CIS 455/555: Internet and Web Systems

The Cloud

November 15, 2021
Plan for today

- Cloud computing
  - Utility computing model
  - Brief overview of EC2
  - Brief overview of EBS
Utility computing: Power plant analogy

- It used to be that everyone had their own power source
  - Challenges are similar to the cluster: Needs large up-front investment, expertise to operate, difficult to scale up/down...
Then people started to build large, centralized power plants with very large capacity...
Power plants are connected to customers by a network

Usage is metered, and everyone (basically) pays only for what they actually use
Why is this a good thing?

**Electricity**

- **Economies of scale**
  - Cheaper to run one big power plant than many small ones

- **Statistical multiplexing**
  - High utilization!

- **No up-front commitment**
  - No investment in generator; pay-as-you-go model

- **Scalability**
  - Thousands of kilowatts available on demand; add more within seconds

**Computing**

- Cheaper to run one big data center than many small ones

- High utilization!

- No investment in data center; pay-as-you-go model

- Thousands of computers available on demand; add more within seconds
According to NIST:

Cloud computing is a model for enabling 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.

Essential characteristics:
- On-demand self service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service
History: Becoming a cloud provider

- Early 2000s: Phenomenal growth of web services
  - Many large Internet companies deploy huge data centers, develop scalable software infrastructure to run them
  - Due to economies of scale, these companies were now able to run computation very cheaply
  - What else can we do with this? Lease it!

<table>
<thead>
<tr>
<th>Technology</th>
<th>Cost in medium DC (~1,000 servers)</th>
<th>Cost in large DC (~50,000 servers)</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>$95 per Mbit/sec/month</td>
<td>$13 per Mbit/sec/month</td>
<td>7.1</td>
</tr>
<tr>
<td>Storage</td>
<td>$2.20 per GByte/month</td>
<td>$0.40 per GByte/month</td>
<td>5.7</td>
</tr>
<tr>
<td>Administration</td>
<td>~140 servers/admin</td>
<td>&gt;1,000 servers/admin</td>
<td>7.1</td>
</tr>
</tbody>
</table>
Today

- An evolution of services, at multiple levels

- Most of the cloud providers develop their services for internal and external use
  - Amortize development costs
  - Many of the technologies are available via open source

- ... Though Google is known to have many proprietary services not available through their cloud
Plan for today

- Cloud computing
  - Utility computing model
  - Brief overview of EC2
  - Brief overview of EBS
What Are Cloud Services?

- Sometimes it’s useful to divide them into layers:
  - Software as a Service (SaaS)
  - Platform as a Service (PaaS)
  - Infrastructure as a Service (IaaS)

- But for our purposes it’s more useful to drill down a bit more...
Cloud Providers are *Gradually Setting on a Standard Set of Services*

<table>
<thead>
<tr>
<th>Google Cloud Platform</th>
<th>Amazon Web Services</th>
<th>Microsoft Azure</th>
<th>Oracle Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Compute Engine</td>
<td>Amazon EC2</td>
<td>Azure Virtual Machines</td>
<td>Oracle Cloud Infra OCI</td>
</tr>
<tr>
<td>Google App Engine</td>
<td>AWS Elastic Beanstalk</td>
<td>Azure App Services</td>
<td>Oracle Application Container</td>
</tr>
<tr>
<td>Google Kubernetes Engine</td>
<td>Amazon Elastic Kubernetes Service</td>
<td>Azure Kubernetes Service</td>
<td>Oracle Kubernetes Service</td>
</tr>
<tr>
<td>Google Cloud Bigtable</td>
<td>Amazon DynamoDB</td>
<td>Azure Cosmos DB</td>
<td>Oracle NoSQL Database</td>
</tr>
<tr>
<td>Google BigQuery</td>
<td>Amazon Redshift</td>
<td>Azure Synapse Analytics</td>
<td>Oracle Autonomous Data Warehouse</td>
</tr>
<tr>
<td>Google Cloud Functions</td>
<td>AWS Lambda</td>
<td>Azure Functions</td>
<td>Oracle Cloud Fn</td>
</tr>
<tr>
<td>Google Cloud Datastore</td>
<td>Amazon DynamoDB</td>
<td>Azure Cosmos DB</td>
<td>Oracle NoSQL Database</td>
</tr>
<tr>
<td>Google Cloud Storage</td>
<td>Amazon S3</td>
<td>Azure Blob Storage</td>
<td>Oracle Cloud Storage OCI</td>
</tr>
</tbody>
</table>
For this Course

