CLOUD COMPUTING APPLICATIONS

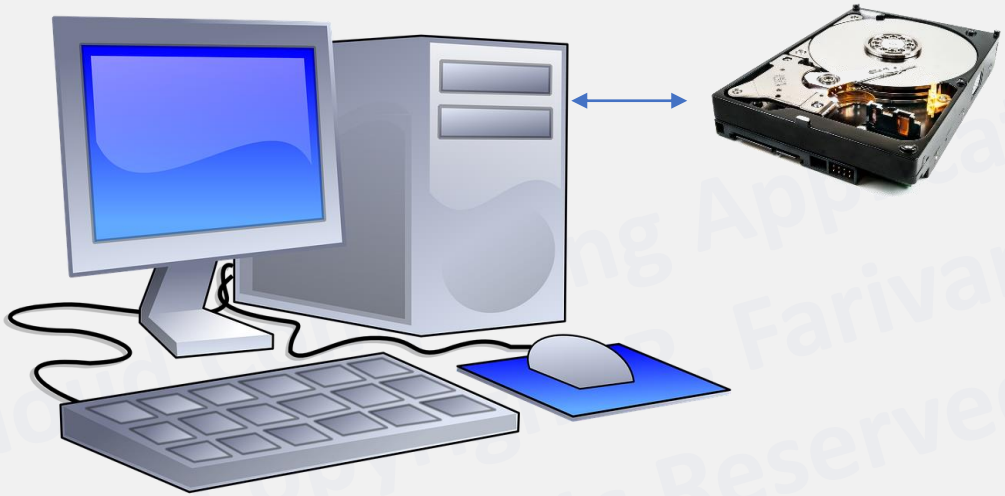
Cloud Storage: Basics

Prof. Reza Farivar

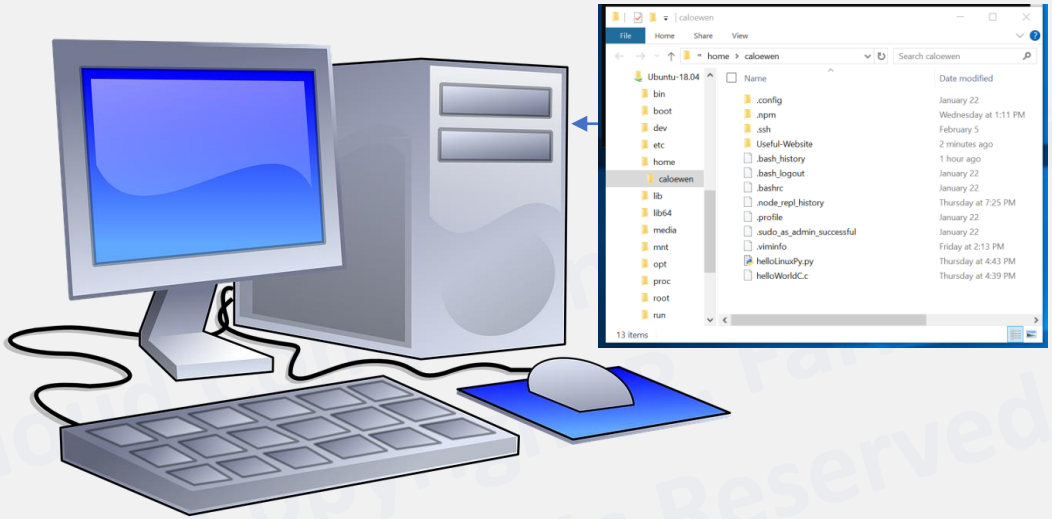
Cloud-based Storage

- Data Storage was one of the first use cases for Cloud Computing
- The competition is very fierce
- The cloud storage models have solidified into a few main categories
- Many different types of cloud storage, sometimes confusing
- We will cover the main categories in this lesson

Back to Basics: Block Storage



File System

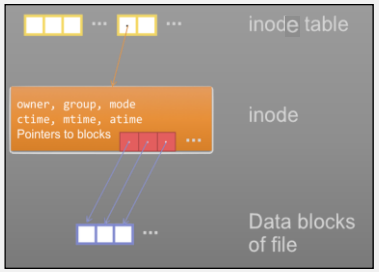


File System

- The middleware between the physical storage device and programs running on top of the OS
  - Block Storage
  - Typically three layers
    - Physical file system
      - Processes physical blocks being read or written
      - Handles buffering and memory management
      - Physical placement of blocks in specific locations on the storage medium
      - Interacts with the device drivers
    - Virtual file system
      - Support for multiple concurrent instances of physical file systems, each of which is called a file system implementation
    - Logical file system
      - File access, directory operations, [and] security and protection
      - Consistency Guarantees

File System

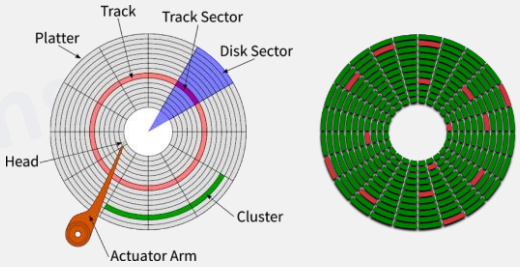
- File systems contain *metadata*
  - There is an *inode* for each file or directory, providing:
    - Locations of the data blocks* for the file
    - Attributes about the file, like “time last accessed” or “owner of the file”
  - The *inode table* records where each *inode* is located, indexed by number.
  - *Directories* are special files listing the names and *inode* numbers of files under the folder.



Internally the data content of a file is stored as a sequence of file system blocks

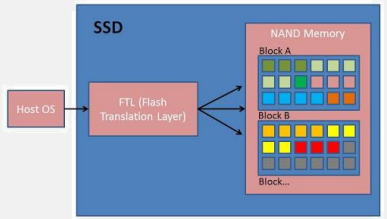
Block Storage

- Internally the data content of a file is stored as a sequence of file system *blocks*
  - Each block is a fixed number of bytes. The last block might not be full.
  - Within a block, all the bytes are sequential. However, within a file, the blocks might not reside sequentially on the disk.
  - Underlying storage systems are usually organized as blocks, and ideally file system blocks are aligned with the storage system's blocks.



Block Storage

- Internally the data content of a file is stored as a sequence of file system *blocks*
  - Each block is a fixed number of bytes. The last block might not be full.
  - Within a block, all the bytes are sequential. However, within a file, the blocks might not reside sequentially on the disk.
  - Underlying storage systems are usually organized as blocks, and ideally file system blocks are aligned with the storage system's blocks.



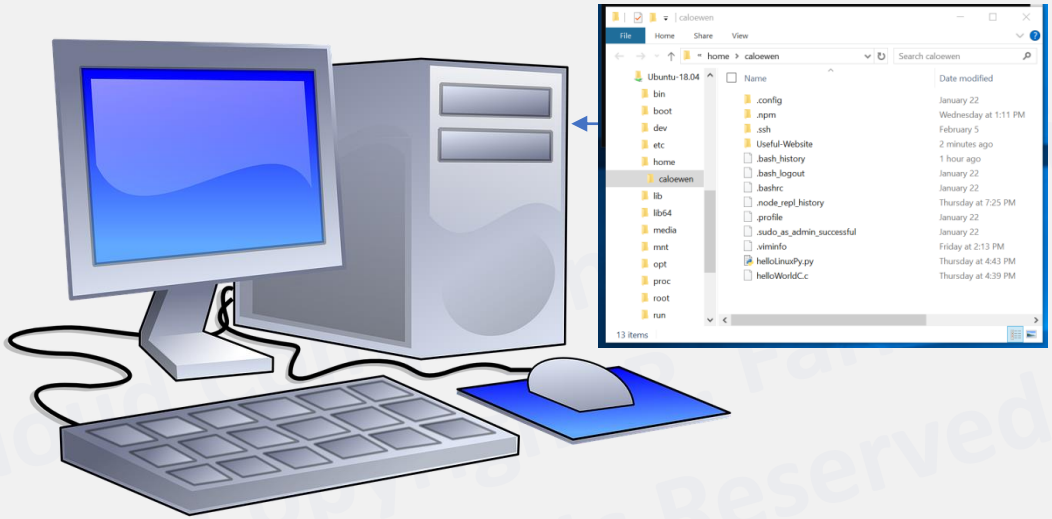
POSIX File System: IEEE Std 1003.1-2017

- Set of operations that the file system has to support
- POSIX file systems are organized as *directories* (folders) containing *files* (documents)
- POSIX file systems are *logically* treated as a sequence of bytes
- POSIX programming API
  - mount, umount the file system
  - open, close file descriptor
  - seek, lseek, fseek to a position in the byte stream
  - write, read to/from file descriptor
  - mkdir, rmdir, creat, unlink, link, symlink
  - fcntl (byte range locks, etc.)
  - stat, utimes, chmod, chown, chgrp
  - Path names are case sensitive, components are separated with forward slash (/).

POSIX Semantics

- POSIX Semantics
  - Define what is and is not guaranteed to happen when a POSIX I/O API call is made.
- *Example:*
  - *After a write() to a regular file has successfully returned:*
    - *Any successful read() from each byte position in the file that was modified by that write shall return the data specified by the write() for that position until such byte positions are again modified.*
    - *Any subsequent successful write() to the same byte position in the file shall overwrite that file data.*
  - Write() is strongly consistent
    - A write() is required to block application execution until the system can guarantee that any other read() call will see the data that was just written

File System



Summary

- Overview of block storage model
- File System Layers
  - Physical file system
  - Virtual file system
  - Logical file system
- POSIX File System
  - API
  - Semantics

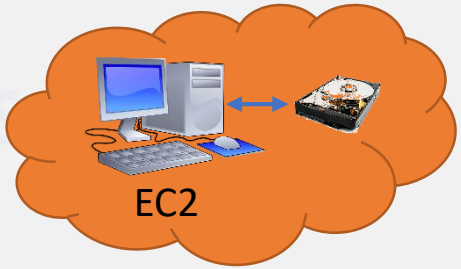
CLOUD COMPUTING APPLICATIONS

Cloud Storage: Block Storage

Prof. Reza Farivar

Cloud Storage Category 1: Block Storage – Instance Stores

- The physical machine running the virtual machine is physically connected to the storage device
  - virtual devices whose underlying hardware is physically attached to the host computer for the instance
  - Data transfer is limited by SATA / NVMe bandwidth
  - I3en.metal: 8 x 7,500 GB (60 TB)



- Since the machine may be rented to someone else in the next minute, the data stored on the drive does not persist, it's **ephemeral**.

