BITS Pilani
Pilani | Dubai | Goa | Hyderabad

DSECL ZG 522: Big Data Systems

Session 11: Cloud computing

Janardhanan PS

Professor

janardhanan.ps@wilp.bits-pilani.ac.in

Where are we in the course

- Basics: LoR, caching, parallel / distributed systems, understand performance, availability etc.
- Advanced concepts: Distributed program design, Consistency models, CAP theorem
- Hadoop ecosystem - file based batch processing systems — compute, storage/FS, DBs, workflows, cluster mgmt, coordination systems (ZK) etc.
- NoSQL DBs
- Cloud compute and storage options <—- design a Big Data system on Cloud
- In-memory / Streaming systems - Spark

Topics for today

- Cloud concepts
- Compute and network (Storage : next session)
- Best practices

Definition

The US National Institute of Standards (NIST) defines cloud computing as follows:

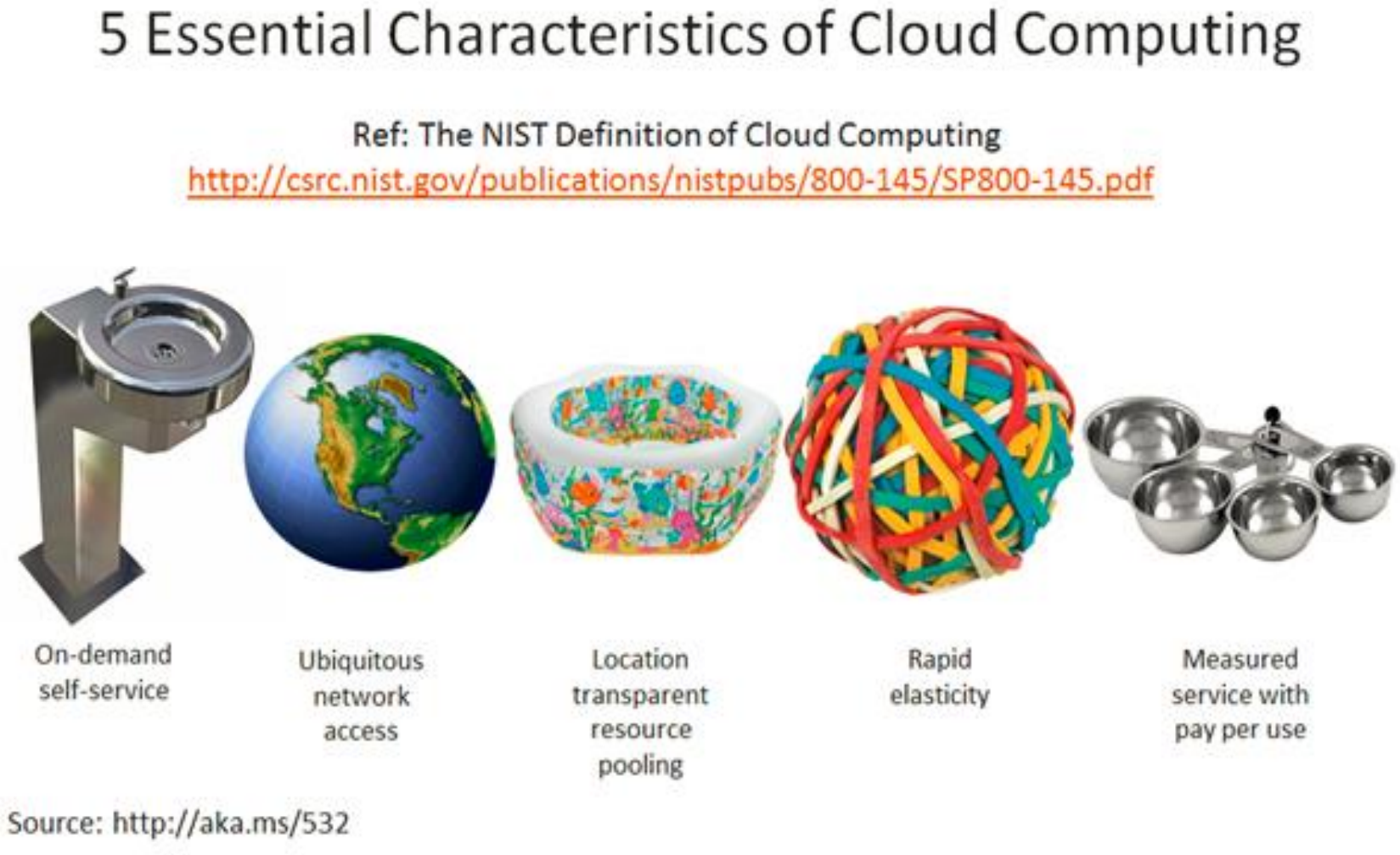
> Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

NIST's 3-4-5 rule of Cloud Computing

- 3 cloud service models or service types for any cloud platform
- 4 deployment models
- 5 essential characteristics of cloud computing infrastructure

5 Characteristics of Cloud Computing

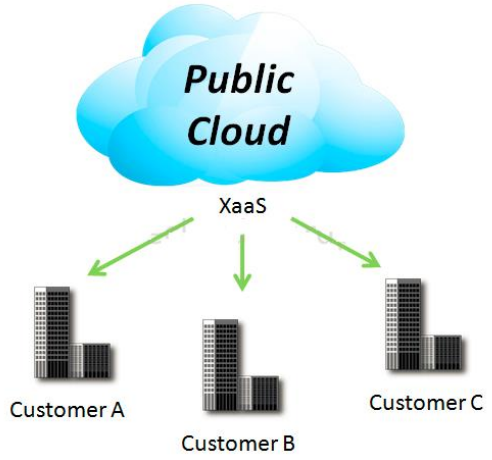
- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service



4 Deployment models

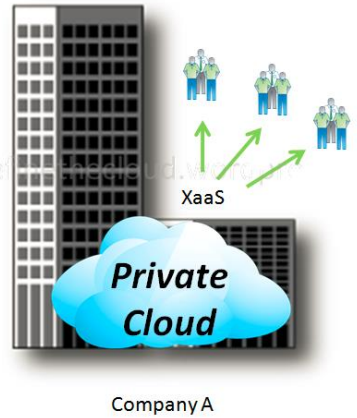
Public Cloud

Mega-scale cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services. E.g. AWS, Azure, Google, …



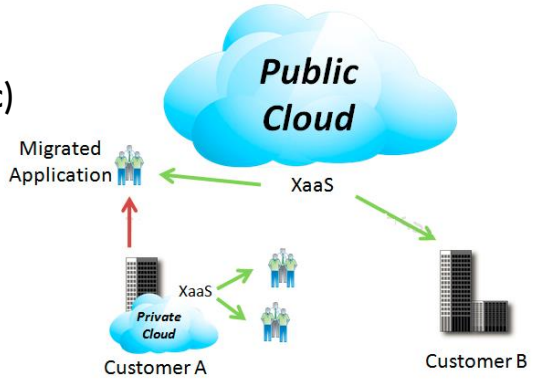
Private Cloud

The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.



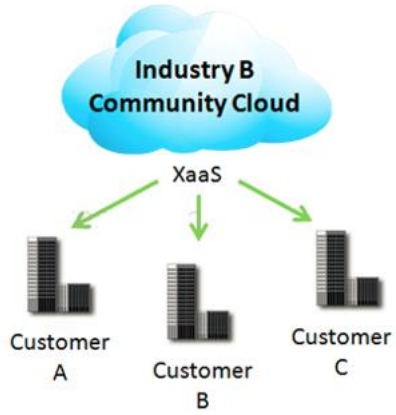
Hybrid Cloud

The cloud infrastructure is a composition of two or more clouds (private or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability, cross domain security etc.