- You are free to sign up for Microsoft Azure, Google Cloud, or AWS
  - Each usually offers a “getting started” or “free” tier

- But we will be giving out login info for AWS@Penn
  - Penn-sponsored AWS cycles, with much greater capabilities than AWS Educate
Important Amazon Services

Amazon Web Services (AWS) provides many different services, including:

- Elastic Compute Cloud (EC2)
  Virtual machines for running custom software
- Simple Storage Service (S3)
  Simple key-value store, accessible as a web service
- DynamoDB
  Distributed NoSQL database, one of several in AWS
- Relational Database Service (RDS)
  Hosted SQL database
- Elastic MapReduce
  Scalable MapReduce or Spark computation
- Mechanical Turk (MTurk)
  A 'marketplace for work'
- ...
Credentials in AWS

- Protect your credits!

- Two main types of "credentials":
  - **keypairs**, which allow you to log into machines
  - **Identity and Access Management (IAM) users**, special users with very limited permissions

![Add user interface](image)
Note Down, in a Safe Place, the Access and Secret Keys

These will be used by your REST calls from your server

- User does not have console management access

<table>
<thead>
<tr>
<th>Console password</th>
<th>Assigned MFA device</th>
<th>Signing certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled</td>
<td>Not assigned</td>
<td>None</td>
</tr>
</tbody>
</table>

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. Learn more

Create access key

<table>
<thead>
<tr>
<th>Access key ID</th>
<th>Created</th>
<th>Last used</th>
<th>Status</th>
</tr>
</thead>
</table>

© 2021 A. Haeberlen, Z. Ives, V. Liu
How Do You *Use* AWS?

- For certain services:
  - You can remotely log in, e.g., via ssh
  - Or you can control via the AWS Console

- But more commonly: an application or service makes requests of AWS
  - Can be called from the AWS Command Line Interface
  - Or a piece of Java code using the AWS Java SDK
  - ...
  - etc
What is Amazon EC2?

- **Infrastructure-as-a-Service (IaaS)**
  - You can rent various types of virtual machines by the hour
  - In your VMs, you can run your own (Linux/Windows) programs
    - Examples: Web server, search engine, movie renderer, ...
  - Different machine types
    - Current/previous generation
    - Memory/Storage/Compute/GPU
    - Micro instances

---

<table>
<thead>
<tr>
<th>Region</th>
<th>vCPU</th>
<th>ECU</th>
<th>Memory (GiB)</th>
<th>Instance Storage (GiB)</th>
<th>Linux/Unix Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (Ohio)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>General Purpose - Current Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1.medium</td>
</tr>
<tr>
<td>a1.large</td>
</tr>
<tr>
<td>a1.xlarge</td>
</tr>
<tr>
<td>a1.2xlarge</td>
</tr>
<tr>
<td>a1.4xlarge</td>
</tr>
<tr>
<td>a1.metal</td>
</tr>
<tr>
<td>t4g.mano</td>
</tr>
<tr>
<td>t4g.micro</td>
</tr>
<tr>
<td>t4g.metal</td>
</tr>
<tr>
<td>t4g.medium</td>
</tr>
<tr>
<td>t4g.large</td>
</tr>
<tr>
<td>t4g.2xlarge</td>
</tr>
<tr>
<td>t2.nano</td>
</tr>
<tr>
<td>t2.micro</td>
</tr>
<tr>
<td>t2.small</td>
</tr>
<tr>
<td>t2.medium</td>
</tr>
<tr>
<td>t2.large</td>
</tr>
<tr>
<td>t2.xlarge</td>
</tr>
<tr>
<td>t3.medium</td>
</tr>
<tr>
<td>t3.large</td>
</tr>
<tr>
<td>t3.xlarge</td>
</tr>
<tr>
<td>t3.2xlarge</td>
</tr>
<tr>
<td>t3.micro</td>
</tr>
<tr>
<td>t3.small</td>
</tr>
<tr>
<td>t3.medium</td>
</tr>
<tr>
<td>t3.large</td>
</tr>
<tr>
<td>t3.xlarge</td>
</tr>
<tr>
<td>t3.2xlarge</td>
</tr>
<tr>
<td>m3.medium</td>
</tr>
<tr>
<td>m3.large</td>
</tr>
<tr>
<td>m3.xlarge</td>
</tr>
<tr>
<td>m3.2xlarge</td>
</tr>
<tr>
<td>m3.4xlarge</td>
</tr>
<tr>
<td>m3.16xlarge</td>
</tr>
<tr>
<td>m3.256xlarge</td>
</tr>
</tbody>
</table>

