

CSCI 5409 Adv Topics in Cloud Computing – Fall 2023
Week 4 – Lecture 1 (Sep 25, 2023)

Cloud-Enabling Technology – Virtualization and WWW

Dr. Lu Yang
Faculty of Computer Science
Dalhousie University
luyang@dal.ca

Housekeeping and Feedback

- First assignment due date extended to 11:59pm, Sep 29.
- PIER tour tomorrow, 10:00-11:00am. I will be waiting for you at the door. A spreadsheet will be put up in our Teams channel later this afternoon. I can only accept 10 students per tour.

Objectives

- Understand the technologies that make cloud computing possible
 - Virtualization
 - World Wide Web

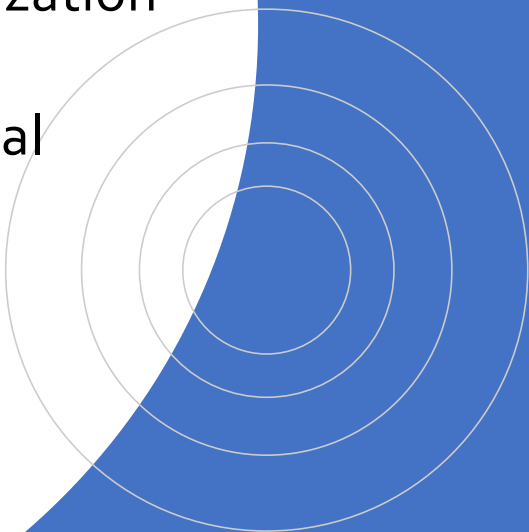
Contents

- Section 1.** **Cloud Virtualization Overview**
- Section 2.** **World Wide Web**



1

Cloud Virtualization Overview

1. Intro
 2. Virtualization technology
 3. What techniques make virtualization possible?
 4. Deploying web apps with virtual machines
 5. Containerization
- 

Cloud Virtualization Overview — Intro (1/2)