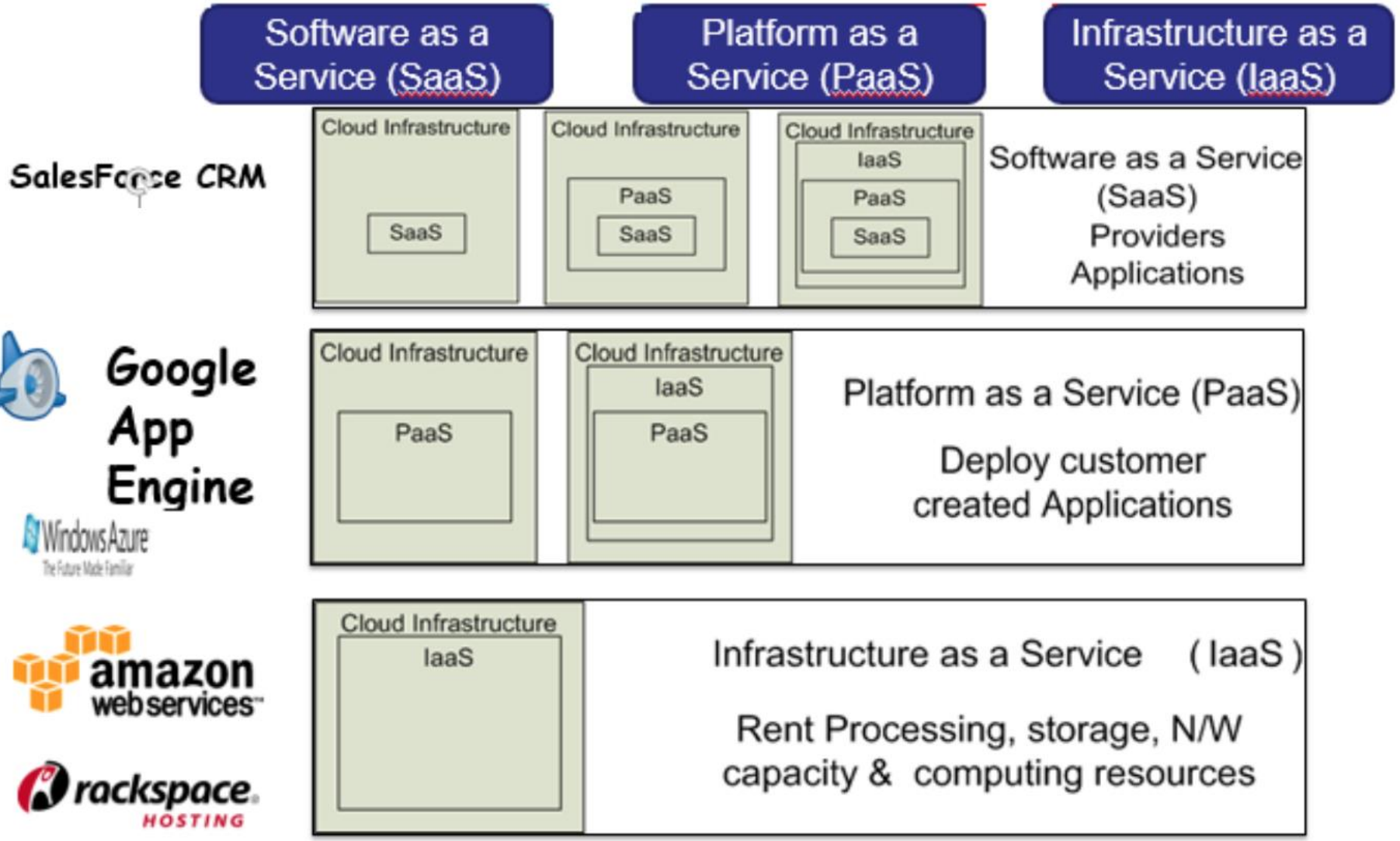


Community Cloud

An 'infrastructure shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise' according to NIST.



3 Cloud Service Models



Cloud services for Big Data

- Cloud follows same model as Big Data, both requiring distributed clusters of computing devices.
  - ✓ Cloud Computing considered as ideal platform for handling big data

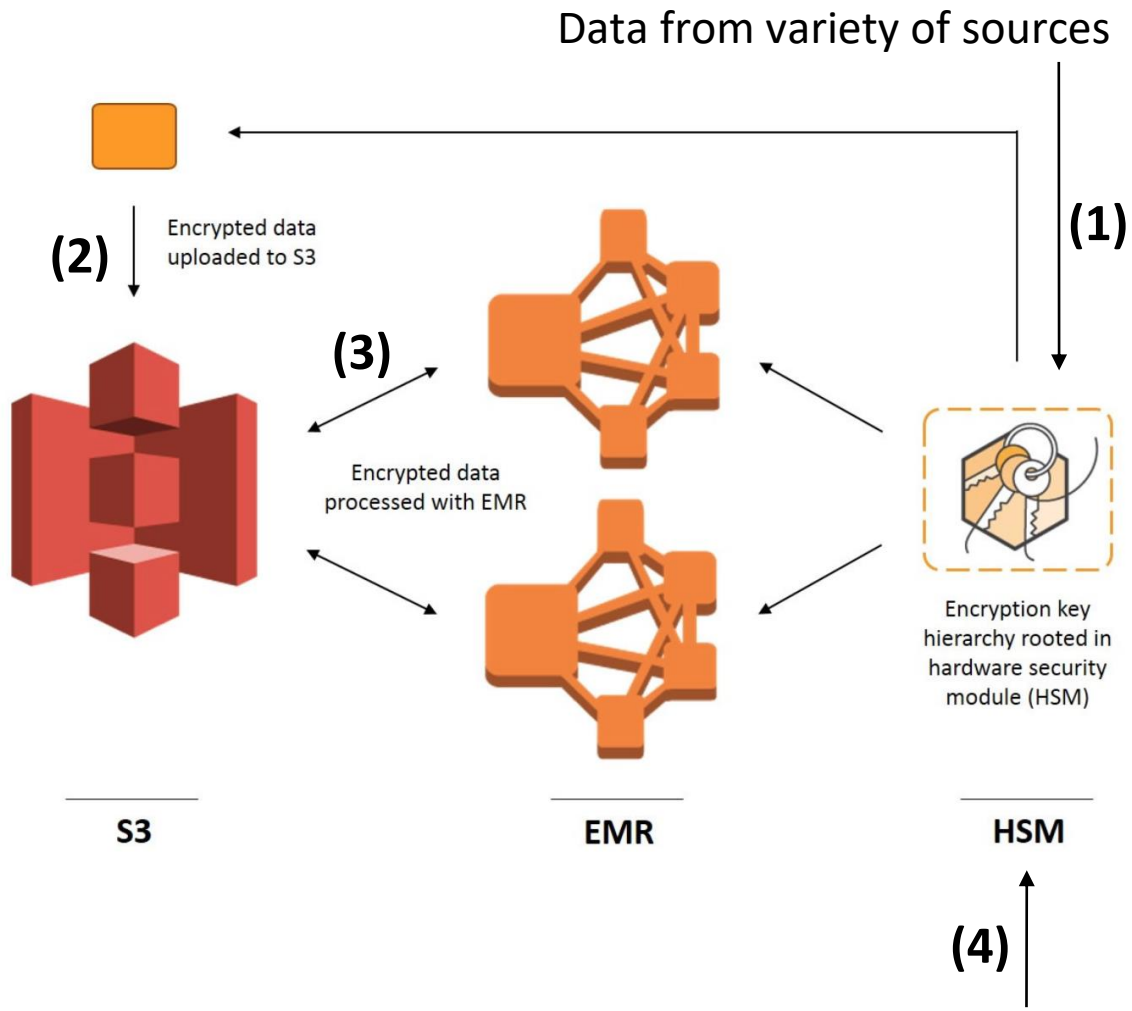
- IaaS:
  - ✓ Can provide huge storage and computational power requirements for Big Data through limitless storage and computing ability of cloud computing, e.g. AWS S3, EC2
- PaaS:
  - ✓ Vendors offers platforms ready with Hadoop and MapReduce (AWS EMR).
  - ✓ Saves hassles of installations and managements of these environments
- SaaS:
  - ✓ Great help to organizations which requires specialized software's for big data like for social media analytics, feedback monitoring etc.
  - ✓ SaaS vendors provides out of the box solution for such common use cases