Demo

- Logging into AWS Management Console
- Launching an instance
- Contacting the instance via ssh
- Terminating an instance
- Using price calculators:

- Have a look at the AWS Getting Started guide
  - https://aws.amazon.com/ec2/getting-started/
Using EC2

Launch instance

To get started, launch an Amazon EC2 instance.

Launch instance

Note: Your instances will launch in the US East (N. Virginia) region.


Root device type: ebs  Virtualization type: hvm  EBS-Enabled

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and needs.

Filter by: All instance types  Current generation  Show/Hide Columns

Currently selected: t2.micro (Variable ECU, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GB memory, EBS only)

Select an existing key pair or create a new key pair

A key pair consists of a public key that AWS stores, and a private key file that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

Key pair name

You have to download the private key file (*.pem file) before you can continue. Store it in a secure and accessible location. You will not be able to download the file again after it's created.

Cancel  Launch Instances
The EC2 Dashboard

(base) keyser:~ liuv$ ssh -i "cis455.pem" ubuntu@ec2-52-23-201-47.compute-1.amazonaws.com
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1045-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:     https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of Sun Oct 31 01:46:48 UTC 2021

System load: 0.62 Processes: 101
Usage of /: 16.6% of 7.69GB Users logged in: 0
Memory usage: 46% IPv4 address for eth0: 172.31.52.201
Swap usage: 0%

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Oct 31 01:46:08 2021 from 69.249.30.117
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-52-201:~$
And *Always*...

- Shut down if you aren’t going to use the VM for more than an hour
Oh no - where has my data gone?

- Not all EC2 instances have persistent storage
  - Data survives stops & reboots, but not termination

If you store data on the virtual hard disk of your instance and the instance fails or you terminate it, your data WILL be lost!

- So where should I put persistent data?
  - Elastic Block Store (EBS)
    You can ask for additional EBS disk space while you are creating the EC2 instance!
  - Caution: Default behavior: delete the EBS upon termination of the instance!
Amazon Machine Images

- When I launch an instance, what software will be installed on it?
  - Software is taken from an Amazon Machine Image (AMI)
  - Selected when you launch an instance
  - Essentially a file system that contains the operating system, applications, and potentially other data
  - Lives in S3

- How do I get an AMI?
  - Amazon provides several generic ones, e.g., Amazon Linux, Fedora Core, Ubuntu, Windows Server, ...
  - You could make your own
    - You can even run your own custom kernel, with some restrictions
Security Groups

- Basically, a set of firewall rules
  - Can be applied to groups of EC2 instances
  - Each rule specifies a protocol, port numbers, etc...
  - Only traffic matching one of the rules is allowed through
- Sometimes need to explicitly open ports
Regions and Availability Zones

- Where exactly does my instance run?
  - No easy way to find out - Amazon does not say

