Cloud Applications Architecture

•••

Course 2 - Infrastructure as a Service

IaaS, PaaS, SaaS - What We Manage

On-Premises	IaaS	PaaS	SaaS
Application	Application	Application	
Data	Data	Data	
Runtime	Runtime		
Middleware	Middleware		
OS	OS		
Virtualization			
Server			
Storage			
Networking			

What Really Is IaaS?

- A way to provision and use (mostly virtualized) infrastructure when and while needed (**on-demand**) without having to worry about hardware and maintenance.

We are still responsible for what happens on the infrastructure (e.g. what operating system it runs or whether it is up to date or not).

- The foundation on which other service models are built.
- Datacenter abstraction.

Virtual Machines (VMs)

“The workhorse of the entire industry”

Everything we will see in this course is more or less based on VMs.

Emulated computers.

A way to create dynamically configured machines on demand.

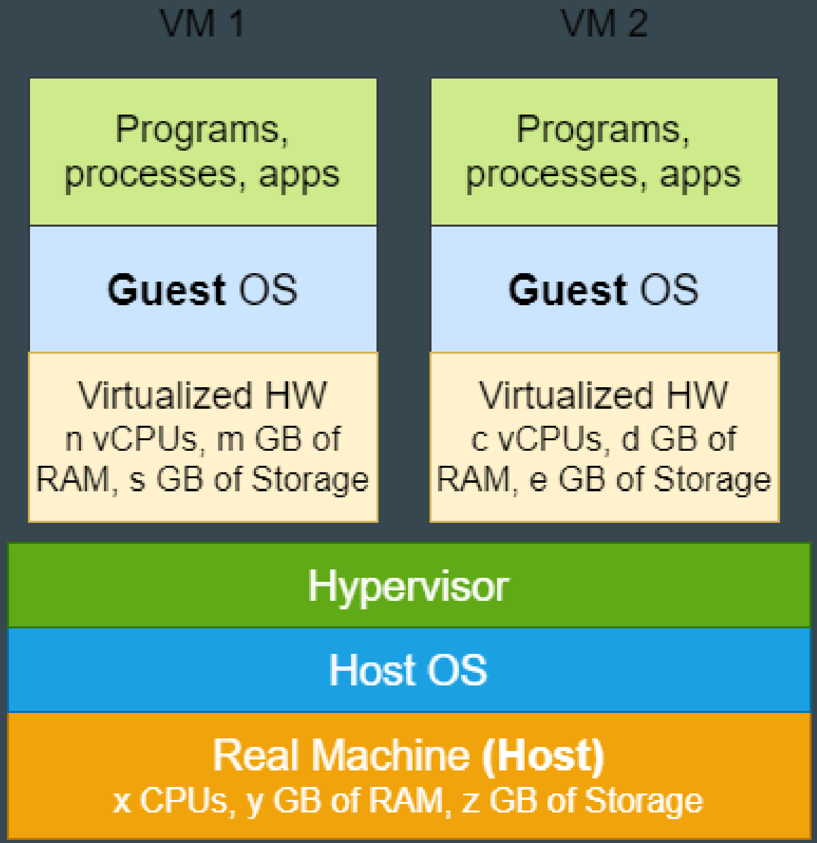
Are also useful for **local development.**

Virtual Machines (VMs)

Host: the machine on which the VMs will be running.

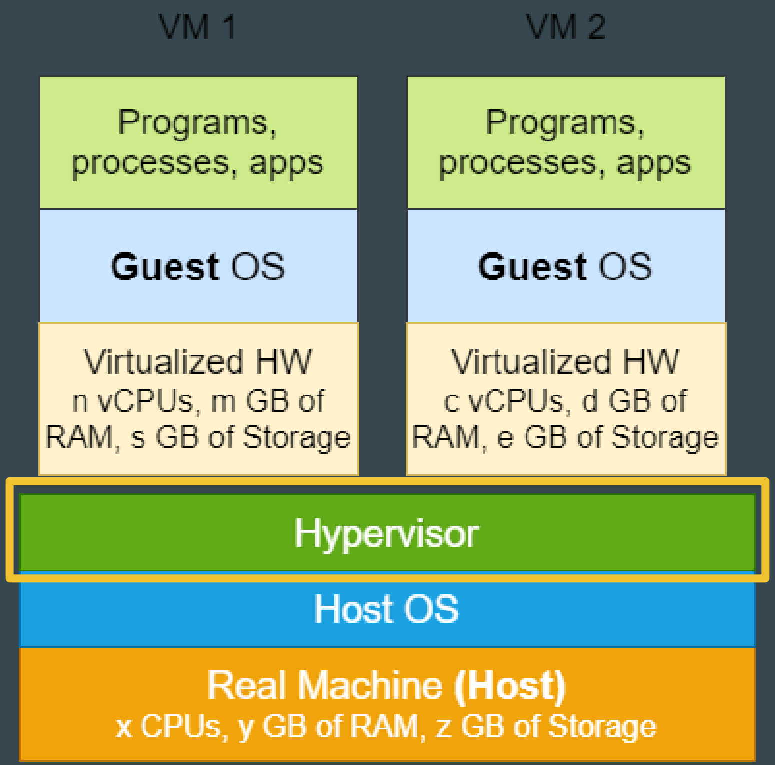
Guest: the virtualized machine

Hypervisor: Creates and manages guest machines, translates and limits instructions.