Cloud providers in Big Data market

- Amazon (Amazon Web Services AWS)
  - ✓ EC2
  - ✓ Elastic MapReduce
  - ✓ Kinesis
  - ✓ DynamoDB
  - ✓ Amazon S3
  - ✓ Redshift
- Google (Google Cloud Platform GCP)
  - ✓ Google Compute Engine
  - ✓ Google BigQuery
  - ✓ Google Prediction API
- Windows Azure
  - ✓ Azure PaaS cloud based on Windows and SQL
  - ✓ Windows Azure HD Insight

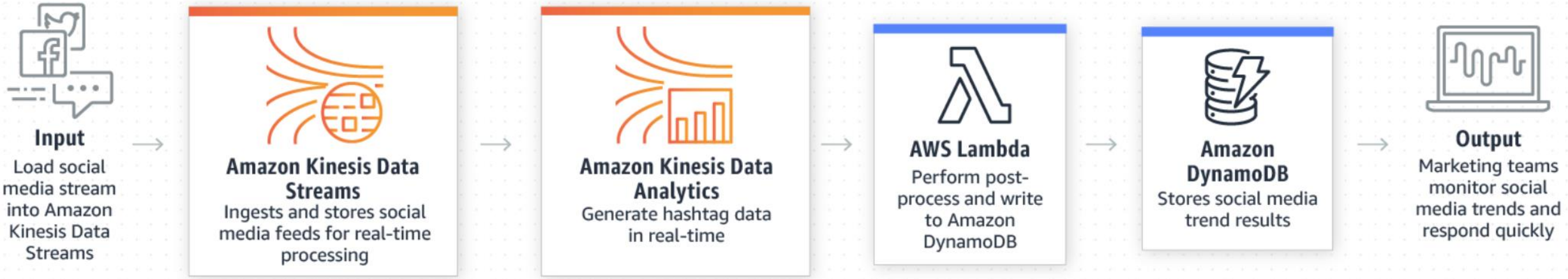
Case study: Nasdaq

- Operates financial exchanges
- AWS Redshift is the data warehouse with 5.5B rows/day with peak 14B/day in Oct 2014.
- A new DWH environment is setup using EMR and S3
  - give more teams access to gigantic data sets
  - cost efficiency
- Hadoop enables large and more varied historical data access at lower cost
- Why EMR and S3 ?
  - On demand set up of Hadoop clusters on Cloud to run short term projects.
  - Existing tech to move around TBs of data.
  - Storage on Cloud in S3 to scale to very large data sets.
- Why not HDFS and use S3 ?
  - Decouple data and compute because data size is disproportionately higher than compute frequency - so can't add Hadoop nodes just for data esp with replication factor
  - Similar to Netflix. EMRFS layer can enable compute nodes access data in S3



[Nasdaq case study]

Case study: Social media analytics



Ref: https://aws.amazon.com/kinesis/

Cloud: Pros and Cons

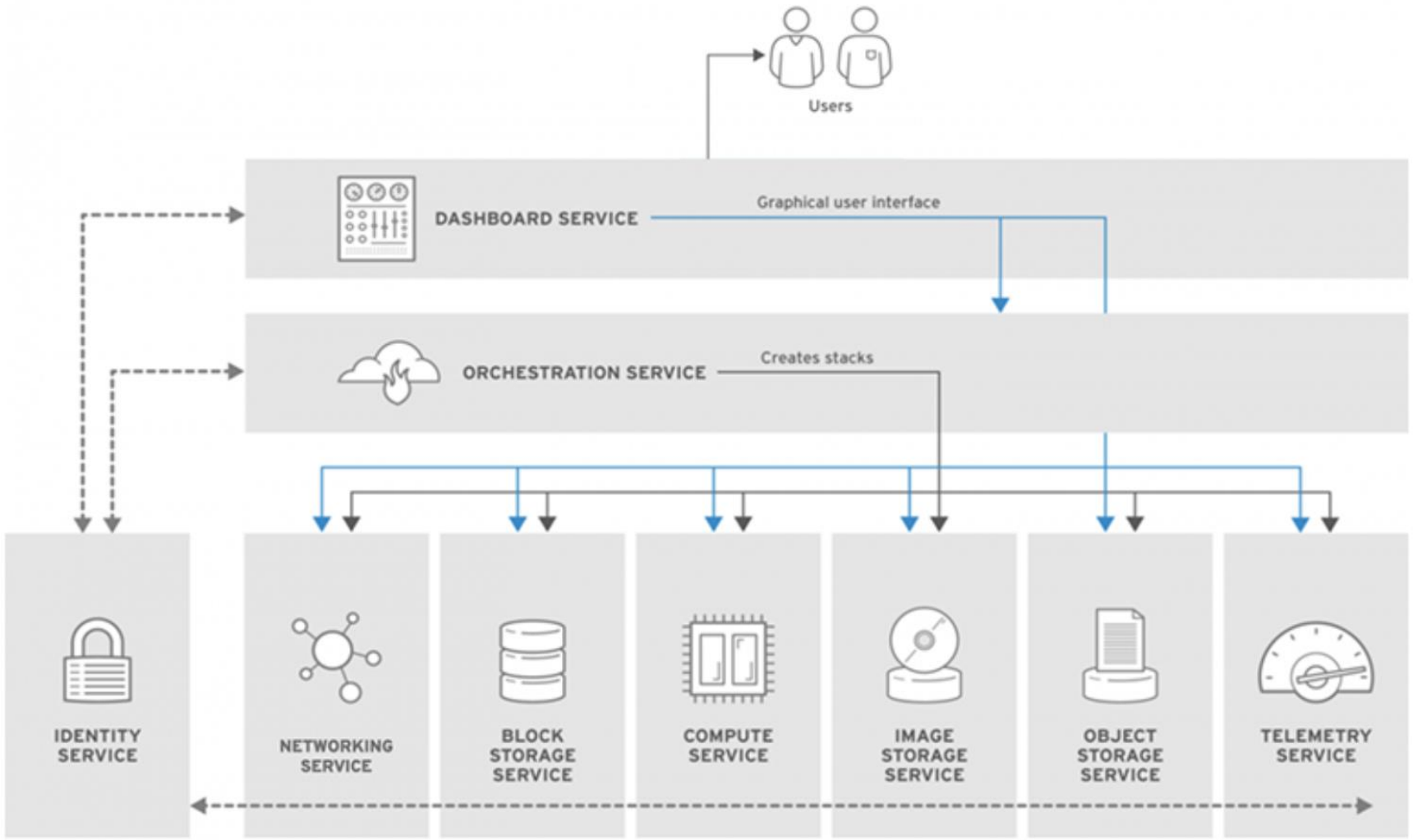
- Cost effective in specific cases, esp for short term projects, experiments, without getting into capital expense.
  - ✓ Be clear about usage
  - ✓ No in-house capability
- Reliability: Shift the onus of uptime on expert providers who can scale manpower, tools, systems to meet the demands
- Backup-recovery managed by experts
- Storage and compute scaling as required for short time periods. Stop guessing capacity.
- Use global DCs that are accessible from anywhere but without having to own one.