- Example:
  - Amazon AWS: Instance Store
  - Google: Local SSD

Cloud Storage Category 1: Block Storage – Instance Stores

- The physical machine running the virtual machine is physically connected to the storage device
  - virtual devices whose underlying hardware is physically attached to the host computer for the instance
  - Data transfer is limited by SATA / NVMe bandwidth
  - I3en.metal: 8 x 7,500 GB (60 TB)

Instance Size	Local Storage (GB)	Read MBps	Write GBps
i3.large *	1 x 475 NVMe SSD	391	137
i3.xlarge *	1 x 950 NVMe SSD	806	273
i3.2xlarge	1 x 1,900 NVMe SSD	1,611	703
i3.4xlarge	2 x 1,900 NVMe SSD	3,223	1,406
i3.8xlarge	4 x 1,900 NVMe SSD	6,445	2,813
i3.16xlarge	8 x 1,900 NVMe SSD	12,891	5,469
i3.metal	8 x 1,900 NVMe SSD	12,891	5,469

** Throughputs are a snapshot in time, might be different now*

- Since the machine may be rented to someone else in the next minute, the data stored on the drive does not persist, it's **ephemeral**.

- Example:
  - Amazon AWS: Instance Store
  - Google: Local SSD

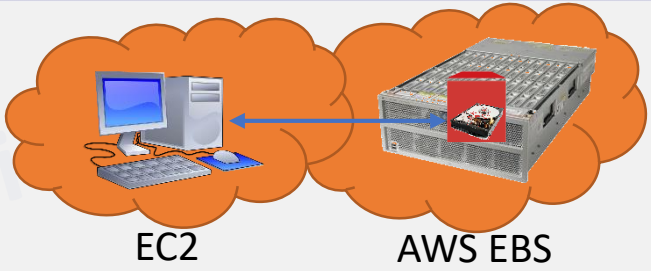
Storage Virtualization

- The process of presenting a logical view of the physical storage resources to a host computer system
  - Block Virtualization
  - File Virtualization

Cloud Block Storage – Virtual Block Stores

- Simulate a hard drive or SSD
- The physical machine running the virtual machine is separate from the physical machine hosting the data
  - NVMe over Fabric
    - NVMf or NVMe-oF
  - Data transfer is limited by network bandwidth

- The Hypervisor on the host machine has middleware that intercepts the network comm. and presents the network stream of bytes to the virtual machine as a “Block Storage Device”
  - E.g. in AWS, EBS as an NVMe device
  - E.g.in Google Cloud: Persistent Disk



Cloud Block Storage – Virtual Block Stores

- Typically less storage bandwidth than the previous option (EBS)
- You can also select how much bandwidth you are willing to pay for
  - Selecting IOPS for io1 type
- gp2 types are preselected and fixed
  - 3 IOPS/ gigabyte, 16KBps / IOPS, min 100, max 16,000
  - Bursting support
- Note: A single EC2 instance can be attached to more than 1 EBS volume
- Some instance types are EBS optimized
  - Bandwidth for EBS access is separate from network bandwidth
  - Other (micro, small, older generations) share network bandwidth

Instance Size	Instance Storage	EBS Bandwidth (MBps)
m5.large	EBS-Only	Up to 594
m5.xlarge	EBS-Only	Up to 594
m5.2xlarge	EBS-Only	Up to 594
m5.4xlarge	EBS-Only	594
m5.8xlarge	EBS Only	850
m5.12xlarge	EBS-Only	1,188
m5.16xlarge	EBS Only	1,700
m5.24xlarge	EBS-Only	2,375
m5.metal	EBS-Only	2,375

** Throughputs are a snapshot in time, might be different now*

AWS Elastic Block Storage

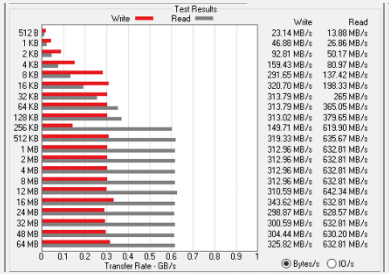
- Depending on IO requirements, different offerings and prices
  - gp2 up to 250 MB/s @ $0.10 per GB-month
  - io1 up to 1,000 MB/s @ $0.125 per GB-month AND $0.065 per IOPS-month
  - …
- Note: A single EC2 instance can be attached to more than 1 EBS volume
  - Max Throughput/Instance for T3 class: 2,375 MB/s

** Prices and bandwidths are a snapshot in time, might be different now*

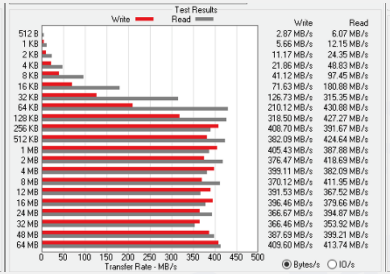
Instance Store vs. EBS Throughput

Experiment on an AWS m5ad.2xlarge instance (General Purpose, 8 vCPU, 32 GB RAM, 10GB network, EBS optimized)

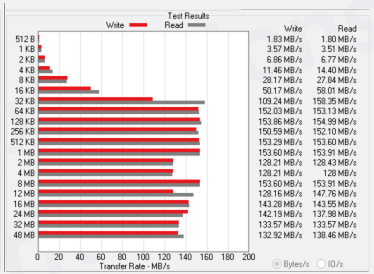
Instance Store 300GB



Io1 @ 200GB with 10000 IOPS



gp2 @ 30GB with 100/3000 IOPS

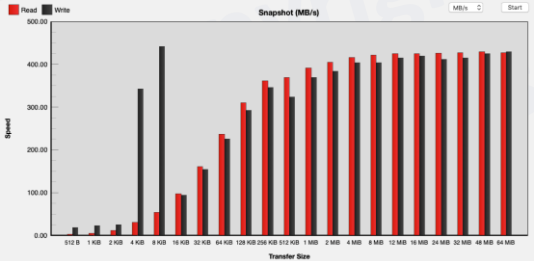


100*16KB = 1.56 MBps
3000*16KB = 45 MBps

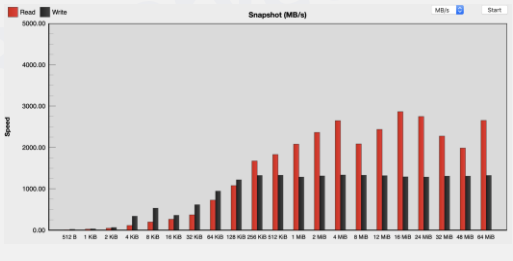
- Bursting
- IO Consolidation at EBS Backend

Experiment on laptop (Macbook Pro)

SSD via USB 3.0 (5 Gbit/s)



NVMe SSD (PCIe 3.0 × 4 8.0 GT/s (31.5 Gbit/s))



Summary

- Block Storage
- Instance Store
- Virtual Block Store
- Performance Comparison

CLOUD COMPUTING APPLICATIONS

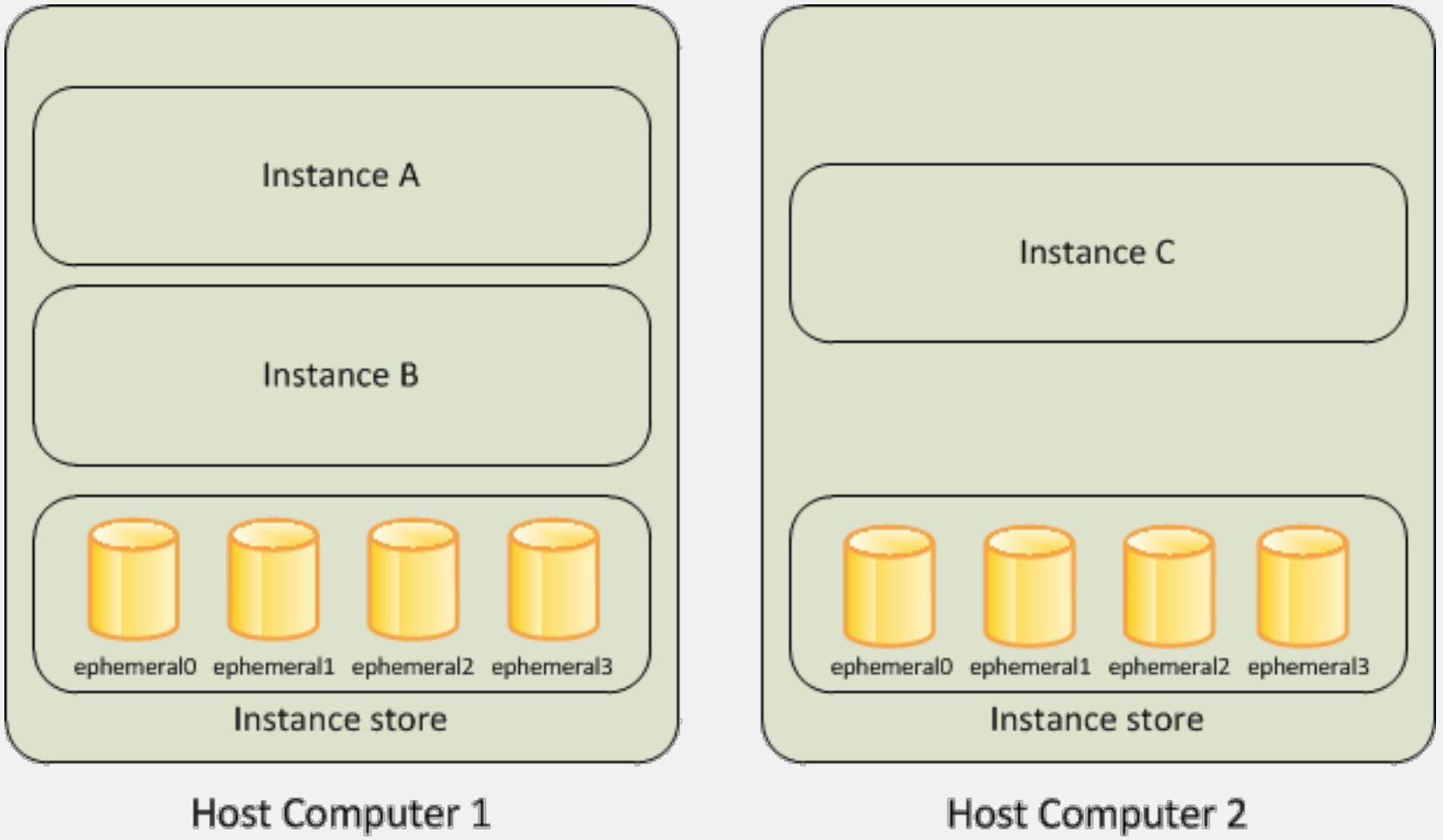
Amazon AWS Block Stores

Roy Campbell & Reza Farivar

Amazon AWS Instance Store

- Instance stores are another form of cloud-hosted **temporary** block-level storage
  - These are provided as part of an 'instance', such as an Amazon EC2
- Their contents will be lost if the cloud instance is stopped.
  - But offer higher performance and bandwidth to the instance.
  - Located on disks that are physically attached to the host computer
- They are best used for temporary storage such as caching or temporary files, with persistent storage held on a different type of server.

