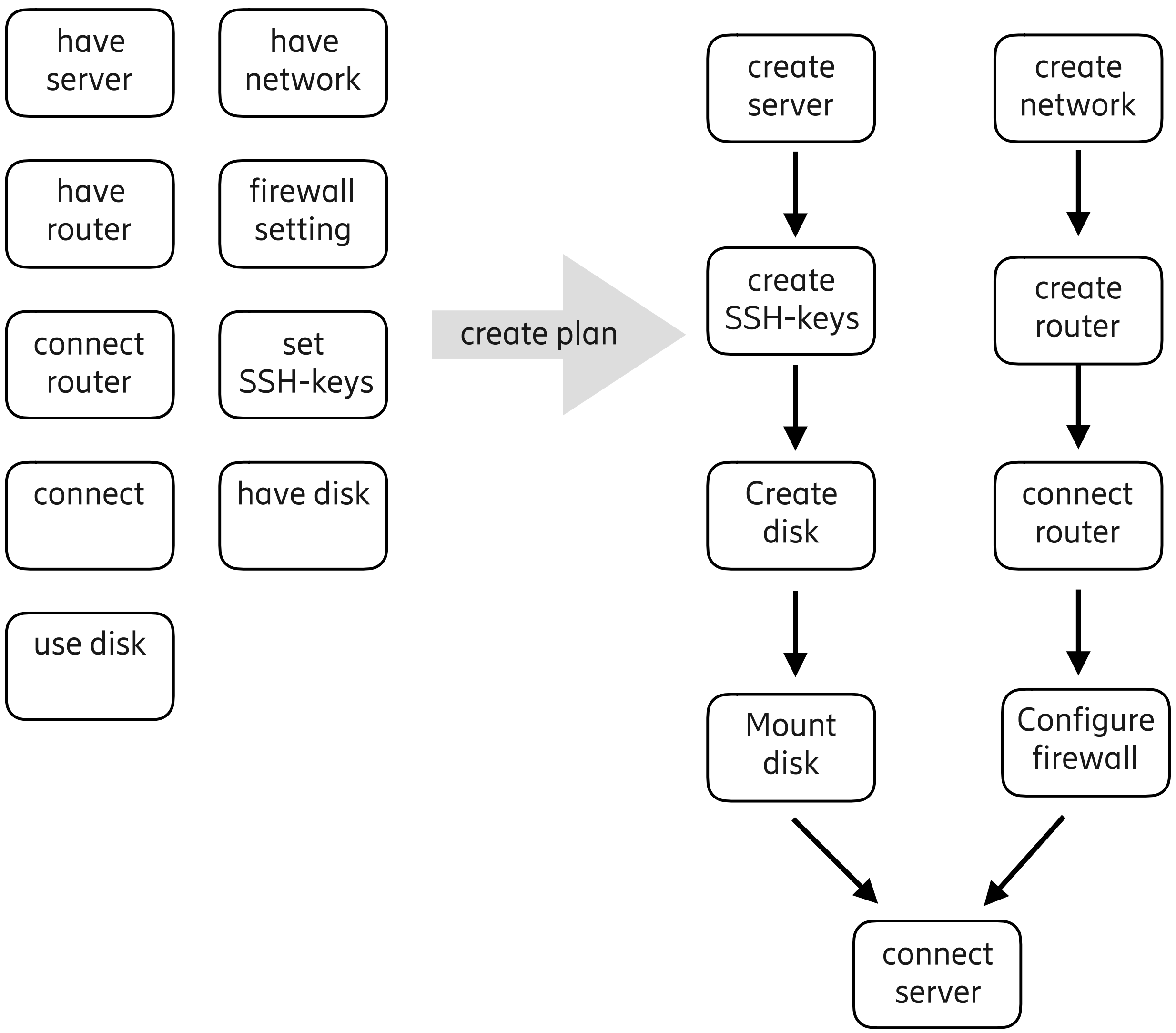
Infrastructure-as-Code

- **Orchestration** is the creation of virtual resources and connecting them together (sometimes called provisioning).
- **Configuration** is the process of installing software on the orchestrated nodes and set parameters, etc.
- Many different tools: Many different tools: Chef, Puppet, Salt, CloudFormation, Heat, Terraform, Ansible, etc.
- We will use Terraform for orchestration & Ansible for configuration.