- Need to re-think security for data on Cloud or hybrid mode of operation where on-prem systems need to be connected
- Need to depend on external providers for infra / applications
- Vendor lock-in
- Not cost effective if not well planned and tracked for usage

Topics for today

- Cloud concepts
- Compute and network (Storage : next session)
- Best practices

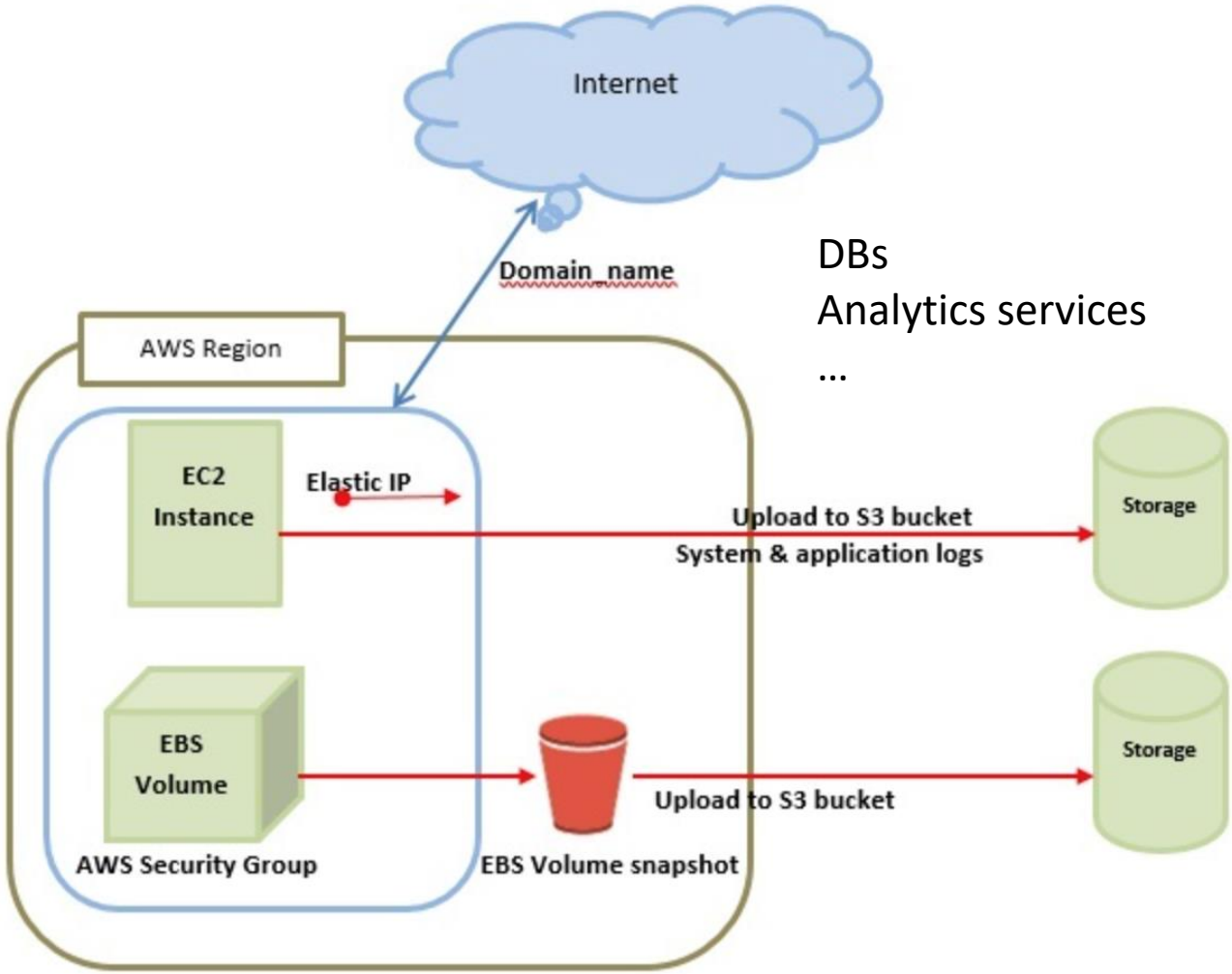
A basic IaaS architecture



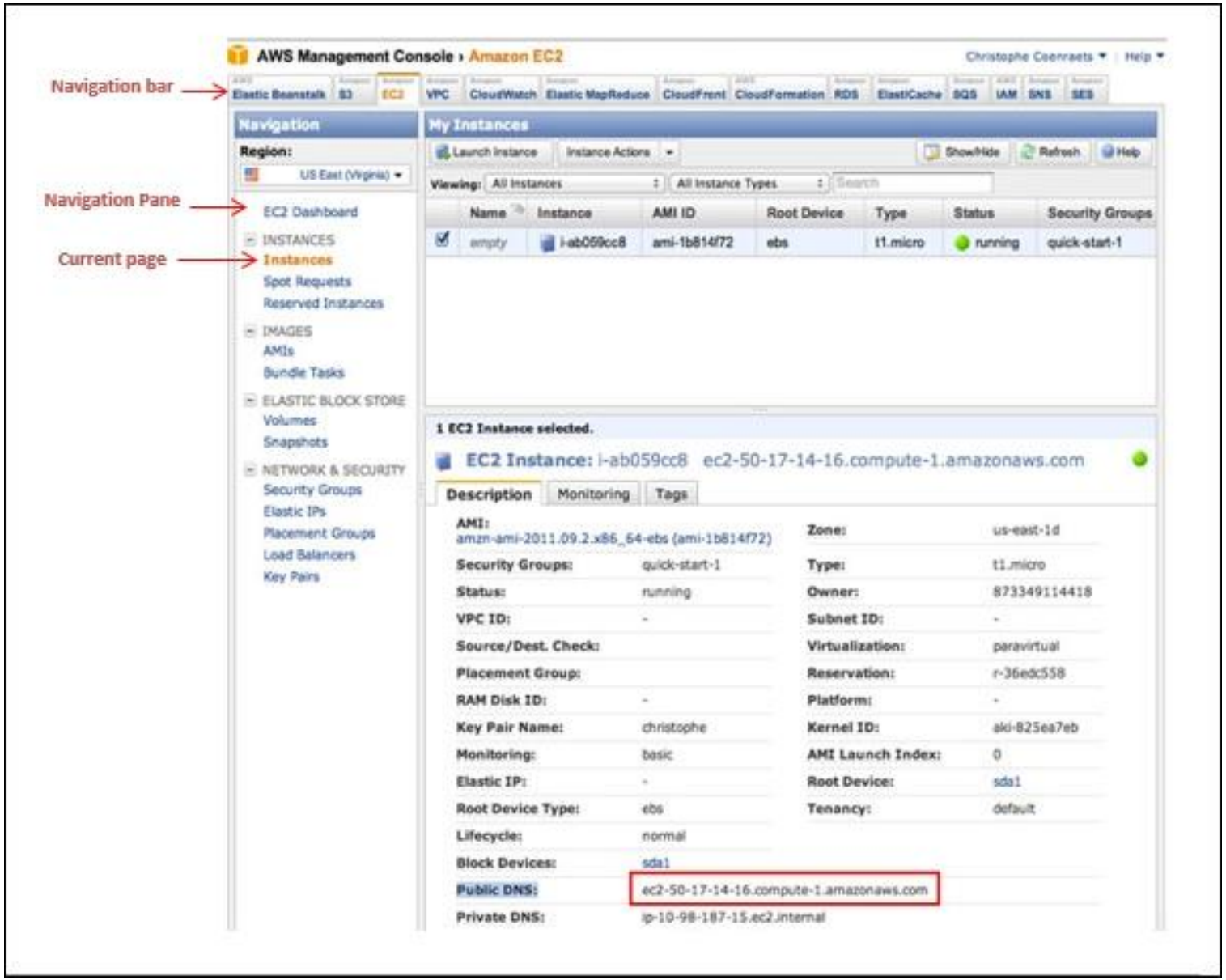
ref: [redhat.com OpenStack]

Example: AWS architecture

- Region: DCs are located in specific regions
- Availability zone: A region is split into AZs to isolate failures. AZs within region are connected with low-latency network and may be within or across DCs but with redundant power and networking.
- Compute instance: A machine, VM, container or function (FaaS). With DNS support, Elastic static IPs. Can be started from images.
- Attached volume: A compute instance is connected to block storage volumes for local FS.
- External storage: To serve as large remote raw data storage. E.g. An object storage like S3
- Security groups: Rules created to control access to the compute and storage instances.
- Load balancing: Service to spray load across multiple compute instances across or within AZ.
- Elastic scaling: Manual or automatically creation / deletion of compute instances.
- Backup/recovery: Snapshots can be taken to preserve state.

