Cloud Computing

an overview: 2009 to 2019

Andy Huang & Stephanie Doan

Overview

1. What is cloud computing?
2. Cloud Computing in 2009
3. Cloud Computing Today

Before Cloud Computing

- In the past, small companies ran server room or provisioned a datacenter to run their app
- Must need a lot of money at the start -- small developers and startups don’t have that
- Lots of resources went to waste because some hardware would go unused at times

Cloud Computing in 2009

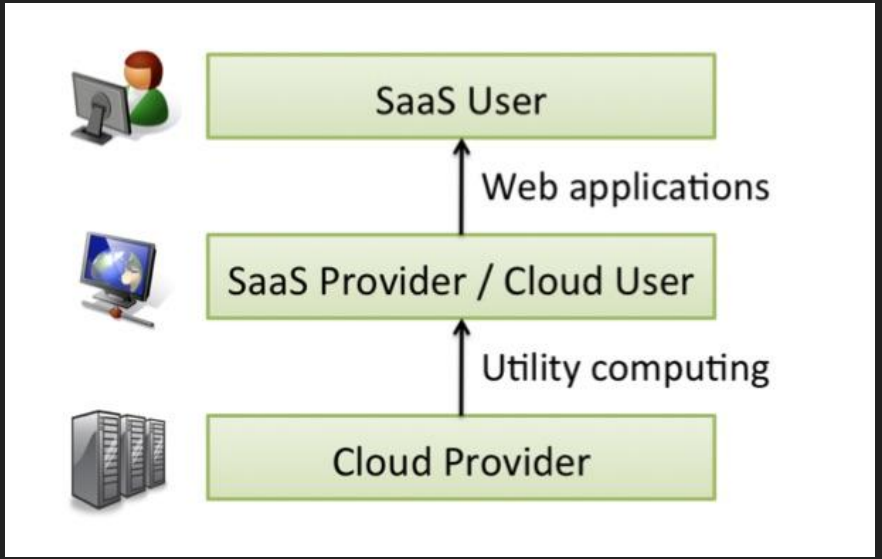
“I don’t need a hard disk in my computer if I can get to the server faster… carrying around these non-connected computers is byzantine by comparison.”

- Steve Jobs

Okay, so what is Cloud Computing?

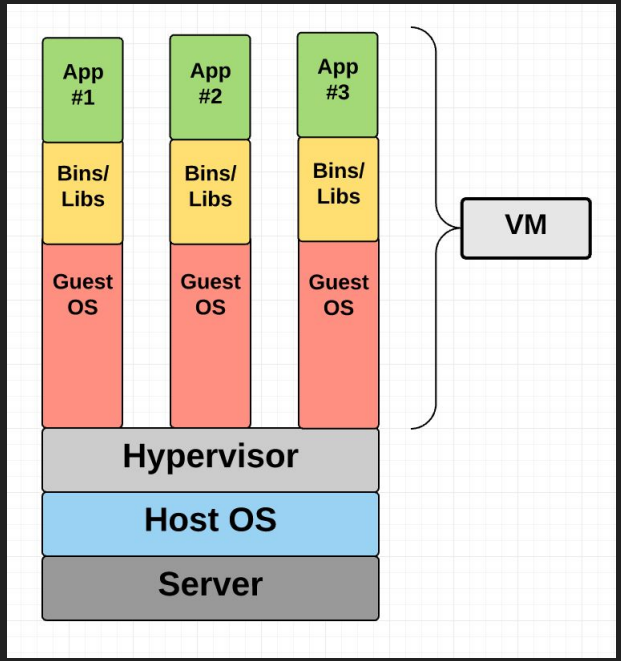
- ***Cloud*** -- datacenter resources (hardware + software) that provide the software as a service (SaaS)

- ***Public cloud*** -- cloud available in pay-as-you-go plan
- ***Utility computing*** -- services sold (AWS, Azure)
- ***Cloud computing*** -- SaaS + utility computing



Why It's Great

1. Illusion of infinite computing resources on demand (ask for more servers, storage? You got it!)
2. No commitment -- pay for service and increase resources when demand increases
3. Short term usage of resources -- when we are done with data storage or processors → another use can use, we don't have to pay anymore



virtual machine

Where cloud computing shines

- Data analysis of terabytes of data: sufficiently parallel application → reduce total time by using many computers
    - Washington Post used 200 EC2 instances (1400 server hours) to convert 17,000 pages of Hillary Clinton's travel documents to a friendlier form within 9 hours of release
- Compute-intensive desktop apps (Matlab, Mathematica) -- use cloud computing to do expensive calculations
    - Applications in machine learning, data science
- Online image rendering, 3D animation -- heavy computation that is embarrassingly parallel

How do cloud providers benefit?