Amazon AWS Instance Store



Instance Store Lifetime

- Data persists only during the lifetime of its associated instance
  - It persists a reboot
- Data lost if
  - The underlying disk drive fails
  - The instance stops
  - The instance terminates
- You can get reliability by
  - A distributed file system (e.g. HDFS)
  - Backup to S3 or EBS

Instance Store Size

- A typical instance store is small
  - SSD: can be any where from around 80 GB to 320 GB SSD, up to 3,840 GB on x1.32xlarge
  - HDD: when available (on older generation instances), up to 1,680 GB

Amazon AWS EBS

- EBS Volumes are highly available and reliable
- Can be attached to running instances in the same availability zones
  - Persist independently of the life of an instance
- Use when data must be quickly accessible and requires long-term persistence
- Support encryption
- Up to 16TB in size

Amazon AWS EBS

- Different types
  - General Purpose SSD (gp2)
    - 100 IOPS/GiB, burst up to 10,000, 160 MB/s throughput
  - Provisioned IOPS SSD (io1)
    - Provision a specific level of performance
    - up to 20,000 IOPS and 320 MB/s of throughput
  - Throughput Optimized HDD (st1)
    - Low cost magnetic storage
    - Throughput of up to 500 MB/s
    - Large, sequential workloads such as Amazon EMR, ETL, data warehouses, and log processing
  - Cold HDD (sc1)
    - Inexpensive magnetic
    - Throughput of up to 250 MB/s

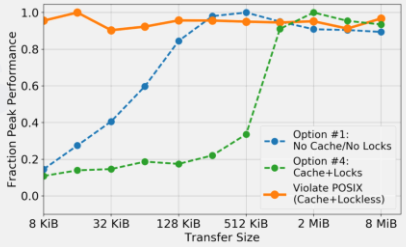
CLOUD COMPUTING APPLICATIONS

Cloud Storage: Object Storage

Prof. Reza Farivar

Cloud Object Storage

- As we have seen, Distributed File Systems are not easy
  - Considerable overhead, complexity, cost
  - Main reason: maintaining consistency while providing transparency
- CAP: Consistency, Availability, Partition Tolerance
- Transparency: "invisible" to client programs, which "see" a system which is similar to a local file system.
  - Behind the scenes, the distributed file system handles locating files, transporting data, and potentially providing other features listed below
  - While transparency may seem trivial, these semantics can incur a significant performance penalty at scale despite not being strictly necessary



Internet Scale Storage: Breaking the Chains of Transparency and Consistency

- What if we want to scale to “unlimited Storage”?
- Solution: Stop trying to have all 3 components of CAP
  - Availability: Important to keep, otherwise customer data may be unavailable
  - Partition Tolerance: Networks do fail, cloud providers have to be resilient
  - Consistency: Can be scarified
- The typical BLOB storage by a cloud provider can scale “infinitely” by being “eventually consistent”
- In addition, they typically are not POSIX compliant
- The access model is through REST APIs
  - GET, PUT, DELETE
- Examples:
  - AWS S3
  - OpenStack Swift

AWS S3 Consistency Model

- Objects have a URI, and are accessible by REST API calls
  - https://cloudApplications.s3.us-west-2.amazonaws.com/photos/picture1.jpg
- If you PUT to an existing key, a subsequent read might return the old data or the updated data, but it never returns corrupted or partial data.
- For Availability, data will be replicated across AWS datacenters (Availability Zones)
  - If a PUT request is successful, your data is safely stored. But temporarily:
    - A process writes a new object to Amazon S3 and immediately lists keys within its bucket. Until the change is fully propagated, the object might not appear in the list.
    - A process replaces an existing object and immediately tries to read it. Until the change is fully propagated, Amazon S3 might return the previous data.
    - A process deletes an existing object and immediately tries to read it. Until the deletion is fully propagated, Amazon S3 might return the deleted data.
    - A process deletes an existing object and immediately lists keys within its bucket. Until the deletion is fully propagated, Amazon S3 might list the deleted object.
  - Amazon S3 does not currently support object locking. If two PUT requests are simultaneously made to the same key, the request with the latest timestamp wins
  - Updates are key-based. There is no way to make atomic updates across keys.

Cloud Object Storage

- By relaxing the consistency model, building the distributed storage system becomes much simpler
- The Storage costs are significantly cheaper
- Bandwidth can be quite high
  - Up to 25 GB/s
- Unlike the previous models (Block Storage, Managed File System), data can be accessible from outside the cloud
  - Your mobile app customers all over the world
  - The small number of computers you personally own

AWS S3 tiers comparison

	S3 Standard	S3 Intelligent-Tiering*	S3 Standard-IA	S3 One Zone-IA†	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved
First byte latency	milliseconds	millseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

Service	Cost per 1TB / month
AWS EFS	$300
AWS FSx Lustre	$290
EBS gp2	$100
AWS EFS infrequent access	$25
S3 standard	**$23**
S3 infrequent	**$13**
S3 Glacier	$4
S3 Glacier deep archive	$1

** Prices and bandwidths are a snapshot in time, might be different now*

Summary

- Cloud Object Storage
- Consistency Model
- API
- Tiers

CLOUD COMPUTING APPLICATIONS

SWIFT – An Object Store

Roy Campbell & Reza Farivar

Definition

A **Binary Large OBject** (**BLOB**) is a collection of binary data stored as a single entity in a database management system.

(The blob was a science fiction movie featuring Steve McQueen.)

https://en.wikipedia.org/wiki/Binary_large_object

Use case

- Store unstructured object data like text or binary data
- Images
- Movies
- Audio, Signal data
- Large queue of messages
- Example is LinkedIn data in a user page (Uses Ambry)
- Usually accessible over the web

Examples

- Windows Azure Blob Storage
- Ambry – LinkedIn
- Facebook’s Warm BLOB Storage System
- Amazon Simple Storage Service (S3)*
- Apache Open Stack Blob Service (SWIFT)

Goals

- Data growth ~ 50% a year
- 50%-70% data is unstructured or archival
- RESTful API (HTTP)
- High availability (no single point of failure)
- Agile data centers
- Open Source
- Multi-region, geographic distribution of data
- Storage policies
- Erasure Coding