VM 1
VM 2
Programs, processes, apps
Guest OS
Virtualized HW
n vCPUs, m GB of RAM, s GB of Storage
Programs, processes, apps
Guest OS
Virtualized HW
c vCPUs, d GB of RAM, e GB of Storage
Hypervisor
Host OS
Real Machine (Host)
x CPUs, y GB of RAM, z GB of Storage

The Hypervisor

- Optimize resource usage
  - lower provider cost => lower customer cost
- Enable multi-tenant usage (VMs for multiple clients on the same machine)
  - Optimize costs
  - **Ensure isolation**
- Most providers have custom hypervisors (AWS - Nitro, Azure - Azure Hypervisor/Customized Hyper-V, GCP - KVM).
- Providers might also support other hypervisors, e.g. **VMWare** (easier migration).


VM 1
VM 2
Programs, processes, apps
Programs, processes, apps
Guest OS
Guest OS
Virtualized HW n vCPUs, m GB of RAM, s GB of Storage
Virtualized HW c vCPUs, d GB of RAM, e GB of Storage
Hypervisor
Host OS
Real Machine (Host) x CPUs, y GB of RAM, z GB of Storage

Virtual Machines - Configurations

You choose the configuration based on your needs.

- General purpose
- CPU heavy
- Memory heavy
- GPU

Providers might give a set of pre-configured options or allow you to customize one.

vCPUs

Highly dependant on the provider.

Might be:

- Timeshares of real CPU cores
- Physical core
- Logical core (think of hyperthreading) (AWS, GCP)

Virtual Machines - OSs

Linux (most common flavors + provider specific)

Windows Server

Optimized OSs (e.g. container optimized OS)

Even Mac OS (e.g. macincloud)

You still have to buy licenses for certain OSs (e.g. Windows)

Virtual Machines - Remote Access

1. **SSH** (Linux VMs)
2. **RDP** (Windows VMs)

Usually a security concern.

Production VMs are usually accessed through **bastion hosts**.

- Intermediary VMs with the sole purpose of granting access

Virtual Machines - Monitoring

Why monitor?

- Business continuity
- Respecting contracts (SLA)
- Know when to scale (up/down, in/out)
- Understand your application
- Understand your users

Virtual Machines - Monitoring

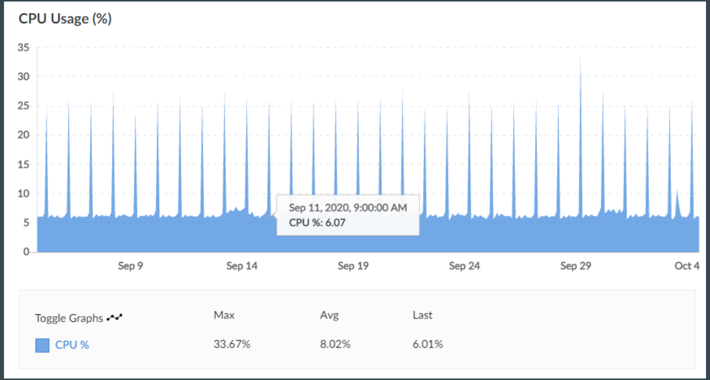
Commonly monitored metrics:

- CPU Usage
- Disk Usage
- Network Traffic (inbound/outbound)
- Memory

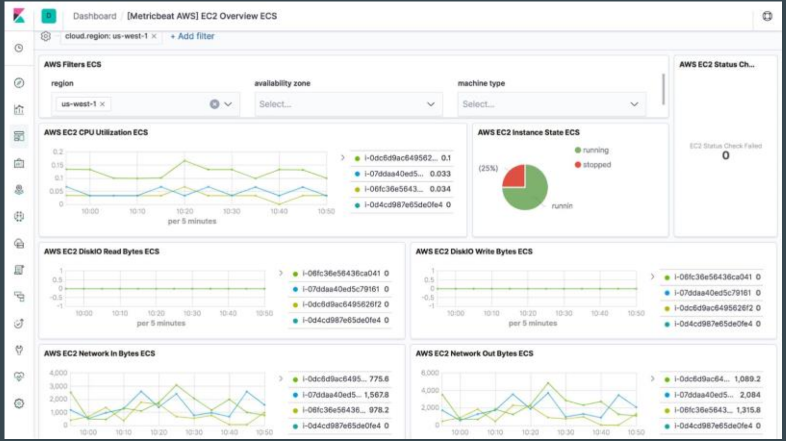
Virtual Machines - Monitoring

Tools:

- Some are already provided by the hypervisor itself
- Provider specific tools
  - CloudWatch
  - Azure Monitor
  - Google Operations (previously known as Stackdriver)
- The **ELK** Stack



	Max	Avg	Last
🟦 CPU %	33.67%	8.02%	6.01%



Virtual Machine Services - Examples

EC2

Azure VM

GCE

Droplets

Oracle VM

...and others

Virtual Machines - Automation

Infrastructure as Code (IaC)

```
Resources:
  ExampleEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: t2.micro
```

AWS CloudFormation

```
{
      "type": "Microsoft.Network/virtualNetworks",
      "apiVersion": "2020-06-01",
      "name": "[variables('virtualNetworkName')]",
      "location": "[parameters('location')]",
      "dependsOn": [
        "[resourceId('Microsoft.Network/networkSecurityGroups', 
variables('networkSecurityGroupName'))]"
      ],
      "properties": {
        "addressSpace": {
          "addressPrefixes": [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets": [
          {
            "name": "[variables('subnetName')]",
            "properties": {
              "addressPrefix": "[variables('subnetPrefix')]",
              "networkSecurityGroup": {
                "id": 
"[resourceId('Microsoft.Network/networkSecurityGroups', 
variables('networkSecurityGroupName'))]"
              }
            }
          }
        ]
      }
    }
```

Azure ARM Template

```
resource "google_compute_instance" "default" {
  name         = "test"
  machine_type = "n1-standard-1"
  zone         = "us-central1-a"
  tags = ["foo", "bar"]
  boot_disk {
    initialize_params {
      image = "debian-cloud/debian-9"
    }
  }
  // Local SSD disk
  scratch_disk {
    interface = "SCSI"
  }
  network_interface {
    network = "default"

    access_config {
      // Ephemeral IP
    }
  }
  metadata = {
    foo = "bar"
  }
  metadata_startup_script = "echo hi > /test.txt"

  service_account {
    scopes = ["userinfo-email", "compute-ro", "storage-ro"]
  }
}
```

Terraform

Boot Up Scripts

Terminal commands executed during the **first boot** of the VM.

- Usually with admin/root privileges

Used to install various programs and configure the system (e.g. create an user).

Impact the boot time.

Images

Manually configure a VM, create image from it, spin up identical VMs in seconds.

Pay for the storage

Best practice (aws: Golden AMI)

Can be sold (marketplaces)

Dedicated Services/Tools

Automatically configure infrastructure and/or system configuration and dependencies.

Usually require an agent to be installed.

Some require a master node (separate VM).

ANSIBLE

CHEF

puppet

SALTSTACK

Storage

Storage is usually separated from the VM.

One of the major differences when migrating to cloud

VM = CPU + RAM, **needs storage**

Volumes can be attached to different VMs.

Storage

Block storage (as opposed to object storage)

Boot disks (where the OS is installed), additional storage

SSD, HDD, Magnetic (can be attached to one VM at a time)

Encryption

Snapshots

Instance store (ultra high throughput, but usually temporary)

NAS (multiple VMs can use it at the same time)

Networking

Some provide more than others (only public instances to fully customizable network topology). We will focus on the latter.

Similarly to storage, network cards are attached to VMs. Primary and secondary.

More details in next course

Working with IaaS

1. Create a VM using one of the tools made available by the provider
   a. Web console
   b. CLI
   c. SDK
   d. API
2. Connect to it (optional)
3. Install the necessary programs (can and should be automatized)
   a. Application dependencies
   b. Monitoring software
4. Deploy your application (can and should be automatized)
5. Monitor the VM and app
6. Keep the VM (OS + programs) and app up-to-date

Other Features

Deploy VMs configured by others

- Can be found on marketplaces (e.g. AWS, GCP, etc.)
- Usually leverage IaC tools

Cost optimization

- Preemptible VMs
- Long-term commitments
- Sustained usage
- Choosing the right specifications
  - Sometimes, the more expensive VM is actually cheaper

When to Use IaaS?