- Amazon Web Services makes up 40% of Amazon's value
- Economy of scale -- large data centers (tens of thousands of computers) provide same services as medium data centers for ⅕ to 1/7 of price
- Large companies (Amazon, Microsoft, Google) already have large infrastructure for internal operations
- Customer retention -- users of their other products want to integrate their app seamlessly with the cloud provider
- Low cost -- datacenters in Washington, Texas for cheap real estate, tax, labor, electricity cost

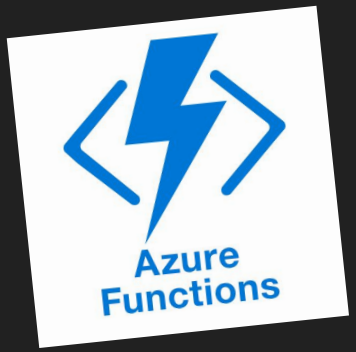
Utility Computing Classes

- Virtual machine, like physical hardware (low-level)
- Gives use lot of control -- whole software stack from kernel upwards
- Hard for Amazon to scale easily because increase of resources is app dependent

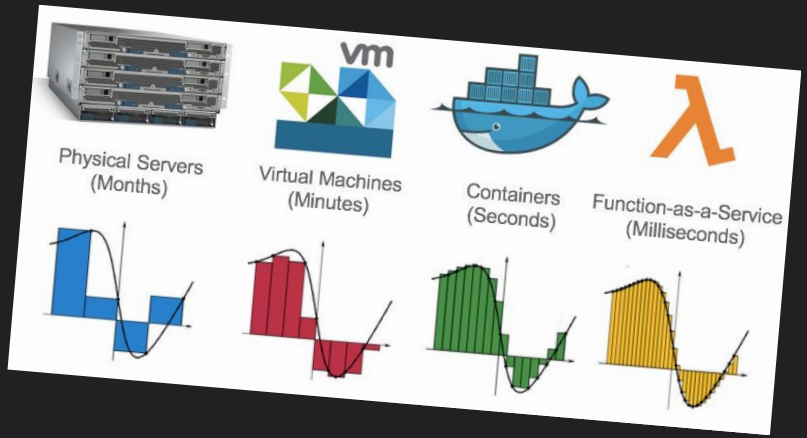
- For only traditional web apps -- rigid but scalable
- Strictly rations amount of CPU time to service each request
- Scales automatically but constrainment on developer's needs, not for general purpose computing

Limitations

- Data transfer bottleneck -- slow and expensive
  - Transfer of 10TB data from NorCal to Seattle takes 45 days, $1000 network transfer fees
  - Why not ship disks overnight? Ten 1TB disks w/ overnight shipping -- $400
  - Amazon began hosting large public datasets on S3 (file storage) easily transferable to EC2
- Data confidentiality -- public networks exposed to more attacks
  - Remedied by encryption, firewalls
- Data lock-in -- difficulty of data extraction, possible data loss
- Bugs must be reproduced at large scale to be fixed



Cloud Computing Today



What’s wrong with the past model?