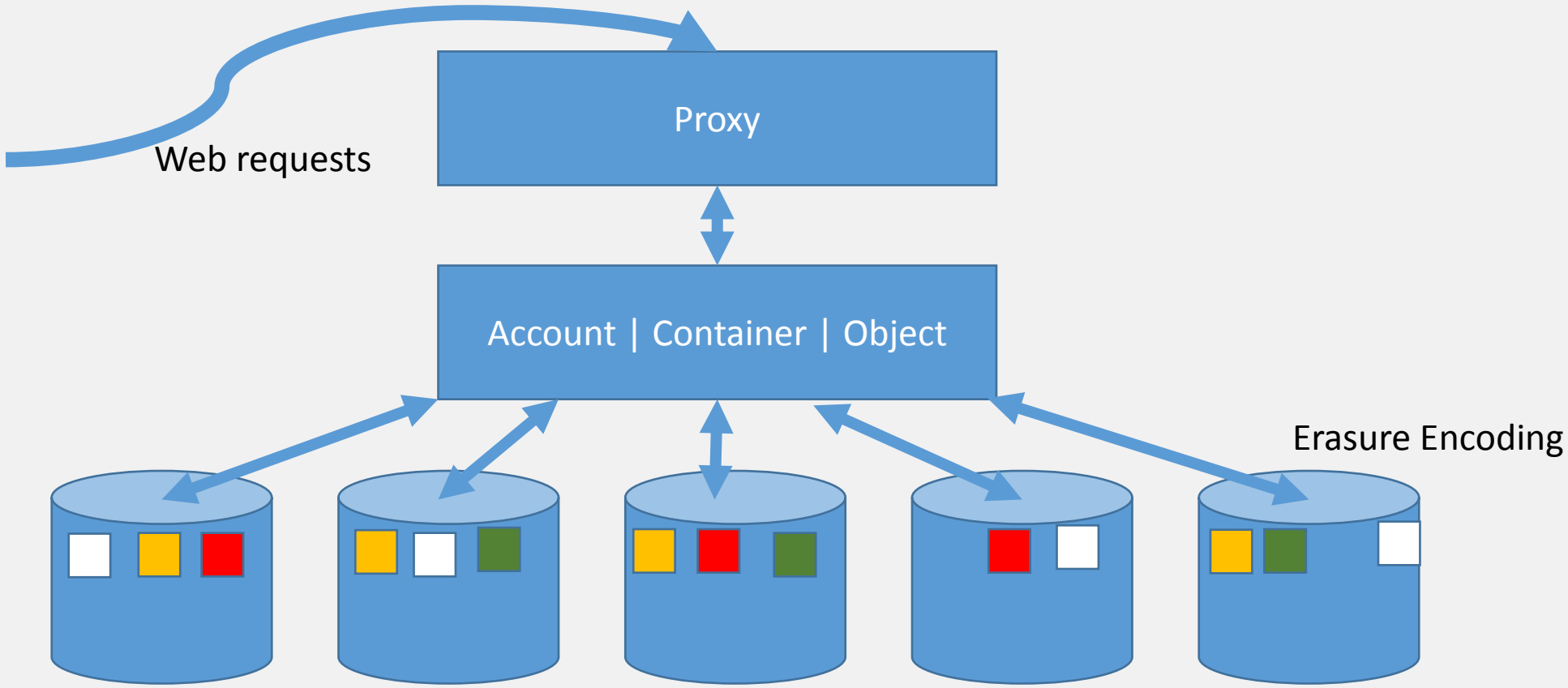
Swift API

https://swift.illinois.edu/version2/auth_account/container/object

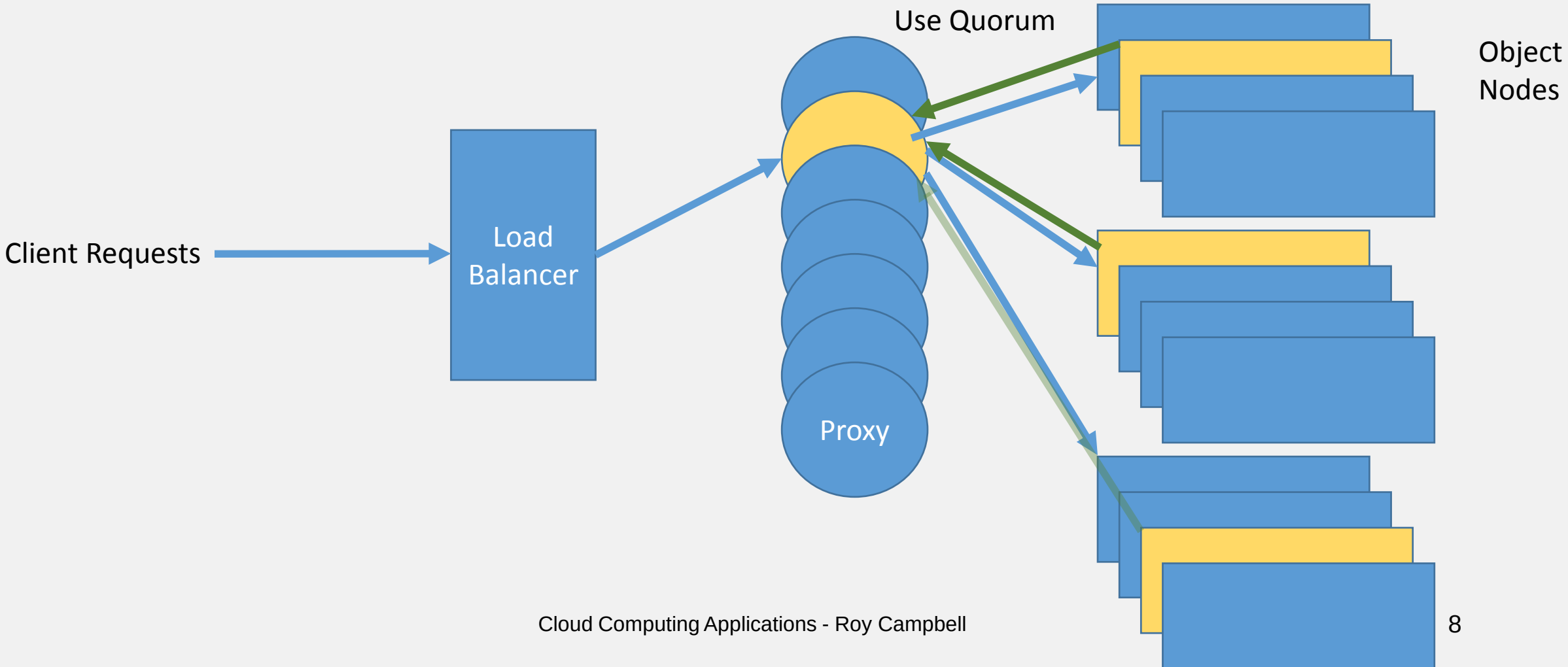
PUT /version2/roy/myblobs/classvideo1

GET /version2/reza/hisblobs/yesterdaysdataforhadoop

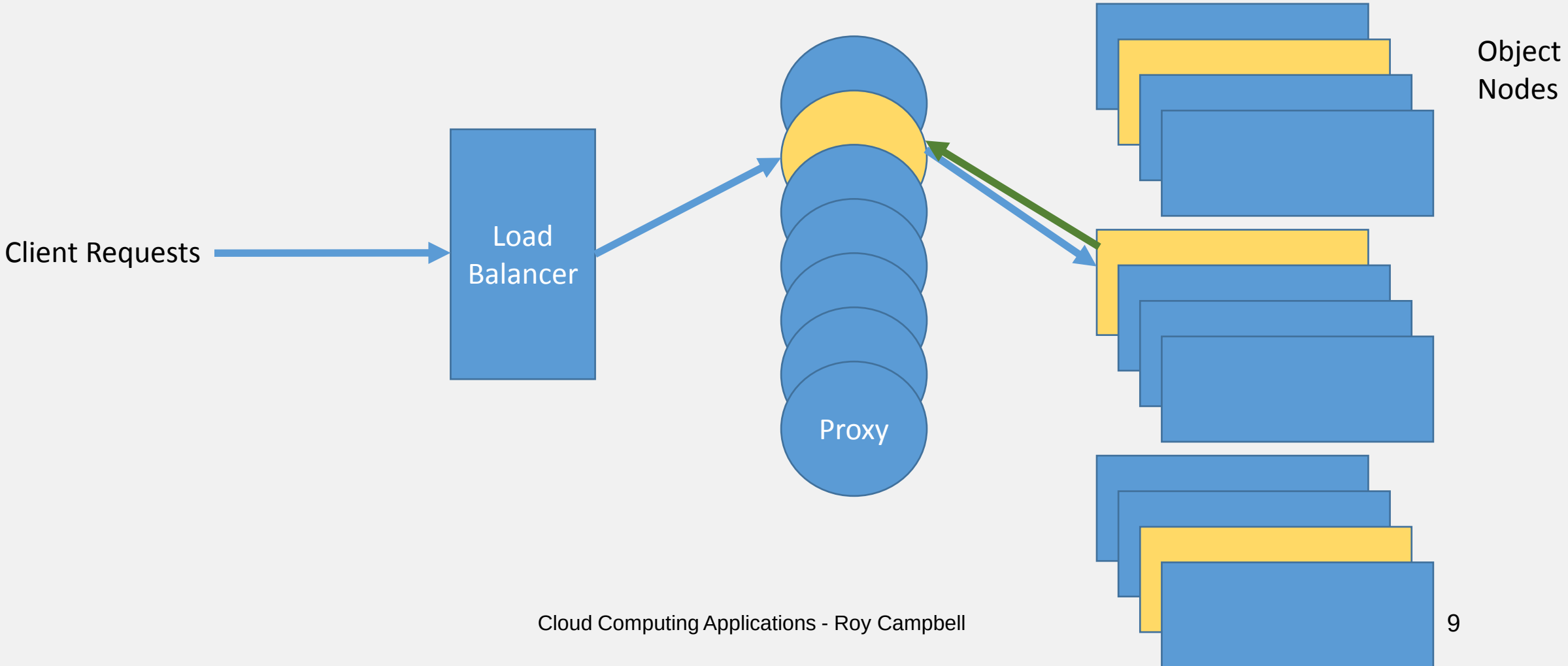
Swift Components



Write Requests - Load Balancer and Proxy



Read Requests - Load Balancer and Proxy

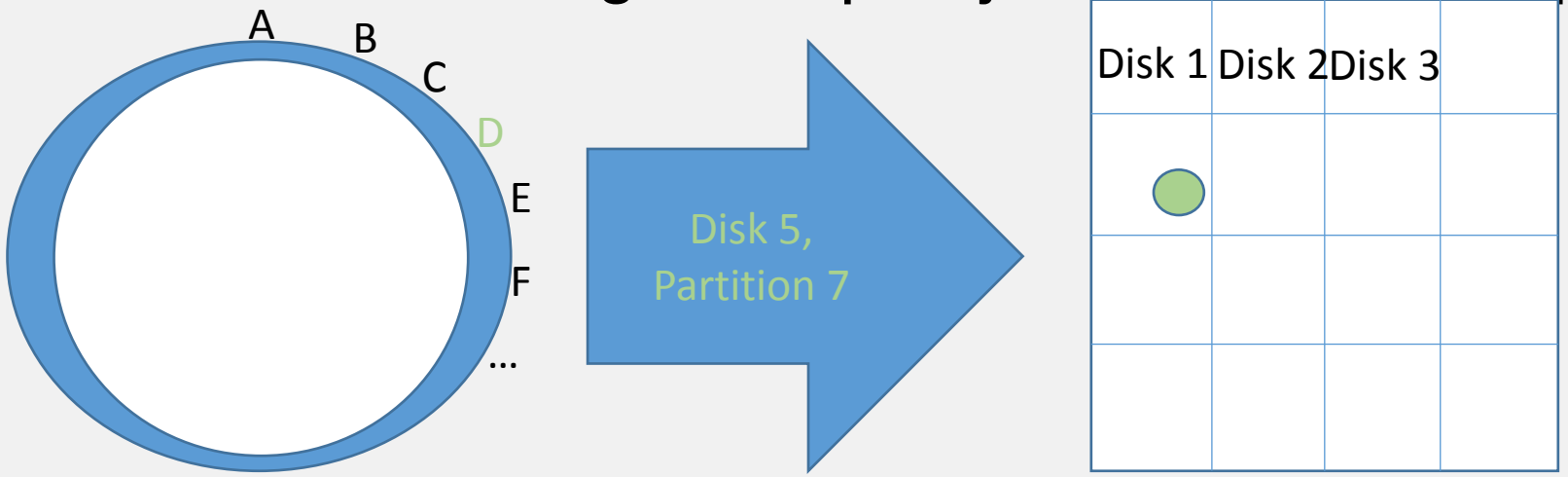


Details

- MD5 Checksums with each object
- Auditing and active replication
- Any sized disks

Swift Partitions

- 1 Node, 8 Disks, 16 Partitions per disk, 8*16 = 128 partitions
- 2 Nodes, 8 Disks each, 8 Partitions per disk, 8*16 = 128 partitions
- Use Hash into Ring to map objects into storage partitions



CLOUD COMPUTING APPLICATIONS

Amazon S3 BLOB Storage

Roy Campbell & Reza Farivar

Definition

Online file storage web service offered by Amazon Web Services. Amazon S3 provides storage through web service interfaces REST, SOAP, BitTorrent. (wikipedia)

https://aws.amazon.com/s3/

Use case

- Scalability, high availability, low latency – 99.99% availability
- Files up to 5 terabytes
- Objects stored in buckets owned by users
- User assigned keys refer to objects

- Amazon Machine Images (exported as a bundle of objects)
- SmugMug, Hadoop file store, Netflix, reddit, Dropbox, Tumbler

Simple Storage Service (S3)

- A **bucket** is a container for objects and describes location, logging, accounting, and access control.
  - A bucket has a name that must be **globally unique**.
  - http://*bucket*.s3.amazonaws.com
  - http://*bucket*.s3-*aws-region*.amazonaws.com.

