

Cloud Construction

Barry Denby

Griffith College Dublin

March 4, 2021

Cloud Construction

- ▶ In the previous lecture we encountered the different models of cloud computing
- ▶ In order for them to work we need a working cloud
- ▶ This lecture will cover how a cloud is put together (using Amazon AWS as an example)
- ▶ The lecture is heavily based on Dan Marinescu's Cloud Computing: Theory and Practice

Cloud Construction

- ▶ There is no one agreed method on how to construct a cloud
- ▶ We will observe and discuss a few different cloud designs
- ▶ Amazon AWS
- ▶ and Windows Azure

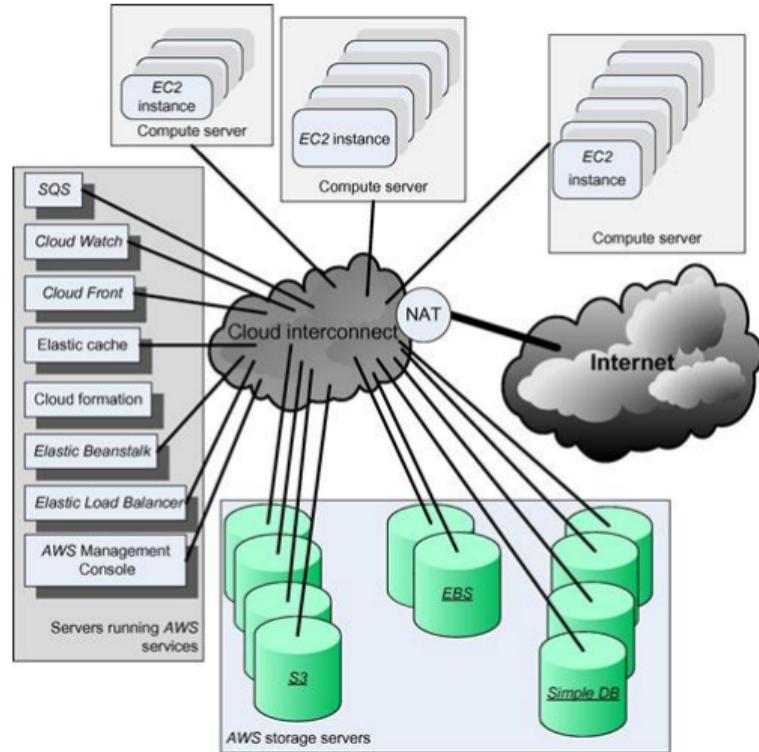
Cloud Construction

- ▶ At the most basic level a cloud is simply a large distributed system of computers
- ▶ Combining these machines enables the cloud to have a large set of resources
- ▶ Including CPU power, Storage, and Bandwidth
- ▶ However, in order for this to work a cloud designer must connect these components in a way that is
 - ▶ reliable
 - ▶ upgradable
 - ▶ fault tolerant
 - ▶ scalable

Cloud Construction

- ▶ On the following slide is a diagram detailing the construction of an Amazon AWS datacentre
- ▶ Amazon has a number of these datacentres around the world
- ▶ Including one in Dublin

Cloud Construction



- ▶ We will go through each component in turn and describe its purpose and reason for existing
- ▶ NAT (Network Address Translation)
 - ▶ required to convert from cloud network addresses to internet addresses and vice-versa
 - ▶ necessary due to exhaustion of the IPv4 address space
 - ▶ all incoming and outgoing traffic of the cloud must undergo translation
 - ▶ the network pipes connecting NAT to internet must have large amounts of bandwidth

- ▶ Cloud interconnect: An internal high speed network to connect all cloud components together
 - ▶ This Network is critical to cloud performance
 - ▶ Any slowdown here affects all instances and services in AWS
- ▶ AWS storage servers
 - ▶ All virtual images and data is stored here.
 - ▶ When a new EC2 instance is launched it will retrieve an image from here
 - ▶ Is split into three different storage types: S3, EBS, and SimpleDB