- Management of virtual resources is fairly complicated
- Stateful services like databases -- often bursty workloads result in unused resources
- Amazon’s EC2 model is most successful -- allows developers to replicate development environment in production
    - Gives more flexibility
    - But… virtual machines have to be managed by system administrators setting up and maintaining environment

Serverless Computing & Cloud Functions

- Not *actually* serverless, simply appears so to developer
- User writes code → cloud provider takes care of server setup (installs OS, necessary software, administration tasks)
- ***Cloud functions*** -- functions executed in the cloud
  - Written in high level language (Python, JavaScript, etc) → called on specified trigger (event-based workflow, more efficient use of resources)
- ***Serverless computing*** -- function + backend as service
    - Scales automatically without explicit setup
    - Billed on usage

AWS EC2

- Serverful
- “Rent out” resources, pay by amount of resources allocated
- Developer needs to know about and specify the hardware
- Greater flexibility and control

AWS Lambda

- Serverless
- Payment not by resources allocated but by execution time (how much function is called) and what is needed to run it
- Provides runtime environments like NodeJS, Python, Java, C#
- Cloud user does not have to specify resources

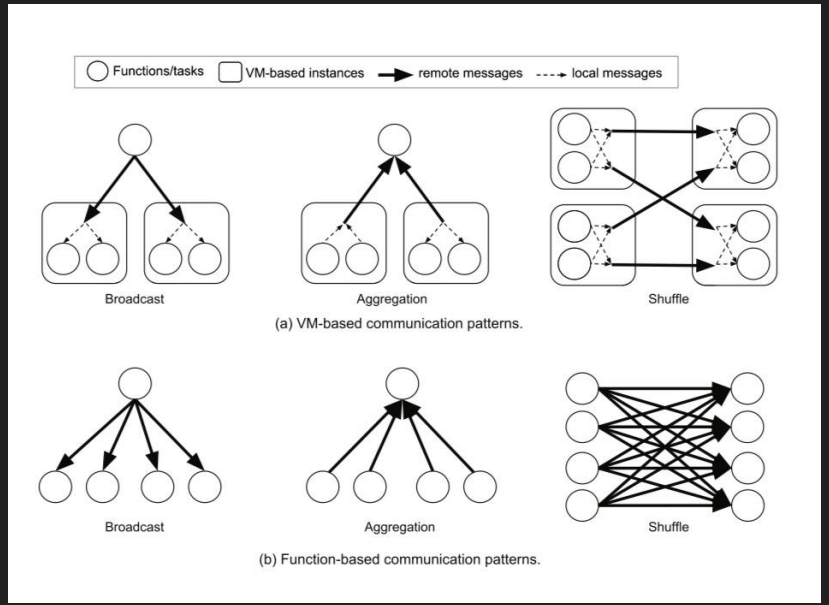
Why this is better

- Better autoscaling -- better tracks loads, scaling up and ***down***
- Better ecosystem support with standardized low level implementation
- Easier to use -- no need to understand low level like EC2
- Stateless functions, short runtime → better task multiplexing, cloud providers maximize their own resources → lower cost
    - No idle time that EC2 instances often experience
- Cloud deployment now high level, not low-level so Amazon can improve low-level implementation under the hood

Percent	*Use Case*
32%	Web and API serving
21%	Data Processing, e.g., batch ETL (database Extract, Transform, and Load)
17%	Integrating 3rd Party Services
16%	Internal tooling
8%	Chat bots e.g., Alexa Skills (SDK for Alexa AI Assistant)
6%	Internet of Things

Limitations

- Figuring out how to implement Serverless Ephemeral Storage and other scalable remote storage spaces
- Better tackle security challenges with cloud attackers
- Minimize risks of unpredictable prices with more structured business models for the consumers
- Task coordination -- communication between functions/tasks



Future of Serverless Computing

- Serverless computing usage will skyrocket while cloud hybrids will slowly dwindle out.
- Open up new paths for stronger, more robust data analytics
- Gradually support more variety of cloud functions like packet processing and storage parsing

Sources

- https://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-3.pdf
- https://www2.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf
- https://apprenda.com/library/glossary/definition-cloud-instance-single-multi/
- https://www.bmc.com/blogs/what-is-batch-processing-batch-processing-explained/
- https://read.acloud.guru/aws-lambda-vs-google-cloud-functions-vs-azure-functions-who-has-the-serverless-advantage-f6c2535e72f4
- https://www.sumologic.com/blog/devops/kubernetes-vs-docker/
- https://web.stanford.edu/~anakli/pdf/serverless-atc18.pdf

Thanks!

Any questions?