Terraform

HashiCorp Terraform

- Declarative
- State you desired state, i.e. what servers should be running and how they should be connected
- Terraform then calculates a plan (a DAG) and applies the actions



DEMO

Terraform

Syntax

```
<BLOCK TYPE> "<BLOCK LABEL>" "<BLOCK LABEL>" {
  <IDENTIFIER> = <EXPRESSION>
}

variable "availability_zones" {
  description = "A list of availability zones"
  type = list(string)
}

resource "aws_vpc" "main" {
  cidr_block = "${var.base_cidr_block}"
}
```

File

Save as Terraform configurations in files "*.tf"

Command line

```
$ terraform
```

an example network

```
resource "openstack_networking_network_v2" "network" {
  name = "simple-network"
  admin_state_up = "true"
}

resource "openstack_networking_subnet_v2" "subnet" {
  name = "simple-subnet"
  network_id = "${openstack_networking_network_v2.network.id}"
  cidr = "192.168.1.0/24"
}

resource "openstack_networking_router_v2" "router" {
  name = "simple-router"
  admin_state_up = "true"
  external_network_id = "${var.external_network_id}"
}

resource "openstack_networking_router_interface_v2" "router_iface" {
  router_id = "${openstack_networking_router_v2.router.id}"
  subnet_id = "${openstack_networking_subnet_v2.subnet.id}"
}
```

Terraform

an example instance

```
resource "openstack_compute_keypair_v2" "ssh_keypair" {
  name       = "simple-keypair"
  public_key = "${chomp(file(var.public_key_path))}"
}

resource "openstack_compute_instance_v2" "server" {
  name       = "simple-server"
  image_name = "${var.image}"
  flavor_id  = "${var.flavor}"
  key_pair   = "${openstack_compute_keypair_v2.ssh_keypair.name}"

  network {
    name = "simple-network"
  }

  resource "openstack_networking_floatingip_v2" "floatingip" {
    pool       = "${var.floatingip_pool}"
  }

  security_groups = [
    "${openstack_networking_secgroup_v2.security_group.name}",
    "default",
  ]
}
```

Terraform

an example security group

```
resource "openstack_networking_secgroup_v2" "security_group" {
  name        = "simple-secgroup"
  description = "Rules for the simple project"
}

resource "openstack_networking_secgroup_rule_v2" "allow_ssh" {
  direction = "ingress"
  ethertype = "IPv4"
  protocol = "tcp"
  port_range_min = "22"
  port_range_max = "22"
  remote_ip_prefix = "0.0.0.0/0"
  security_group_id = "${openstack_networking_secgroup_v2.security_group.id}"
}
```

Ansible

- Playbooks are the basis in Ansible for configuration management and multi-machine deployment system
- Each playbook is composed of one or more 'plays' in a list.
  - The goal of a play is to map a group of hosts to some well-defined roles., represented by things ansible calls tasks. At a basic level, a task is nothing more than a call to an ansible module.
- By composing a playbook of multiple 'plays', it is possible to orchestrate multi-machine deployments, running certain steps on all machines in the webservers group (role), then certain steps on the database server group, then more commands back on the webservers group, etc.
- "plays" are more or less a sports analogy. You can have quite a lot of plays that affect your systems to do different things. It's not as if you were just defining one particular state or model, and you can run different plays at different times.

```
ansible all --private-key=mykey.key --inventory-file hosts.ini -m ping -u ubuntu
```

Ansible

A simple playbook

```yaml
---
- hosts: webservers
  remote_user: root

  tasks:
  - name: ensure apache is at the latest version
    yum:
      name: httpd
      state: latest
  - name: write the apache config file
    template:
      src: /srv/httpd.j2
      dest: /etc/httpd.conf

- hosts: databases
  remote_user: root

  tasks:
  - name: ensure postgresql is at the latest version
    yum:
      name: postgresql
      state: latest
  - name: ensure that postgresql is started
    service:
      name: postgresql
      state: started
```

Inventory

```
---
[webservers]
113.56.188.22
www1.ericsson.net

[databases]
136.241.4.34
db1.lu.se
```

```
ansible-playbook -i hosts site.yml
```