- A bucket can hold any number of **objects**, which are files of up to 5TB.
  - http://bucket.s3.amazonaws.com/object
  - http://johnsmith.s3.amazonaws.com/photos/puppy.jpg

Fundamental operations corresponding to HTTP actions:

http://**bucket**.s3.amazonaws.com/**object**

- POST a new object or update an existing object.
- GET an existing object from a bucket.
- DELETE an object from the bucket
- LIST keys present in a bucket, with a filter.

A bucket has a **flat directory structure**

S3 Weak Consistency Model

“Updates to a single key are **atomic**….”

“Amazon S3 achieves high availability by replicating data across multiple servers within Amazon's data centers. If a PUT request is successful, your data is safely stored. However:

- A process writes a new object to Amazon S3 and immediately attempts to read it. Until the change is fully propagated, Amazon S3 might report "key does not exist."
- A process writes a new object to Amazon S3 and immediately lists keys within its bucket. Until the change is fully propagated, the object might not appear in the list.
- A process replaces an existing object and immediately attempts to read it. Until the change is fully propagated, Amazon S3 might return the prior data.
- A process deletes an existing object and immediately attempts to read it. Until the deletion is fully propagated, Amazon S3 might return the deleted data.”

S3 Command Line Interface

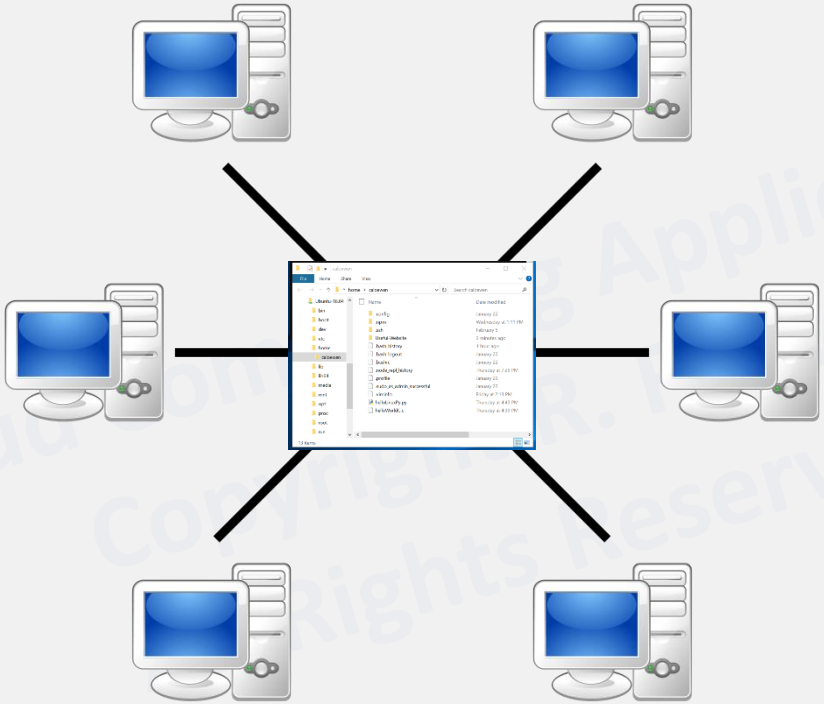
```
aws    s3   mb    s3://bucket
. . .       cp    localfile   s3://bucket/key
            mv    s3://bucket/key    s3://bucket/newname
            ls    s3://bucket
            rm    s3://bucket/key
            rb    s3://bucket

aws   s3    help
aws   s3    ls help
```

CLOUD COMPUTING APPLICATIONS

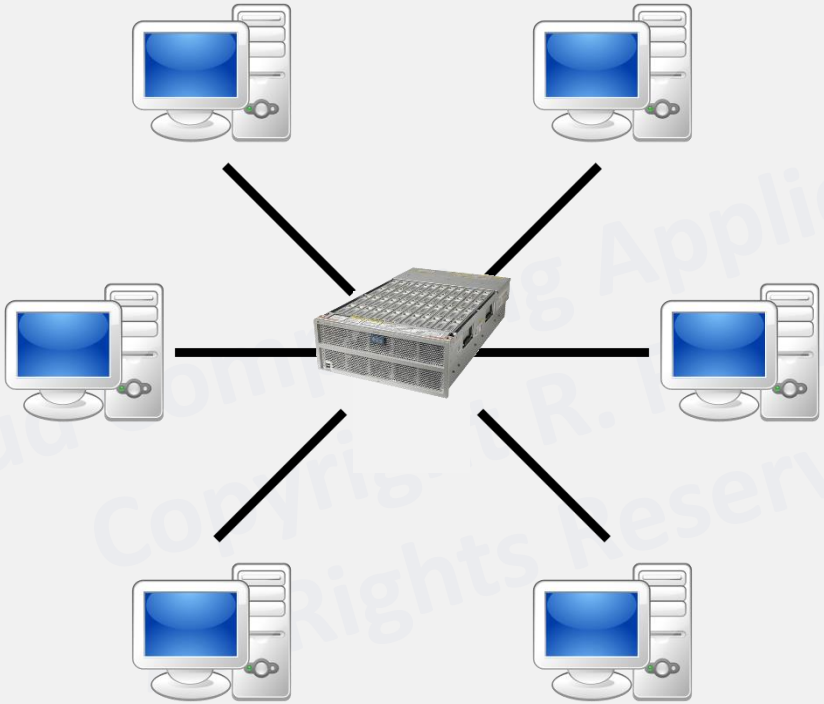
Cloud Storage: Managed File Systems

Prof. Reza Farivar

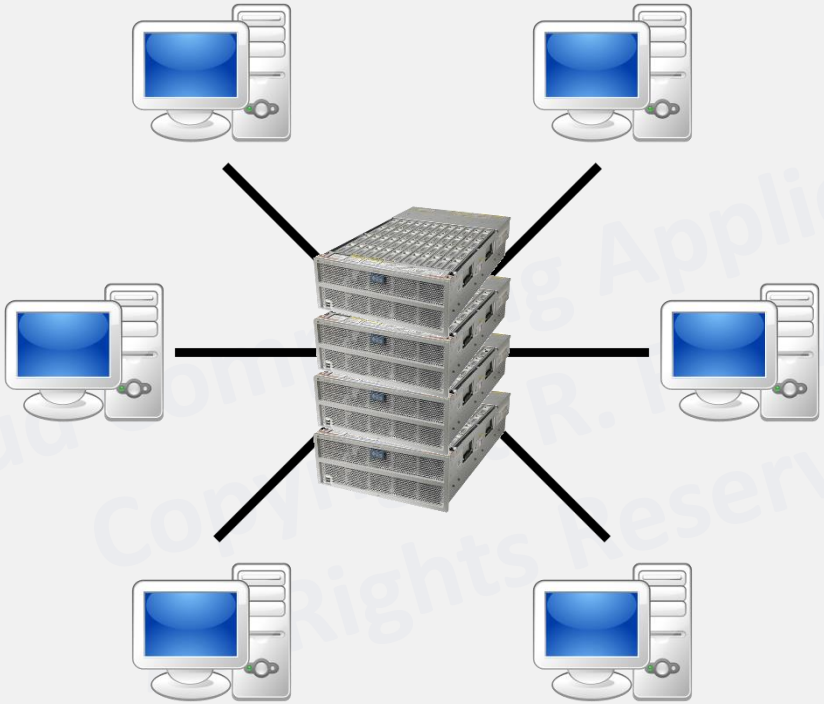
Logical View Networked File System



Network Attached Storage (NAS)



Network Attached Storage (NAS)



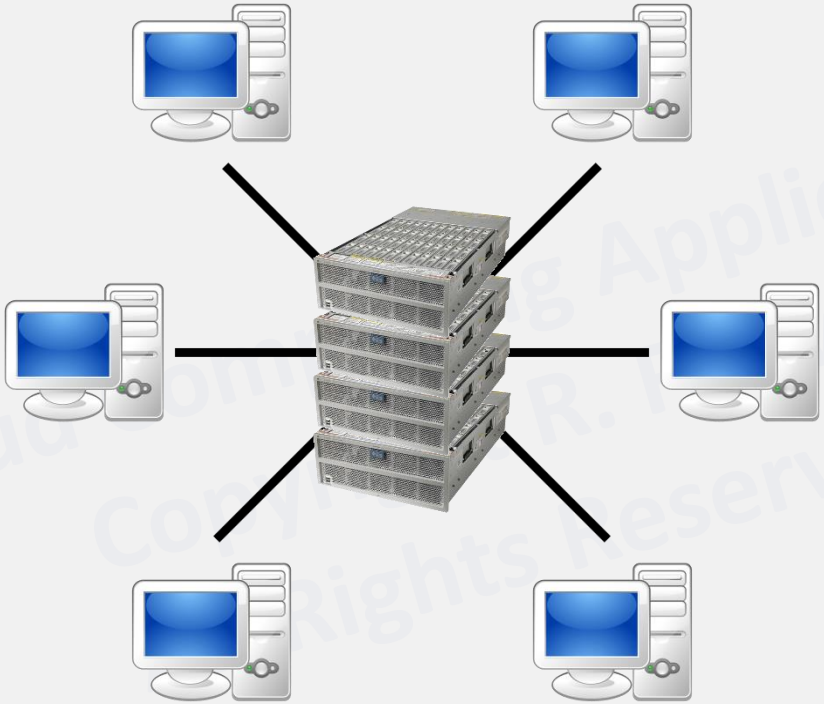
Clustered File Systems

- A clustered file system allows files to be accessed using the same interfaces and semantics as local files – for example, mounting/unmounting, listing directories, read/write at byte boundaries, system's native permission model.
  - Fencing
  - Concurrency
  - Consistency
- Examples
  - NFS
    - Unix
    - V4 , V4.1 (pNFS extension)
  - SMB
    - Windows
  - Lustre
    - HPC
  - Ceph
    - Many OpenStack implementations use Ceph as the storage substrate
  - Gluster
    - Classic file serving, second-tier storage, and deep archiving