- Instances can be assigned to **regions**
  - Currently 24 available:
    - US East (N. Virginia, Ohio), US West (N. California, Oregon), EU (Ireland, Frankfurt, London, Paris, Stockholm), Asia/Pacific (Hong Kong, Mumbai, Osaka, Seoul, Singapore, Sydney, Tokyo), Canada, China (Beijing, Ningxia), South America (Sao Paulo), Middle East (Bahrain) AWS GovCloud (East+West)
  - Important, e.g., for reducing latency to customers

- Instances can be assigned to **availability zones**
  - Purpose: Avoid correlated faults
  - Several availability zones within each region
Network pricing

AWS charges for network traffic
- Price depends on source and destination of traffic
- Free within EC2 and other AWS services in same region (e.g., S3)
- Remember: ISPs are typically charged for upstream traffic
Instance types

- So far: **On-demand instances**
- Also available: **Reserved instances**
  - One-time reservation fee to purchase for 1 or 3 years
  - Usage still billed by the hour, but at a considerable discount
- Also available: **Spot instances**
  - Spot market: Can bid for available capacity
  - Instance continues until terminated or price rises above bid

Service Commitment

AWS will use commercially reasonable efforts to make the Included Products and Services each available with a Monthly Uptime Percentage (defined below) of at least 99.99%, in each case during any monthly billing cycle (the “Service Commitment”). In the event any of the Included Products and Services do not meet the Service Commitment, you will be eligible to receive a Service Credit as described below.

Definitions

- "Monthly Uptime Percentage" is calculated by subtracting from 100% the percentage of minutes during the month in which any of the Included Products and Services, as applicable, was in the state of “Region Unavailable.” Monthly Uptime Percentage measurements exclude downtime resulting directly or indirectly from any Amazon Compute Services SLA Exclusion (defined below).
- “Availability Zone” and “AZ” mean an isolated location within a region identified by a letter identifier following the region code (e.g., us-west-1a).
- “Region Unavailable” and “Region Unavailability” mean:
  - For Regions with only one AZ, when that AZ and one AZ in any other Region, in which you are running an instance or task (one or more containers), as applicable, are concurrently “Unavailable” to you.
  - For all other Regions, when more than one AZ within the same Region, in which you are running an instance or task (one or more containers), as applicable, are concurrently “Unavailable” to you.
- "Unavailable" and "Unavailability" mean:
  - For Amazon EC2, Amazon ECS, or Amazon Fargate, when all of your running instances or running tasks, as applicable, have no external connectivity.
  - For Amazon EBS, when all of your attached volumes perform zero read write IO, with pending IO in the queue.
- A “Service Credit” is a dollar credit, calculated as set forth below, that we may credit back to an eligible account.

Service Commitments and Service Credits

Service Credits are calculated as a percentage of the total charges paid by you (excluding one-time payments such as upfront payments made for Reserved Instances) for either Amazon EC2 or Amazon EBS (whichever was Unavailable, or both if both were Unavailable) in the Region affected for the monthly billing cycle in which the Region Unavailability occurred in accordance with the schedule below.

<table>
<thead>
<tr>
<th>Monthly Uptime Percentage</th>
<th>Service Credit Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than 99.99% but equal to or greater than 99.0%</td>
<td>10%</td>
</tr>
<tr>
<td>Less than 99.0%</td>
<td>30%</td>
</tr>
</tbody>
</table>

http://aws.amazon.com/ec2-sla/ (4/2/2018 version; excerpt)
AWS and your project

What cloud resources are useful where?

**Crawler:** EC2 is a good fit
- Combine HW2 crawler with HW3 cross-machine parallelism
- Can run several instances on EC2 nodes
- Try to keep data structures in instance-local storage during the crawl; afterwards, download copy or keep in S3

**Indexer/PageRank:** EC2 or Elastic MapReduce
- Indexing and PageRank are great MapReduce jobs!
- You can execute them on EC2 instances, using your HW3 framework, or you can run native Hadoop jobs on EMR
- Store the index in BDB, DynamoDB, or RDS

**Frontend:** EC2
- The frontend can run on an EC2 instance, using your HW1 code with custom routes for query processing