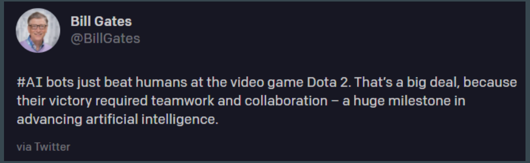
- First migration phase(s) (lift & shift)
- “Special” programs
  - Usually native programs
- When ultra-high performance is a must
  - You can rent physical machines (actual racks)
- “Special” licenses
- When trying to avoid vendor lock-in at all costs (rarely worth)
- Certain long-running workloads
  - Usually lower cost than alternatives
- When you want to be in control

Case Study

OpenAI (AI for playing and winning Dota 2)

1v1 (First project) and **5v5 (current)**

Has a win rate of **99.4%** (people could play against it).



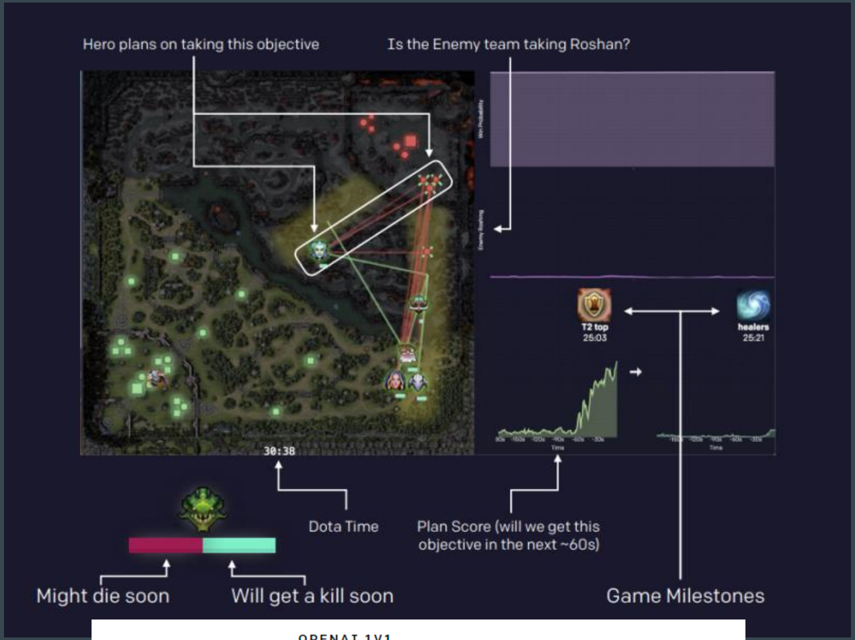
Bill Gates
@BillGates

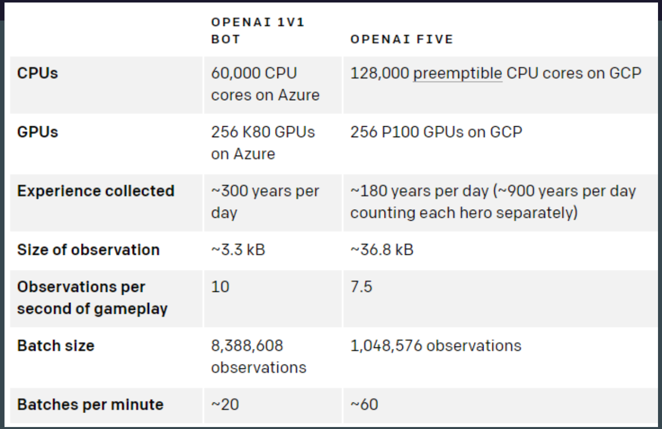
#AI bots just beat humans at the video game Dota 2. That's a big deal, because their victory required teamwork and collaboration – a huge milestone in advancing artificial intelligence.

via Twitter

Timeline of events

~$25k / day


Hero plans on taking this objective
Is the Enemy team taking Roshan?
T2 top
25:03
healers
25:21
30:38
Dota Time
Plan Score (will we get this objective in the next ~60s)
Might die soon
Will get a kill soon
Game Milestones



	OPENAI 1V1 BOT	OPENAI FIVE
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)
Size of observation	~3.3 kB	~36.8 kB
Observations per second of gameplay	10	7.5
Batch size	8,388,608 observations	1,048,576 observations
Batches per minute	~20	~60

Summary

1. The responsibility model
    a. OS, other tools/programs, for setting up our software, keeping everything up to date
2. The **hypervisor,** the guest, the host machine
3. Configure the VM based on our needs
4. OS (linux, container optimized OSs, windows)
    a. SSH/RDP for connecting to the VM
5. Monitoring (choose the relevant metric)
    a. ELK stack to handle monitoring for multiple VMs (allows us to get a better overview)
    b. Munin
    c. CloudHealth – for cost monitoring
6. Automation: IaC, boot up scripts, images, tools for provisioning software
7. Storage, Networking