Ansible

templates, notifications and handlers

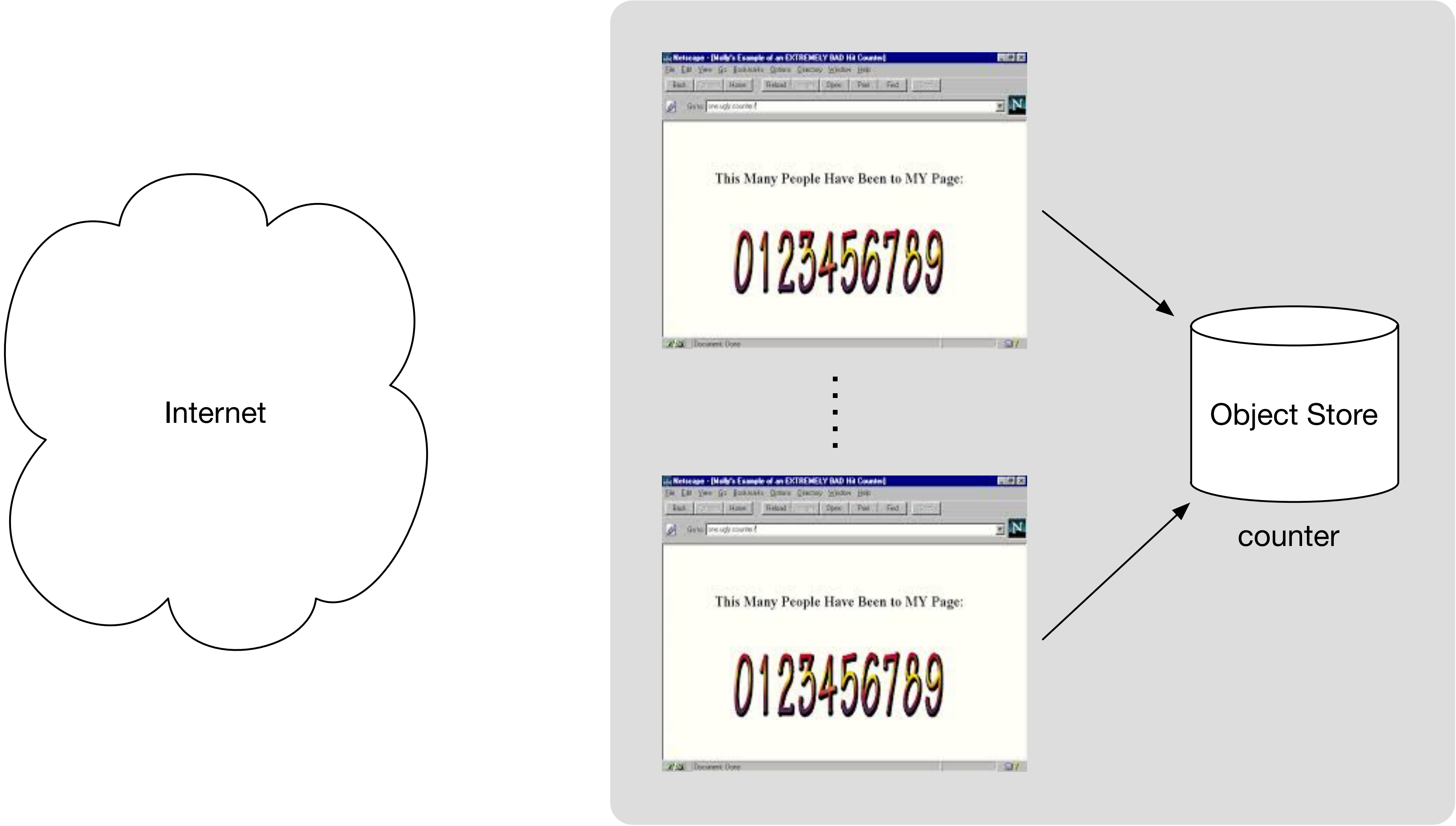
```yaml
name: template configuration file
 template:
   src: template.j2
   dest: /etc/foo.conf
 notify:
    - restart memcached
    - restart apache

 handlers:
   - name: restart memcached
     service:
       name: memcached
       state: restarted
   - name: restart apache
     service:
       name: apache
       state: restarted
```

```
ansible-playbook -i hosts site.yml
```