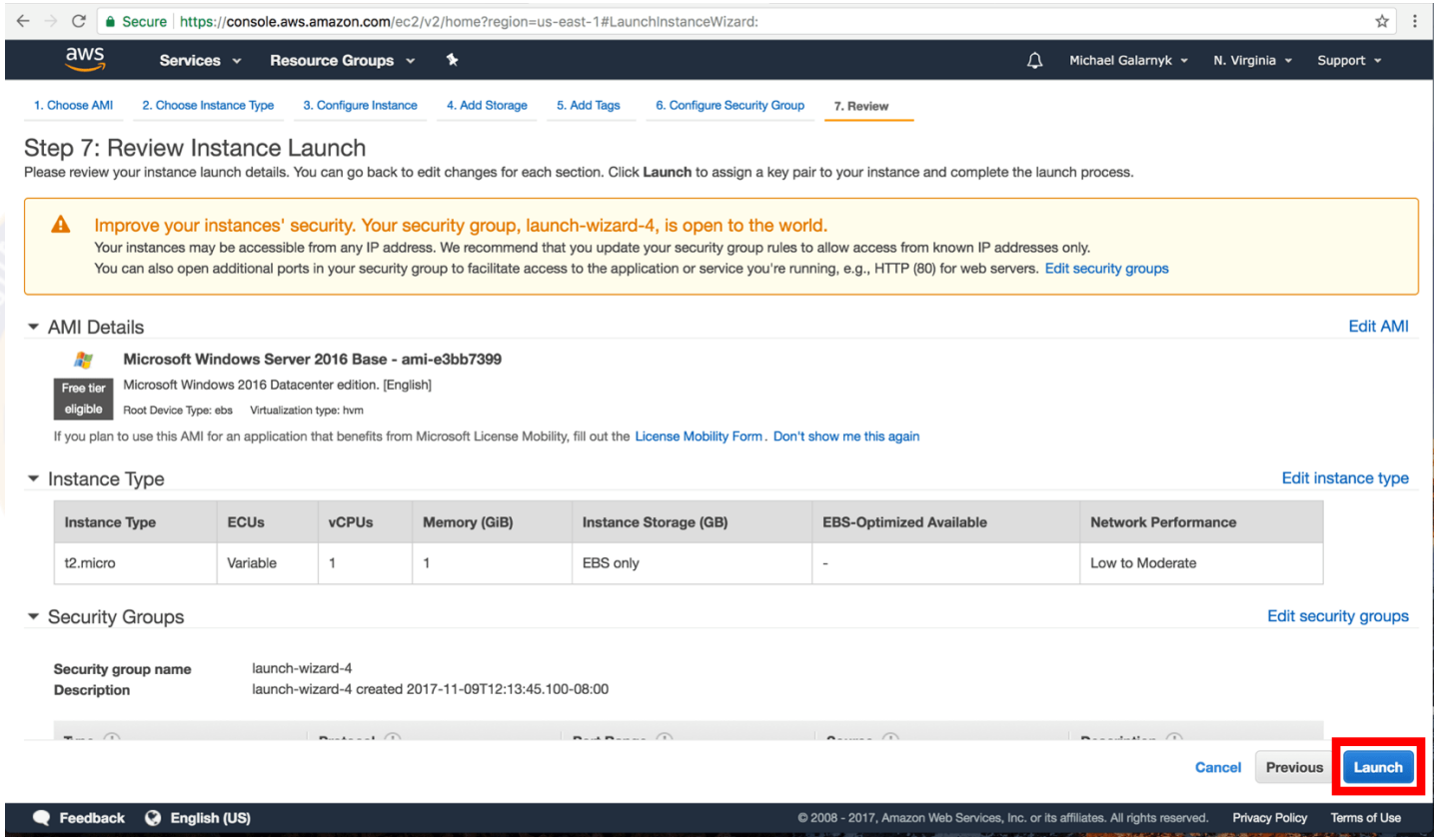


Centrally managed via consoles



Compute (1)

- Pick from a set of machine specs + OS / container / container cluster and launch within a region / AZ
- Configure security profile to allow/deny specific traffic - Firewall rules
- Can choose multiple instances across regions / AZ for fault tolerance
- Use elastic IP to quickly reassign instances in other AZ and stay connected
- Create new instances on the fly using Auto-scaling or manually add new instances to a Load Balancer

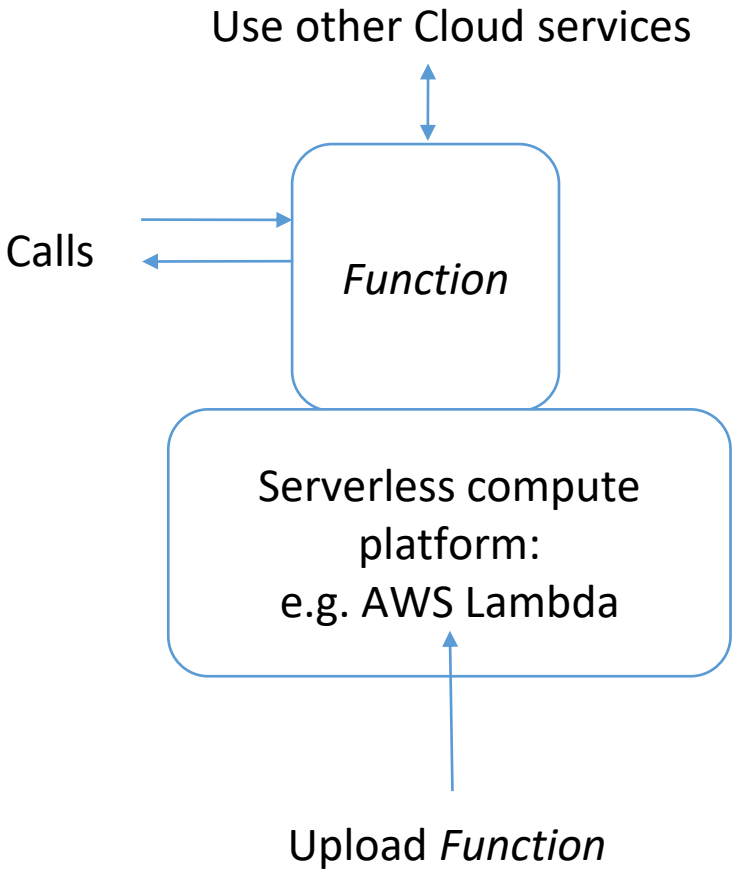


Compute (2): Serverless and FaaS

- Creating and running a Cloud application needs infrastructure - core code, libraries, integrations, event triggers, messaging, databases, app servers, web front ends, containers, networking etc.
  - ✓ Users still have to manage / configure some of this even in a Cloud platform
- How can we simplify this —
  - ✓ Simplify Infrastructure management and servers, esp in large scale deployments
    - Adopt serverless platforms
    - E.g. we can use a server less DB or a Kube cluster and leave management, scaling, config to service provider
  - ✓ Just focus on core code and micro-services that can run on a stateless compute platform
    - Adopt function-as-a-service and event-driven programming model

FaaS (1)

- FaaS computing - a new way to host code at function level where user is abstracted away from the underlying infrastructure needs to run the function
  - ✓ Pricing could be in terms of #requests, duration, max memory
  - ✓ e.g. AWS Lambda
- Stateless functions
- Pick from a set of runtimes - Node, Python, Ruby …
- Specify memory requirements, timeouts for function execution, where to get data from etc.
- FaaS platform has some limits on resources allocated for functions
- State has to be managed externally
- Not meant for complex functionalities
- Need to manage multiple functions

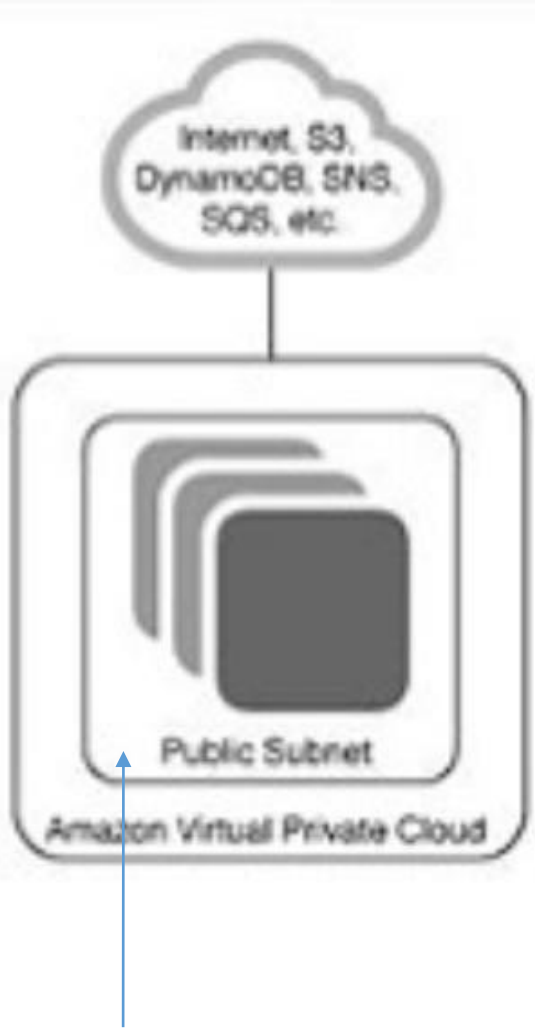


FaaS (2)