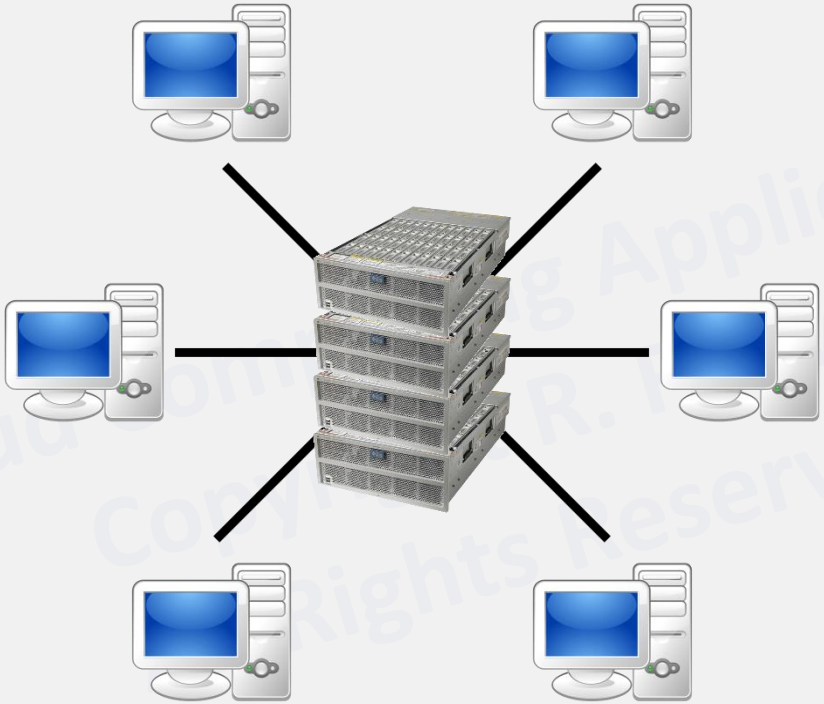
Clustered File System Consistency

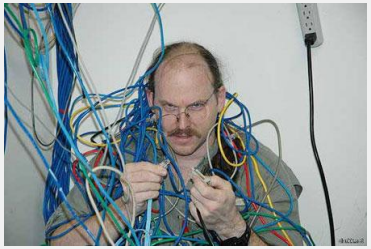
- Ideally, a parallel file system would completely hide the complexity of distributed storage and exhibit a behavior as it is specified by POSIX
  - Fencing
- Relatively easy on one Machine
- Much harder on a cluster of servers
- Maintaining CAP is extremely hard
  - **Consistency**: Every read receives the most recent write or an error
  - **Availability**: Every request receives a (non-error) response, without the guarantee that it contains the most recent write
  - **Partition tolerance**: The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes

Network Attached Storage (NAS)

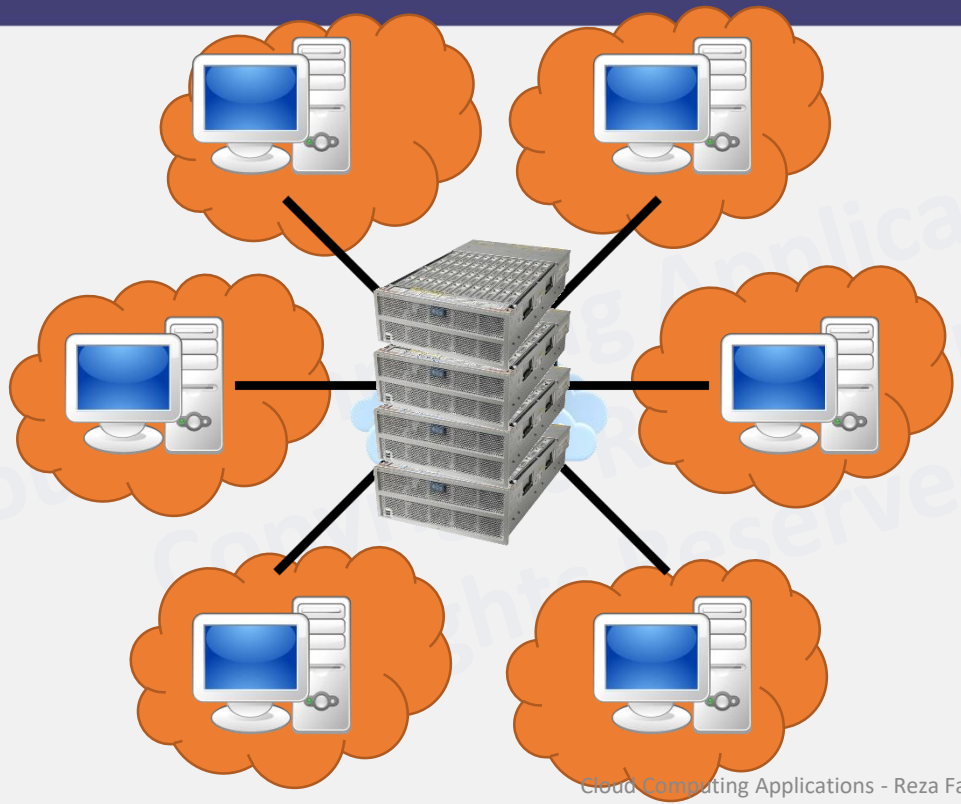


Network Attached Storage (NAS)



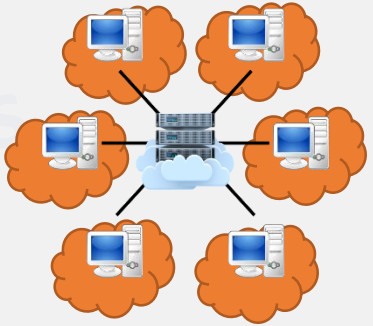


Network Attached Storage (NAS)



Cloud-Based File Systems

- Roll your own Clustered Distributed File System
  - Grab a number of “storage optimized” virtual machines, or metal machines
    - Each with large and fast instance block stores
      - A.k.a. SSDs / HDDs attached to the instance
    - Keep those instances on 24/7
    - Install a distributed file system on them
      - Lustre, MS DFS, NFS, Gluster, Ceph, Hadoop, etc.
- Managed filesystem deployed by the Cloud Provider



Cloud-managed file system

- Amazon
  - AWS FSx for Lustre
  - AWS FSx for Windows File Server
  - AWS EFS
- Azure
  - Azure Files
    - SMB access protocol
    - REST API
  - Azure DataLake Storage
    - Hadoop compatible file system
- Google
  - Cloud Filestore
    - NFS v3
    - Up to 64 TB

Google Filestore, IBM Cloud File Storage, AWS EFS

- Google Filestore

Simple commands to create a Filestore instance with gcloud.

```
gcloud filestore instances create nfs-server \
    --project=[PROJECT_ID] \
    --zone=us-central1-c \
    --tier=STANDARD \
    --file-share=name="vol1",capacity=1TB \
    --network=name="default"
```

Simple commands to install NFS, mount your file share, and set access permissions.

```
sudo apt-get -y update
sudo apt-get -y install nfs-common
sudo mkdir /mnt/test
sudo mount 10.0.0.2:/vol1 /mnt/test
sudo chmod go+rw /mnt/test
```

	Standard	Premium
Max read throughput	100 MB/s (1 TB), 180 MB/s (10+ TB)	1.2 GB/s
Max write throughput	100 MB/s (1 TB), 120 MB/s (10+ TB)	350 MB/s
Max IOPS	5,000	60,000
Max capacity per share	63.9 TB	63.9 TB
Typical customer availability	99.9%	99.9%
Protocol	NFSv3	NFSv3
Price	See Cloud Filestore pricing for more information	See Cloud Filestore pricing for more information

- IBM Cloud File Storage
  - Up to 12 TB
  - Up to 48,000 IOPS
- One beefy machine can handle them
  - AWS i3en.24xlarge: 8 x 7,500 NVMe SSD, 100 Gbps network
    - $60,405 / year per reserved instance
    - Cost of 64 TB standard storage EFS: $0.3*12*1024*64 = $235,929
    - Provisioned IO up to 1 GBps ~= 10 Gbps

Are Managed File Systems Distributed?

- FSx for Windows File Server
  - Max size: 64 TB
  - Can Utilize Microsoft DFS to unify data from many file servers for hundreds of petabytes
    - Shared namespace: Location transparency
    - Replication: Redundancy
- FSx for Lustre
  - Max size: 100 TB
  - Throughput: Read 50-200 MB/s per TB, can burst to 3,000 MB/s per TB
  - E.g. 50.4 TB runs on 22 file servers
    - Hundreds of GB/s of throughput
- AWS EFS
  - Maximum size: “Petabytes”
  - The throughput available to a file system scales as a file system grows
    - 50 MB/s per TB, can burst to 100 MB/s per TB
  - Supports NFS v4.1
- Azure
  - Azure Files
    - 100 TB
  - Data Lake Storage Gen 2
    - HDFS semantics
    - Built on top of Azure Blob Storage
    - Distributed file system
      - Can serve “many exabytes”
      - Throughput measured in gigabits per second (Gbps)

Distributed file Systems Design Goals

- *Access transparency*: clients are unaware that files are distributed and can access them in the same way as local files are accessed.
- *Location transparency*: a consistent namespace exists encompassing local as well as remote files. The name of a file does not give its location.
- *Concurrency transparency*: all clients have the same view of the state of the file system. This means that if one process is modifying a file, any other processes on the same system or remote systems that are accessing the files will see the modifications in a coherent manner.
- *Failure transparency*: the client and client programs should operate correctly after a server failure.
- *Heterogeneity*: file service should be provided across different hardware and operating system platforms.
- *Scalability*: the file system should work well in small environments (1 machine, a dozen machines) and also scale gracefully to bigger ones (hundreds through tens of thousands of systems).
- *Replication transparency*: Clients should be unaware of the file replication performed across multiple servers to support scalability.
- *Migration transparency*: files should be able to move between different servers without the client's knowledge.