AWS storage systems

- ▶ S3: Simple Storage System, stores large objects
 - ▶ Three basic operations: read, write, delete
 - ▶ Uses REST and SOAP for data management
 - ▶ Supports HTTP and BitTorrent for object distribution
- ▶ EBS: Elastic Block Store: persistent block storage for AWS
 - ▶ Volumes range from 1GB to 1TB, cannot be shared
 - ▶ filesystem is user determined
 - ▶ used as additional storage for EC2 instances
- ▶ SimpleDB: NoSQL database that support store and query operations
 - ▶ can be queried through web requests
 - ▶ provides data redundancy and geographic distribution of data to ensure data safety

AWS

- ▶ Servers running AWS services
 - ▶ This is the core of AWS
 - ▶ Manages all components of the AWS cloud
 - ▶ Essentially is the distributed OS of AWS
 - ▶ Contains many components
- ▶ SQS: Simple Queue Service
 - ▶ Hosted message queue
 - ▶ Provides a means of message passing between instances
 - ▶ Particularly useful if machines are handling different jobs and need to communicate with each other
- ▶ Cloud Watch
 - ▶ The metrics and instrumentation component of AWS
 - ▶ How you will measure and optimise apps running on AWS

AWS

- ▶ Cloud Front: A content delivery service dependant on geographic location
- ▶ Elastic Cache: cache based in main memory of instances for temporary storage
 - ▶ Accessing main memory much faster than disk
 - ▶ Used to reduce response times
- ▶ Cloud Formation: used for automatic building of VMs with a software stack
 - ▶ Software stack is described as a JSON object
 - ▶ Can be parametrised to give customisation at run time
- ▶ Auto Scaling: Automatically adds and removes VMs based on demand
 - ▶ links into Cloud Watch metrics for this
 - ▶ Sets triggers and policies to automatically determine when to scale up and scale down

- ▶ Elastic Beanstalk: Auto scaling application that also uses Cloud Watch
 - ▶ Interacts with EC2, S3, Elastic Load Balancer and Auto Scaling
 - ▶ Is concerned with the wider set of resources than just VMs to scale up and down
- ▶ Elastic Load Balancer
 - ▶ Necessary for web applications that have multiple servers
 - ▶ Distributes the workload amongst many machines using one of a number of strategies
 - ▶ Round Robin: each request is forwarded to the next server in the list
 - ▶ Server with minimum load: whichever server has the least amount of load will get the request

AWS

- ▶ AWS Management Console: the user frontend that manages everything
 - ▶ Users set their instances running and interact with AWS through this
 - ▶ Provides a breakdown of all used resources thus far
- ▶ Compute Servers: compute units dedicated for running all EC2 virtual machines
 - ▶ Has a thin Operating System that is dedicated to only running and managing virtual machines

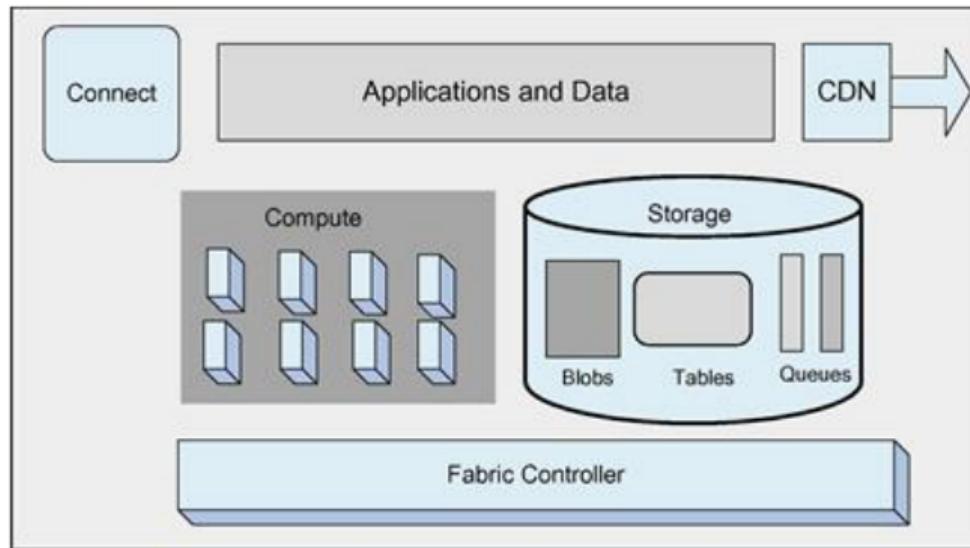
Things to not about the AWS structure

- ▶ There is a clear distinction between the different areas of the design. Each machine in AWS has a purpose
- ▶ Network bandwidth is the limiting factor in performance
 - ▶ Communication should be kept to a minimum
 - ▶ As Network speeds have not evolved at the same pace as processor technology
 - ▶ Principle of Data Locality should be exploited whenever possible
 - ▶ i.e. keep data transfers to a minimum and keep all required data as close to the CPU as possible