- Functions execute when triggered by events, e.g.
  - ✓ A REST API call to an API gateway
  - ✓ A file upload, e.g. an image upload to AWS S3 bucket
  - ✓ A timer expired, e.g. in AWS CloudWatch
  - ✓ A message arrival event in a queuing service
  - ✓ An alert or notification, e.g. in AWS SNS
  - ✓ …
- Functions can use storage and DB services, SNS etc. for input or output

Network (1): Virtual Private Cloud

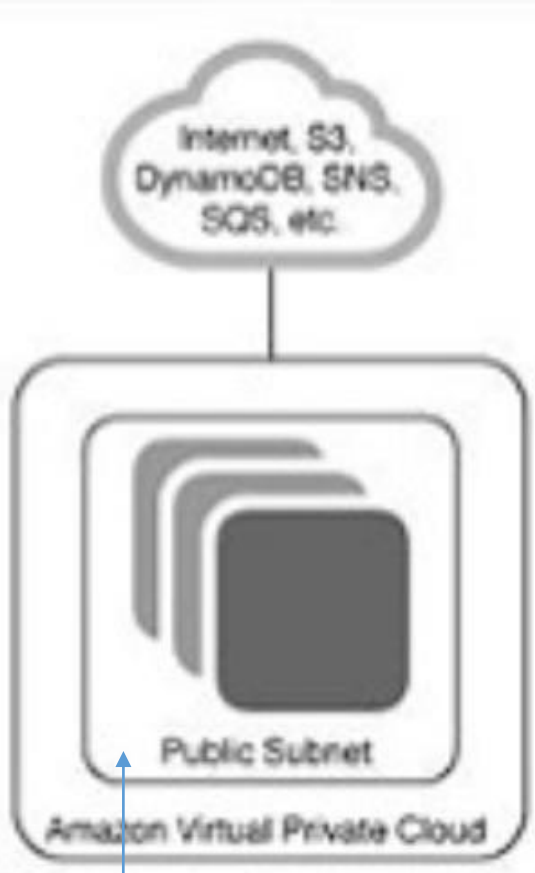
- Private space offered on Public Cloud
- Ideal for Hybrid Cloud deployment
- User has control over networking within VPC and create isolation
  - ✓ Specific subnets for public access
  - ✓ Specific subnets for private IPs not externally accessible
  - ✓ Subnets are within an AZ in AWS
  - ✓ Optional Network Address Translation - multiple devices can use same public IP to access internet
- Use VPN for external access
- Optional BGP routing for connecting VPC to other enterprise sites



Set up services within the subnets
Assign Elastic IPs

Network (2)

- DNS services may be provided for name to IP address translation for Internet-based applications
  - ✓ e.g. AWS Route53
- VLAN services may be provided for connecting on-prem network to Cloud / VPC
  - ✓ e.g. AWS Direct Connect
  - ✓ Typically needs additional constraints whether the Cloud location has support, Enterprise router has the right Internet routing protocols, physical link types etc.



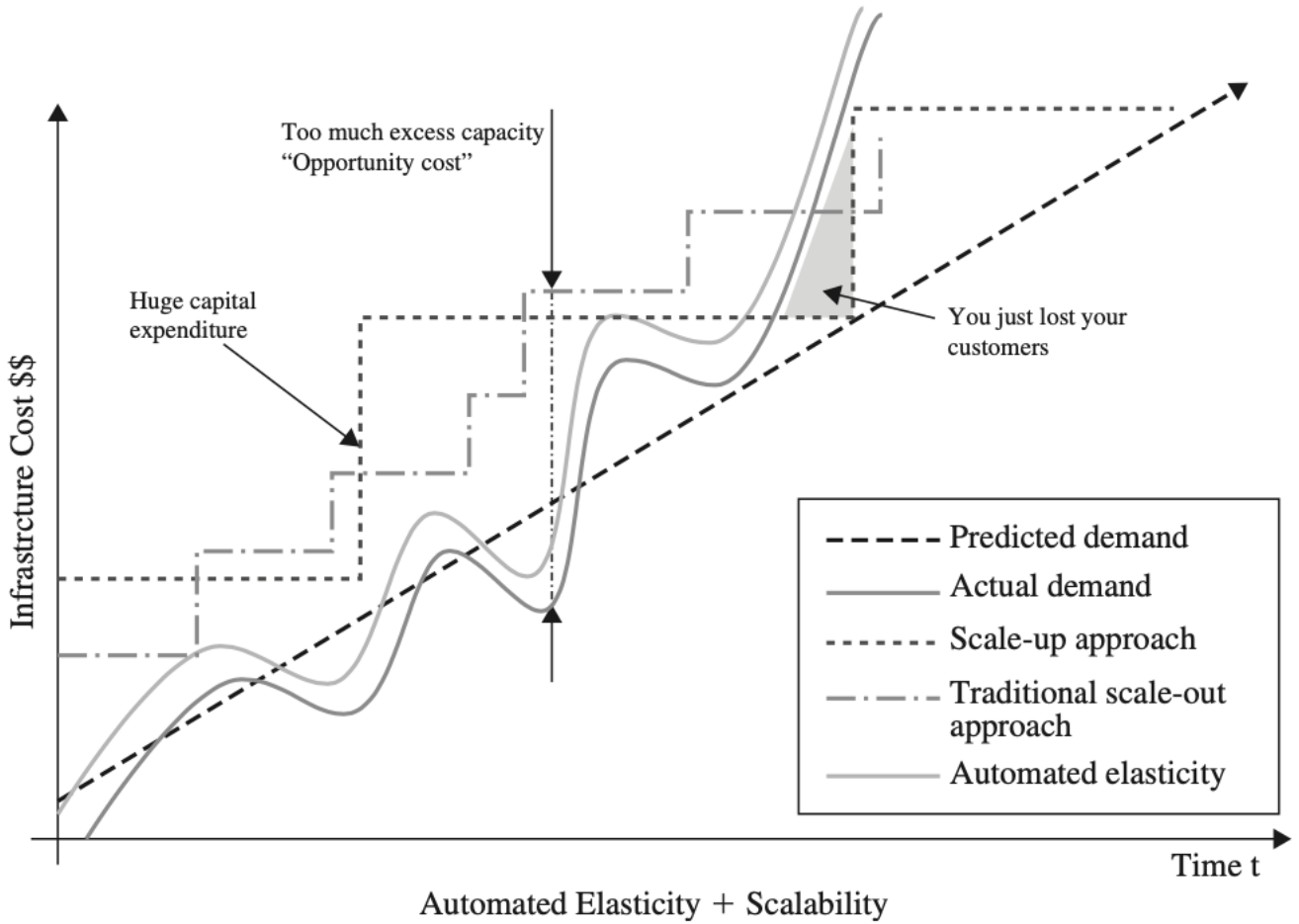
Set up services within the subnets
Assign Elastic IPs

Topics for today

- Cloud concepts
- Compute and network (Storage : next session)
- Best practices

Automated elasticity

- Applications can be provisioned with resources on-demand when they need it
- Re-think application design
  - ✓ Understand which application components can be elastic
  - ✓ What is the impact on overall application being elastic