Summary

- Clustered File Systems
- File Systems in the Cloud

CLOUD COMPUTING APPLICATIONS

Amazon AWS EFS

Roy Campbell & Reza Farivar

Amazon AWS EFS

- Elastic File System
- Motivation: enterprise customers need a large distributed file system
  - S3 is large and distributed, but it is an object store without performance guarantees and eventual consistency model
  - Block storage (EBS, instance store) are small
    - Enterprise can build a distributed file system on top of these, but it requires operational expertise
  - Glacier: Good for only archival storage
- EFS provides a fully NFSv4 compliant network file system

Amazon AWS EFS

- SSD backed
- Highly available and highly durable
  - files, directories, and links are stored redundantly across multiple Availability Zones within an AWS region
- Grow or shrink as needed
  - No need to pre-provision capacity

Amazon AWS EFS

		Amazon EFS	Amazon EBS PIOPS
Performance	Per-operation latency	Low, consistent	Lowest, consistent
	Throughput scale	Multiple GBs per second	Single GB per second
Characteristics	Data Availability/Durability	Stored redundantly across multiple AZs	Stored redundantly in a single AZ
	Access	1 to 1000s of EC2 instances, from multiple AZs, concurrently	Single EC2 instance in a single AZ
	Use Cases	Big Data and analytics, media processing workflows, content management, web serving, home directories	Boot volumes, transactional and NoSQL databases, data warehousing & ETL

CLOUD COMPUTING APPLICATIONS

Ceph

Roy Campbell & Reza Farivar

Motivation

- Ceph is an emerging technology in the production-clustered environment
- Designed for:
  - Performance – Striped data over data servers
  - Reliability – No single point of failure
  - Scalability – Adaptable metadata cluster
  - More general than HDFS
    - Smaller files
- GlusterFS has more or less similar ideas
  - Uses ring-based hashing; Ceph uses CRUSH

Ceph Overview

- MDS – Meta Data Server
- ODS – Object Data Server
- MON – Monitor (now fully implemented)

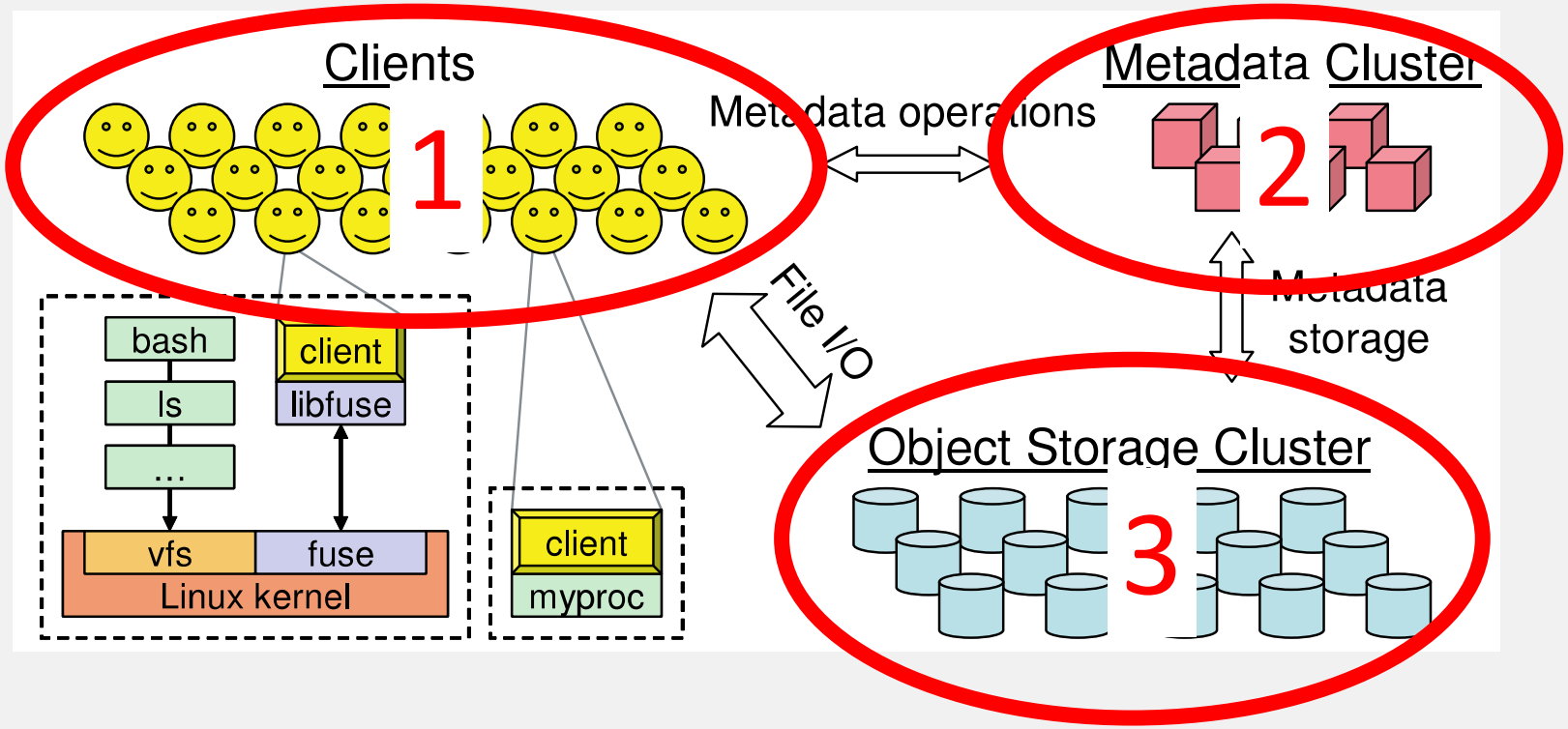
- **Decoupled data and metadata**
  - I/O directly with object servers

- **Dynamic distributed metadata management**
  - Multiple metadata servers handling different directories (subtrees)

- **Reliable autonomic distributed storage**
  - ODS’s manage themselves by replicating and monitoring

Ceph Components

- Ordered: Clients, Metadata, Object Storage



Decoupled Data and Metadata

- Increases performance by limiting interaction between clients and servers
- Decoupling is common in distributed filesystems: HDFS, Lustre, Panasas…
- In contrast to other file systems, Ceph uses a function to calculate the block locations

Dynamic Distributed Metadata Management

- Metadata is split among cluster of servers
- Distribution of metadata changes with the number of requests to even load among metadata servers
- Metadata servers also can quickly recover from failures by taking over neighbors' data
- Improves performance by leveling metadata load

Reliable Autonomic Distributed Storage

- Data storage servers act on events by themselves
- Initiates replication and
- Improves performance by offloading decision making to the many data servers
- Improves reliability by removing central control of the cluster (single point of failure)

CLOUD COMPUTING APPLICATIONS

Cloud Storage: Archive Storage and Backup

Prof. Reza Farivar

Cloud Archive Storage

- The availability of the storage has a direct impact on costs
- If you know file requests are very infrequent, you can store them in cold storage
- Cloud providers offer archive storage at much reduced costs
  - However, access to them is relatively expensive
- Different tiers of archive storage

Amazon Archive Storage: S3 Glacier

- AWS S3 Glacier
  - $0.004 per GB / mo
  - Compare with S3
    - Frequent access tier: $0.023 per GB / mo
    - Infrequent access tier: $0.0125 per GB / mo
- Glacier Retrieval
  - retrieval option from 1 minute to 12 hours

Retrieval Time	Data Retrievals
Expedited (typically available within 1 – 5 minutes)	$0.03 per GB / mo
Standard (typically accessible within 3 – 5 hours)	$0.01 per GB / mo
Bulk	$0.0025 per GB / mo

  - Retrieval request

Retrieval Time	Retrieval Requests
Expedited	$10.00 per 1,000 requests
Standard	$0.05 per 1,000 requests
Bulk	$0.025 per 1,000 requests

- S3 Glacier Deep Archive
  - For long-term data archiving that is accessed once or twice in a year and can be restored within 12 hours
  - $0.00099 per GB / mo

** Prices and bandwidths are a snapshot in time, might be different now*

AWS S3 tiers comparison

	S3 Standard	S3 Intelligent-Tiering*	S3 Standard-IA	S3 One Zone-IA†	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved
First byte latency	milliseconds	millseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

Service	Cost per 1TB / month
AWS EFS	$300
AWS FSx Lustre	$290
EBS gp2	$100
AWS EFS infrequent access	$25
S3 standard	**$23**
S3 infrequent	**$13**
S3 Glacier	$4
S3 Glacier deep archive	$1

** Prices and bandwidths are a snapshot in time, might be different now*

Cloud-managed Backups

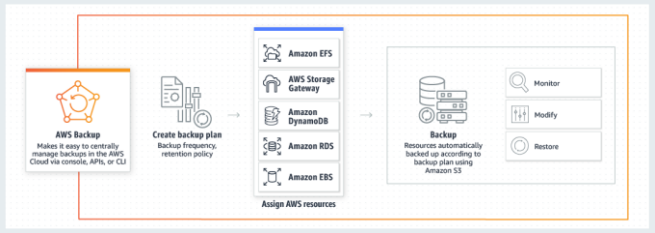
- Some cloud providers allow managed backups from their block stores or managed file systems or other data stores
  - Distributed Backup is not a simple task
- Example: Amazon backup

Resource Type	Warm Storage	Cold Storage
Amazon EFS File System Backup	$0.05 per GB-Month	$0.01 per GB-Month
Amazon EBS File System Backup	$0.05 per GB-Month	n/a†
Amazon RDS Database Snapshot	$0.095 per GB-Month	n/a†
Amazon DynamoDB Table Backup	$0.10 per GB-Month	n/a†
AWS Storage Gateway Volume Backup	$0.05 per GB-Month	n/a†