Windows Azure Services

- ▶ Windows Azure Services is similar to Google App Engine and provides a PaaS service to developers
- ▶ Windows Azure is the distributed OS that manages Microsoft's cloud cluster
- ▶ Installed on every node in the cloud
- ▶ Coordinates together to produce what appears to be a single system
- ▶ System architecture is shown on next slide

Windows Azure Services



Windows Azure Services

- ▶ Note that unlike AWS no network structure is presented
- ▶ The idea being that at the PaaS level the developer should not need to know about this
- ▶ All resources are virtualised and hidden from the developer
- ▶ There is no indication as to where these resources are coming from

Windows Azure Services

- ▶ In Azure services there are three core components
- ▶ Storage: composed of Blobs, Tables, and Queues
- ▶ Compute: completely virtualised and no idea where it is coming from in the cloud
- ▶ Fabric Controller: ties everything together
 - ▶ Responsible for deploying, managing, and monitoring applications
 - ▶ Handles scaling, load balancing, memory management etc all automatically

Windows Azure Services

- ▶ All applications are written as a series of roles
 - ▶ Roles are application types that are commonly used in windows azure
 - ▶ Thus azure has optimisations in place for applications written to these roles
- ▶ The two most common role types are: web roles and worker roles
 - ▶ Web roles: used to replicate web services and designed to handle multiple requests as efficiently as possible
 - ▶ Worker roles: used to run windows based code for automating and back end processing

Windows Azure Services

- ▶ There are three main types of storage in Azure
- ▶ Blobs: Binary Large OBjects: used to store arbitrary data that is not easy to store in a DB. Each blob can be upto 1TB in size
- ▶ Tables: standard SQL relational databases for storing data
- ▶ Queues: provide a means of asynchronous message passing between web and worker roles

Cloud Design Decisions

- ▶ From the models of cloud that we have seen there are a few important properties of cloud design we can discern.
- ▶ Compute resources should be homogeneous
 - ▶ As all resources in the cloud are virtualised, the user will not know where their applications are hosted
 - ▶ Heterogeneous resources would permit some applications to compute quicker than others
 - ▶ will show as faster/slower compute and response times
 - ▶ presents an NP-complete problem of allocating computations in heterogeneous network
 - ▶ homologating computing power avoids this problem to a simple issue of load balancing
 - ▶ clouds will try to be completely homogeneous but unavoidable due to upgrades

Cloud Design Decisions

- ▶ Network Design: the network should be homogeneous in design.
- ▶ Fast links must also be present due to the amount of data that must be transferred
- ▶ TCP/IPs routing rules will affect the path packets take on the network.
- ▶ Thus it is important all routes have equal speed to simplify the problem
- ▶ As communication is the slowest link in the cloud, any improvements made here will benefit all applications

Cloud Design Decisions

- ▶ That said there are generally two networks in a cloud design
- ▶ The cloud interconnect we discussed earlier
- ▶ We also have a SAN (Storage Area Network) for connecting the storage nodes together
 - ▶ Storage demands very high bandwidth due to data replication
 - ▶ Data access can come from multiple nodes at the same time thus requiring very high bandwidth

Cloud Design Decisions

- ▶ Storage must also be reliable
- ▶ Application data is priceless and irreplaceable
- ▶ Generally some form of RAID array or PFS will be used as primary storage
 - ▶ Thus if a drive fails it can be replaced
 - ▶ RAID will reconstruct itself with the new drive
- ▶ Data should be replicated off site
 - ▶ If all backup copies of data are stored in a single data center then you lose all data should it be destroyed