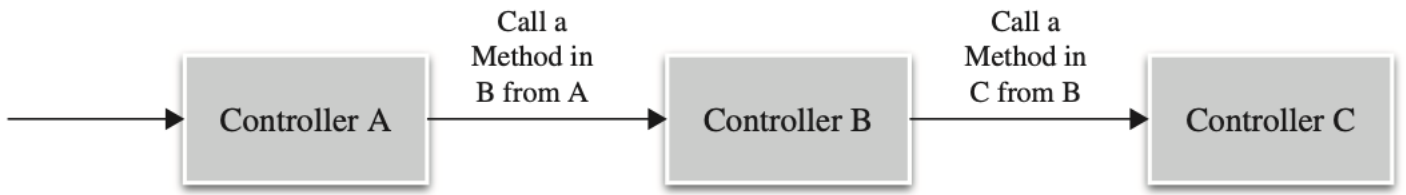


Design for failure (e.g. in AWS)

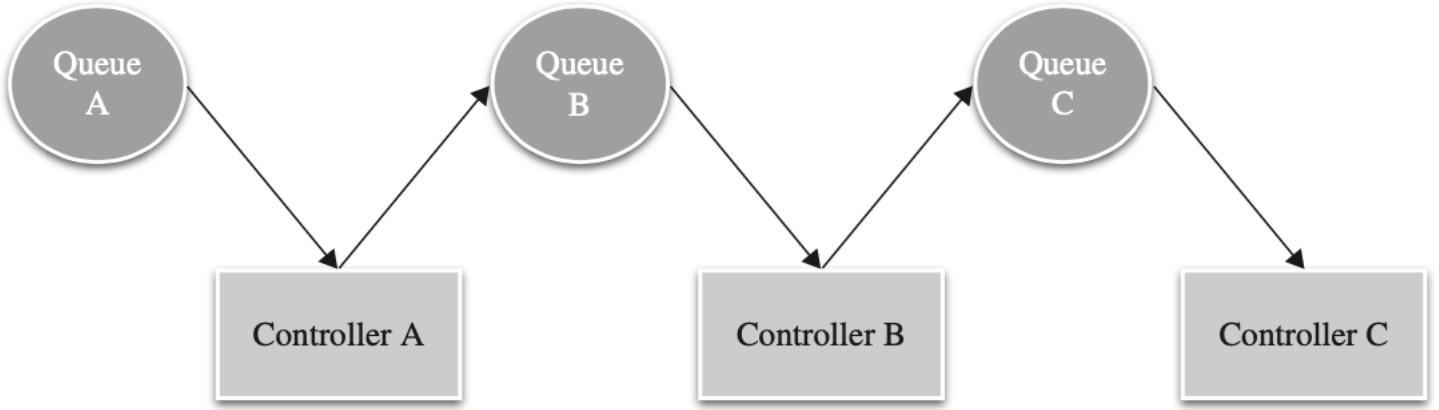
- Use Elastic IP, a static IP that can be mapped to different servers
- Use multiple AZ (logical DCs) during deployment - create replicas across AZ
- Maintain images (Amazon Machine Images) for replication / cloning, quick restoration
- Use monitoring services, e.g. AWS CloudWatch
- Create Auto Scaling groups to replace failed instances quickly
- Take snapshots and backup on external storage, e.g. S3

Decoupling application components

- Have application components interact using queues
- Stateless components as much as possible
- Store any session state in DB



Tight coupling (procedural programming)



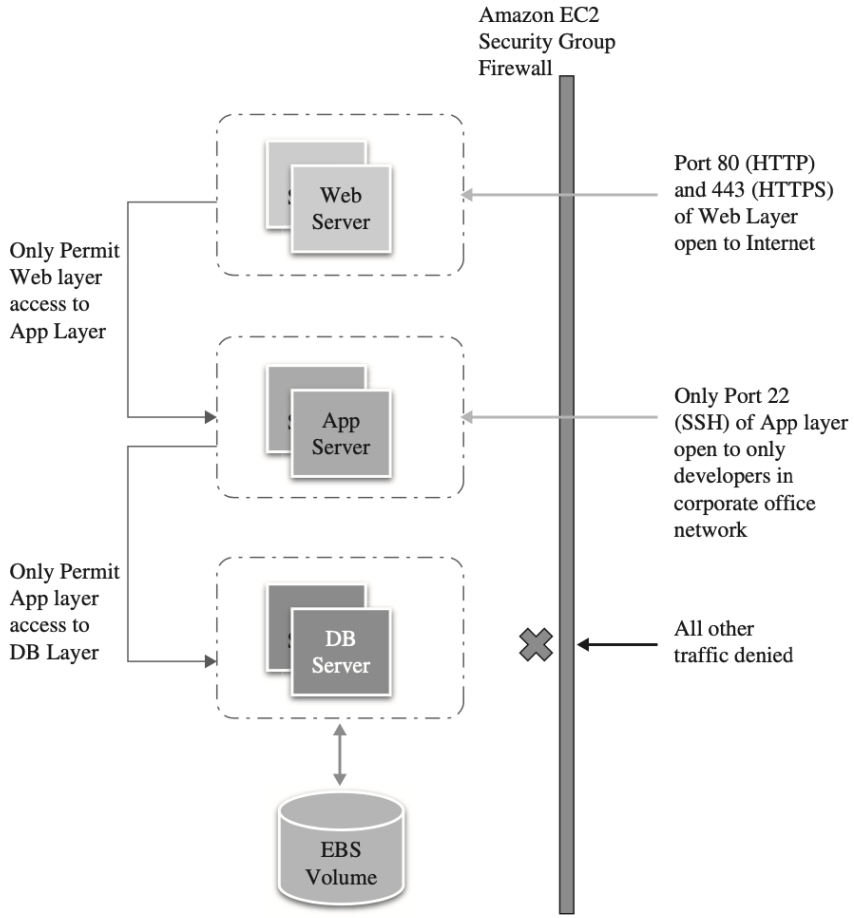
Loose coupling (independent phases using queues)

Automation of infrastructure

- Key to auto-scaling .. create auto-scaling groups for different clusters
- Monitor for resources (CPU, mem, I/O) and take actions like using machine images to launch new instances or sending notifications
- Store config data used for automation in a DB
- Build process : latest binaries go on a global external storage (e.g. AWS S3) - so any instance can spin up using latest build
- Open source config management tools: Chef, Puppet, Ansible etc.
- Build machine images with minimum OS for quick deployment. Configs and and user details can be passed on and after launch.
- Bootup from 1 or more attached block storage volumes, e.g. EBS volumes attached to EC2 instance
- App components should not depend on location, hardware, IP addresses because the image/binary can be moved anywhere on failure