Assignment #1A

Implement a visitor counter service. Use Horizon and the CLI



Assignment #1A

Implement a visitor counter service. Use Horizon and the CLI

— Create virtual infrastructure by clicking in the GUI

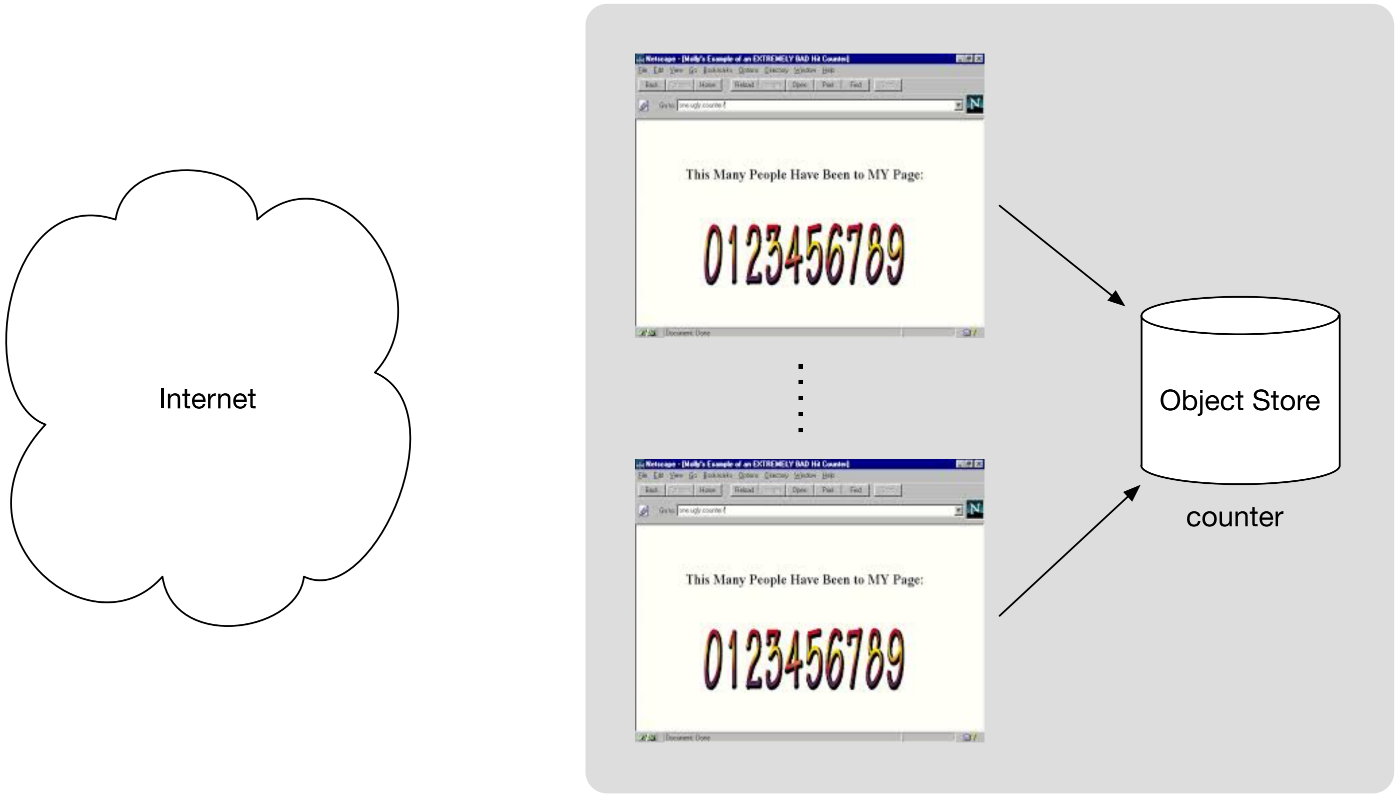
— Install the OpenStack command-line tools and learn how use (list servers, start servers, list objects in Swift, upload and download objects)

— Implement a very simple web server that reads and writes from persistent storage provided by OpenStack

— Show that it is possible to add and remove web servers and maintain a consistent behaviour for the visitor counter

Assignment #1B

Implement a visitor counter service. Use Terraform and maybe a bit of Ansible



- Same as in 1A, but using Terraform.
- You may reuse the VM server image from 1A or use Ansible for configuration.