- Amazon restore

Resource Type	Warm Storage	Cold Storage	Item-level Restore
Amazon EFS File System Backup	$0.02 per GB	$0.03 per GB	$0.5 per request
Amazon EBS Volume Snapshot	Free	n/a†	n/a**
Amazon RDS Database Snapshot	Free	n/a†	n/a**
Amazon DynamoDB Table Backup	$0.15 per GB	n/a†	n/a**
AWS Storage Gateway Volume Backup	Free	n/a†	n/a**

- Example: Azure
  - Only supports backups of VMs (block stores) and SQL workloads



CLOUD COMPUTING APPLICATIONS

Amazon AWS Glacier

Roy Campbell & Reza Farivar

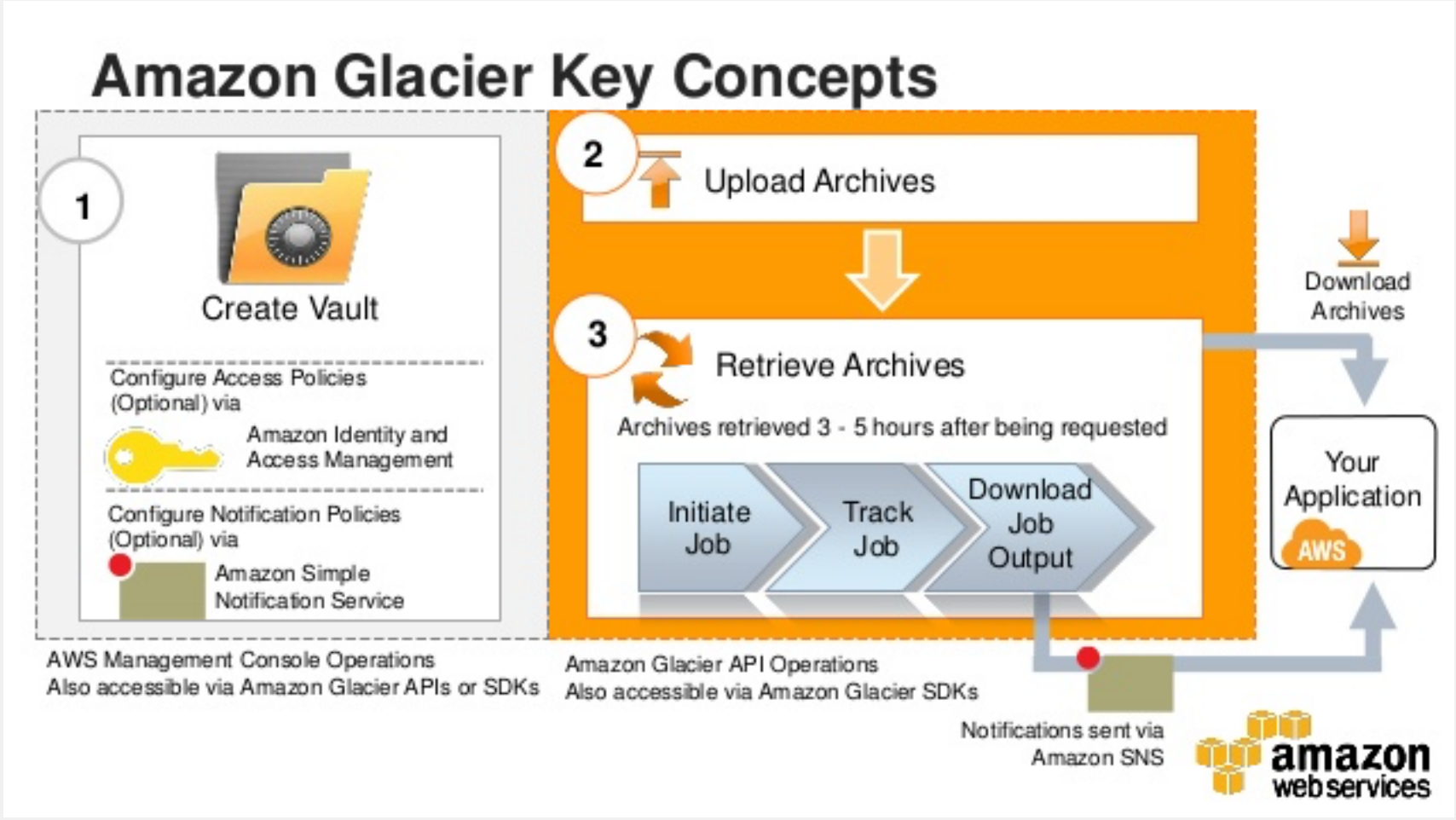
Amazon AWS Glacier

- Allows you to archive your data
- Very low cost, $0.007 per GB per month
- Very durable
  - Average annual durability of 99.999999999%
- Each single archive up to 40 TB

Amazon AWS Glacier

- Archives stored in vaults
- The main access point to glacier is S3
- Typically takes between **3 to 5 hours** to prepare a download request
  - After that you have 24 hours to download from the staging location

Amazon AWS Glacier



CLOUD COMPUTING APPLICATIONS

Cloud Storage: Storage Gateway and Mass Data Transfer

Prof. Reza Farivar

Cloud Storage Gateways

- Hybrid Cloud
  - Some VMs on your infrastructure, some on public cloud
  - Need a way to connect storage locations
- AWS Storage Gateway
  - File Gateway
    - NFS, SMB
    - S3 REST API access
      - POSIX-style metadata, including ownership, permissions, and timestamps stored as S3 user metadata
  - Tape Gateway
  - Volume Gateway
    - Presented as iSCSI protocol
    - Stores in AWS as EBS
- Azure
  - Data Box Gateway
    - SMB, NFS
    - Azure Blob Storage
    - Azure Files
  - File Sync
  - StorSimple

Cloud Mass Data Transfer

- Once a company decides to move to the cloud, there is a mass of data that needs to be transferred to the cloud
- Others may need to transfer data into and out of the cloud
- Problems:
  - The cost of mass data transfer could get astronomical
  - Network reliability could make it very hard
- Typical Solution: Cloud provider sends you a physical device
- AWS
  - Snowball: $250 / 80 TB
    - Moving data into Amazon is free, same as S3
    - In comparison, transferring 80TB into/out of S3 can take 7-8 days on a 1Gbps internet connection, if there is no data transmission error
      - 80*8*1024 / (60*60*24) = 7.58 days
    - Moving 80TB out of S3 can cost
      - 80*1024*0.08 = $6,553
    - Moving 80TB out through Snowball
      - 80*1024*0.03 = $2,457
  - Snowmobile
- Azure Data Box
  - Data Box: $250 / 100 TB
  - Data Box Disk: $50 / 8 TB
  - Data Box Heavy: $4,000 / 1 PB

- *Shipping fees also apply*

AWS Snowball: 50-80 TB

Cloud Mass Data Transfer

- Once a company decides to move to the cloud, there is a mass of data that needs to be transferred to the cloud
- Others may need to transfer data into and out of the cloud
- Problems:
  - The cost of mass data transfer could get astronomical
  - Network reliability could make it very hard
- Typical Solution: Cloud provider sends you a physical device
- AWS
  - Snowball: $250 / 80 TB
    - Moving data into Amazon is free, same as S3
    - In comparison, transferring 80TB into/out of S3 can take 7-8 days on a 1Gbps internet connection, if there is no data transmission error
      - 80*8*1024 / (60*60*24) = 7.58 days
    - Moving 80TB out of S3 can cost
      - 80*1024*0.08 = $6,553
    - Moving 80TB out through Snowball
      - 80*1024*0.03 = $2,457
  - Snowmobile
- Azure Data Box
  - Data Box: $250 / 100 TB
  - Data Box Disk: $50 / 8 TB
  - Data Box Heavy: $4,000 / 1 PB

AWS Snowmobile: 100 PB

Assuming a 7-day transfer time, bandwidth = 824 Gbps

- *Shipping fees also apply*

Summary

- Storage Gateway
- Mass Data Transfer

CLOUD COMPUTING APPLICATIONS

Cloud Storage: Internet-level Filesystem Storage

Prof. Reza Farivar

Internet-level Filesystem Storage

- Object Storage
  - AWS S3, Azure Blob Storage, etc.

- DropBox
- Google Cloud
- Box
- Apple iCloud Drive

Internet-level Filesystem Storage

- Synch and access your file system across a limited number of owned devices
- Block-level file copying
- Synch throttling
- Shared folders and links
- Data recovery (e.g. 30 days, 180 days, etc.)
- File Sync with local file system
- Web-based access
- Shared folders with other users
- Analytics services
  - Text search in documents and images
  - etc.

Data access frequency

- Unlike Object Stores (AWS S3, Azure Blobs, etc.) this market segment does not accept access pattern limits
- Cloud providers use statistics to extract average access patterns and set pricing
  - They may lose money on some customers that access / change their data all the time
  - In average, they have positive profit
- Case study: DropBox
  - Dropbox started as a startup by storing their customer's data on Amazon S3
  - IN 2016 Dropbox moved off of Amazon and onto their own datacenters for storage
  - The economy of scale had grown so far that their "cloudonomics" dictated they should build vs. rent
    - Think of Amazon AWS as an incubator for startups
    - Netflix moved off its CDN from third parties to its own network once it grew large enough
      - Netflix still uses AWS & Google Cloud for everything else: compute, analytics, etc.
      - No Datacenter

Do not include transfer costs

Service	Cost per 2TB / month
AWS EFS	$600
AWS EFS infrequent access	$50
S3 standard	$46
S3 infrequent	$25
S3 Glacier	$8
S3 Glacier deep archive	$2
Box	$15
Dropbox	$10
Google Cloud	$10
Amazon Drive	$10
Apple iCould Drive	$10
pCloud	$10
Microsoft OneDrive	$7
Sync.com	$7

** Prices and bandwidths are a snapshot in time, might be different now*

Integration

- Custom integration in company ecosystem
  - Microsoft OneDrive: Comes with Office 365
  - Google Cloud: Integration with Google Docs, Gmail
  - Apple iCould Drive: Integration with iPhones, Macs
  - Dropbox: first to market, product typically more robust, integration with both MS Office and Google Docs
  - Box: Enterprise and government features

Summary

- End-user facing Cloud Storage
- Many competing offerings
- Utilize low frequency access pattern for low-cost offerings