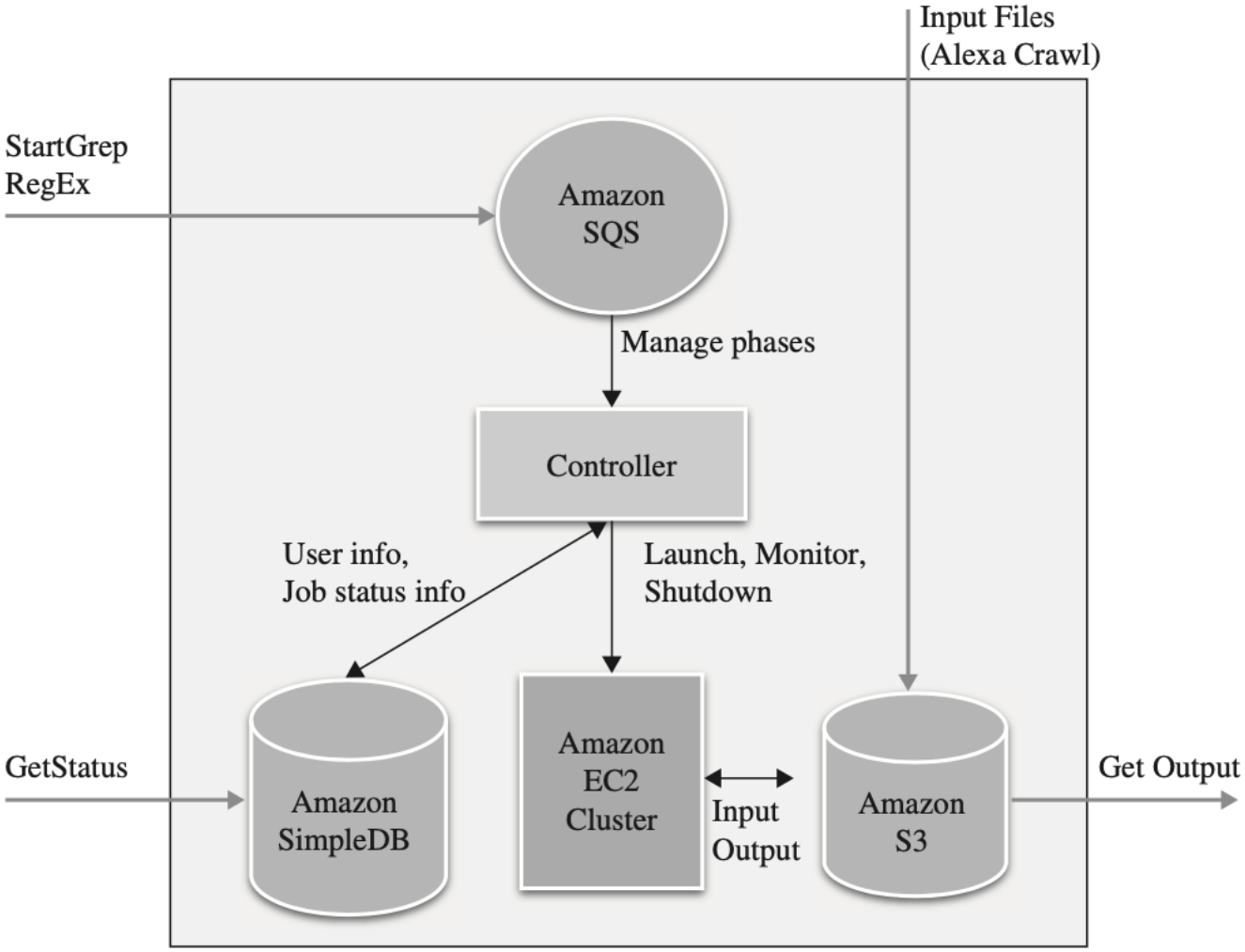
Security

- Traditional Enterprise perimeter security doesn't work on Cloud
- Security needed at every layer of application architecture
- Protect data in motion
  - ✓ Use SSL certificates and encryption in data movement
  - ✓ VPC to isolate within public cloud
  - ✓ Secure VPN for connecting on-prem to Cloud instances
- Protect data at rest
  - ✓ Can encrypt files or entire volumes
  - ✓ Key management is critical: Make sure keys are secure and not lost
  - ✓ Snapshot volumes for recovery, e.g. if compromised at some point of time
- Use the Identity and Access Management service (IAM) given by the Cloud provider
- Use instance specific virtual firewalls (e.g. AWS security groups) for compute instances, load balancers, DB etc.
- Use Network ACLs for subnet level access control
- Use Web App Firewall (WAF) along with application level load balancers to block application specific attacks



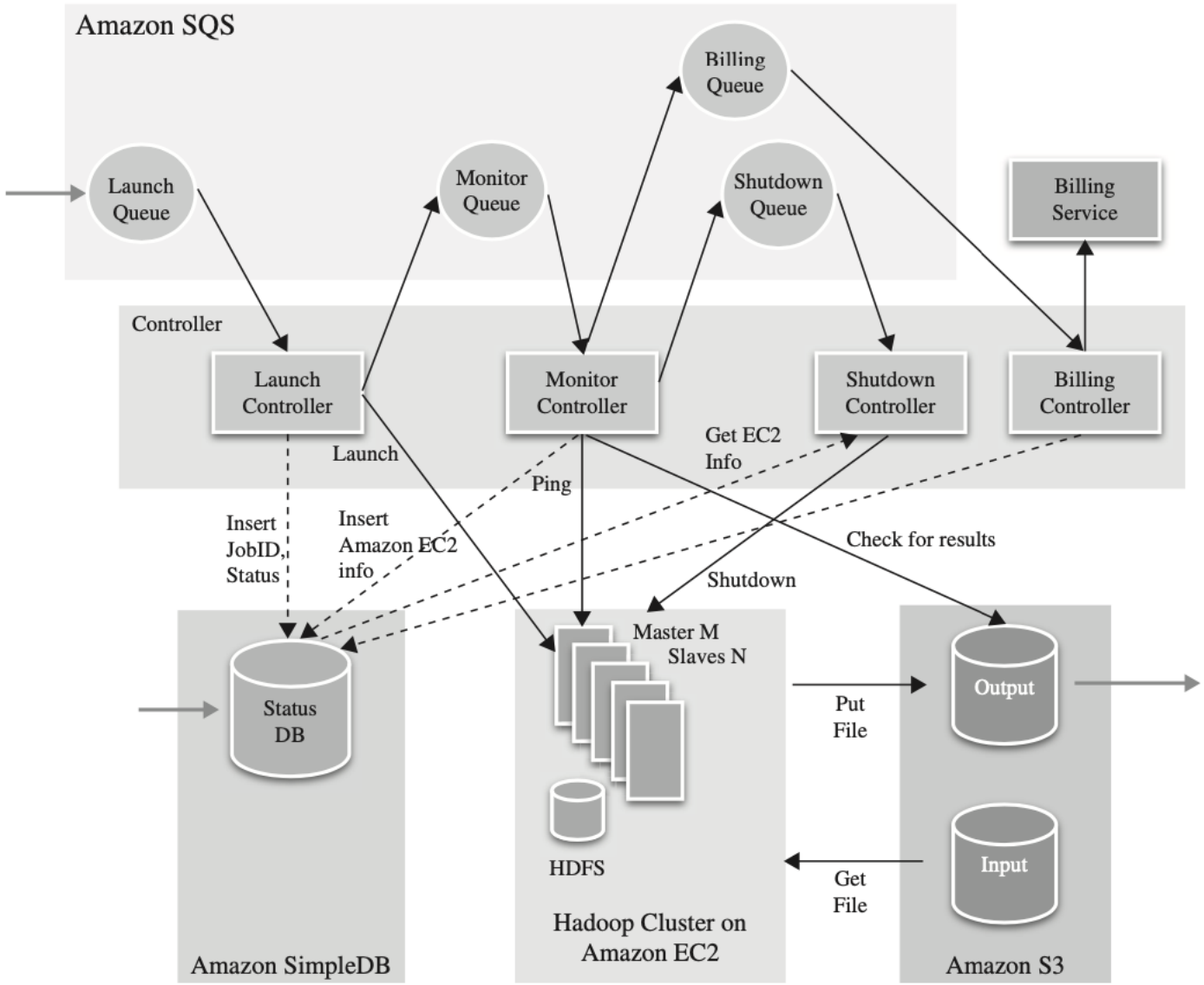
For interested readers: https://searchaws.techtarget.com/tip/AWS-security-groups-vs-network-ACLs-and-other-AWS-firewalls

Example application: GrepTheWeb



- Store crawled URLs on global storage
- Run a parallel search application for multiple users using Cloud best practices architecture
  - ✓ Queueing
  - ✓ Clustering
  - ✓ DB to manage meta-data
  - ✓ Monitor

Example application: GrepTheWeb



System design exercise

Design an Instagram as a Cloud native Big Data Application.
How would you go about creating a design ?
Identify the Big Data requirements - what kind of DBs/Storage would you need ?
May not be just one.
What is the data consistency desired ?
What compute instances would you have ?
Any load balancers, messaging systems ?
What about caching ?
How would you interact with users - Query? Notifications / Updates ?
What APIs would you have for users or internally ?

Summary

- Different Cloud models and types of services
- Key best practices
  - ✓ Horizontal scale out
  - ✓ Layered security, esp for data
  - ✓ Application componentization and message queues
  - ✓ Design for failures
  - ✓ Automate build, deployment and configuration

- Additional Reading: AWS Case study from - “Cloud Computing Principles and Paradigms” (Wiley Series on Parallel and Distributed Computing), R. Buyya et al, Chapter 18

Next Session:
Cloud storage