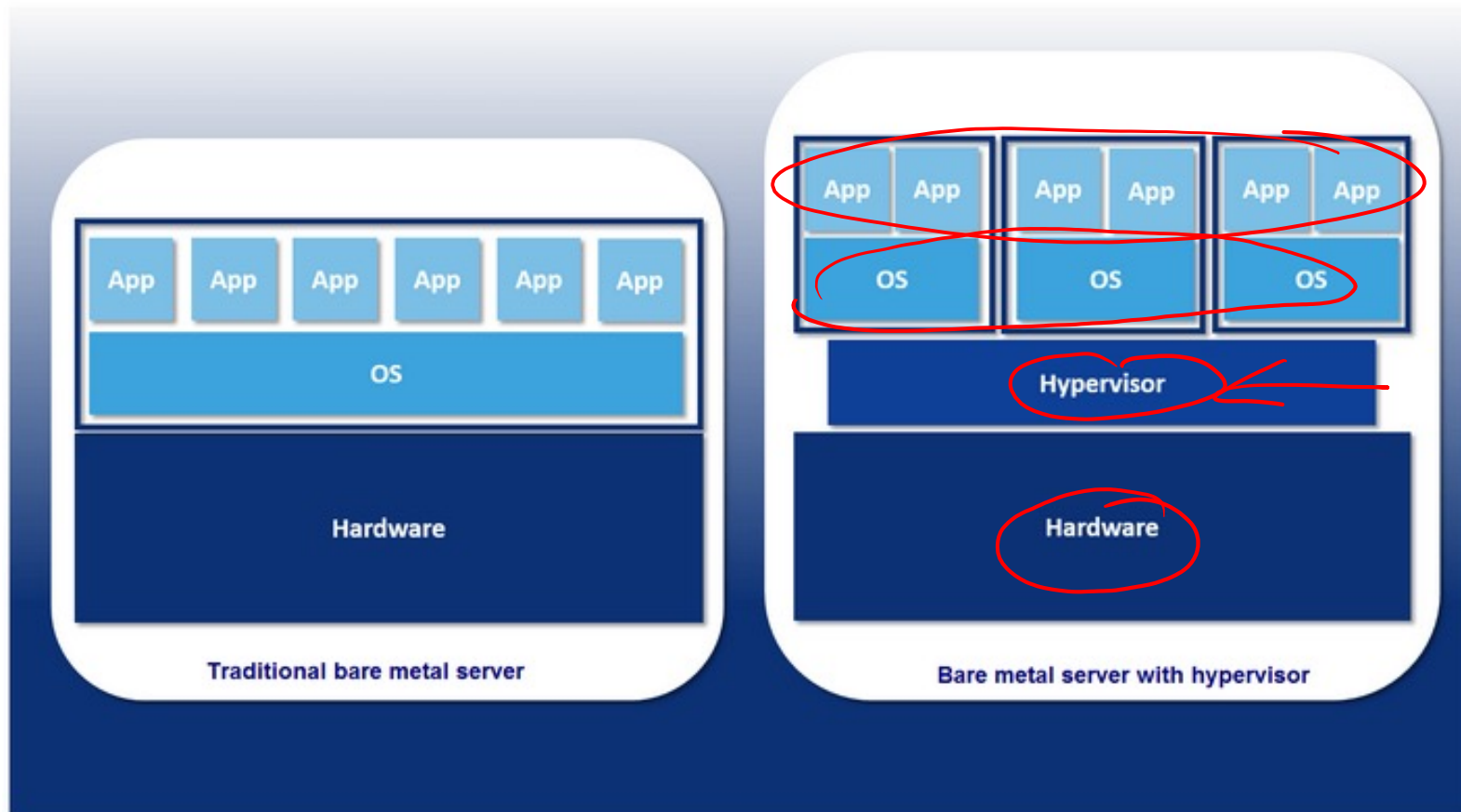


Figure from: <https://www.ionos.ca/digitalguide/server/know-how/bare-metal-servers-definition-and-structure/>

Cloud Virtualization Overview — Intro (2/2)

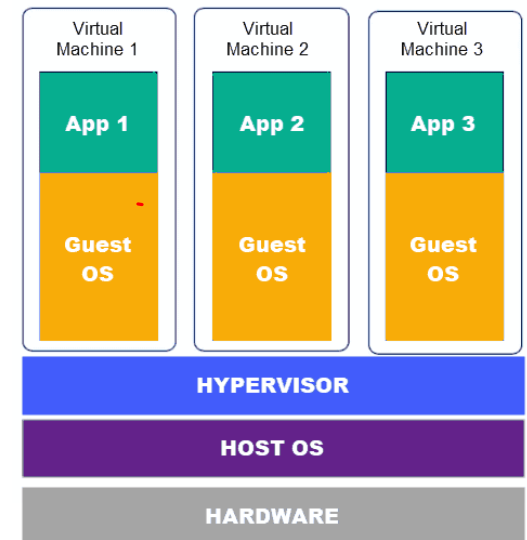
- Virtualization is a process of converting a physical IT resources into a virtual IT resource ^[1]
- Most types of IT resources can be virtualized, including^[1]
 - Servers ✓
 - Storage ✓
 - Network ✓
 - Power ✓
- Virtualization allows multiple virtual machines on one physical machine ^[2]
 - heterogeneous operating systems
 - run in isolation
- The operating system and application software running on the virtual server are unaware of the virtualization process
- Virtualization enables both horizontal AND vertical scaling
- Benefits of virtualization^[2]
 - Reduced capital and operation costs;
 - Minimized or eliminated downtime;
 - Increased IT productivity, efficiency, agility and responsiveness;
 - Faster provisioning of applications and resources;
 - Greater business continuity and disaster recovery;
 - Simplified data center management;
 - Availability of a true Software-Defined Data Center.

1. Cloud Computing (T. Erl, Z. Mahmood, R. Puttini), Section 5.3
2. <https://www.vmware.com/in/solutions/virtualization.html>

Virtualization Technology (1/5) — Terms

- Virtualization refers to technologies that are designed to provide a layer of abstraction between layers of hardware and software that decouples the physical hardware from the operating system^[2]
- Terms
 - Host machine
 - Virtual machine
 - VMM = Hypervisor

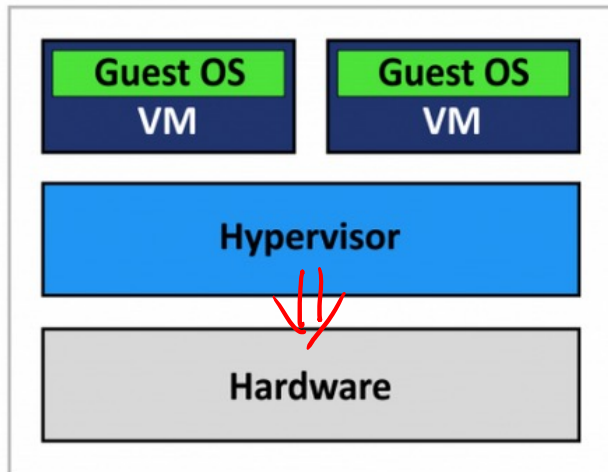
VMM is also called a “hypervisor,” this is one of many hardware virtualization techniques that allow multiple operating systems, termed guests, to run concurrently on a host computer. It is so named because it is conceptually one level higher than a supervisory program. The hypervisor presents to the guest operating systems a virtual operating platform and manages the execution of the guest operating systems. Multiple instances of a variety of operating systems may share the virtualized hardware resources.
- Virtualization characteristics^[3]
 - Partitioning
 - Encapsulation
 - Isolation
 - Hardware Independence



1. Figure from: <https://phoenixnap.com/kb/containers-vs-vms>
2. Cloud Computing Basics (T. B. Rehman), Chapter 3
3. <https://www.vmware.com/in/solutions/virtualization.html>

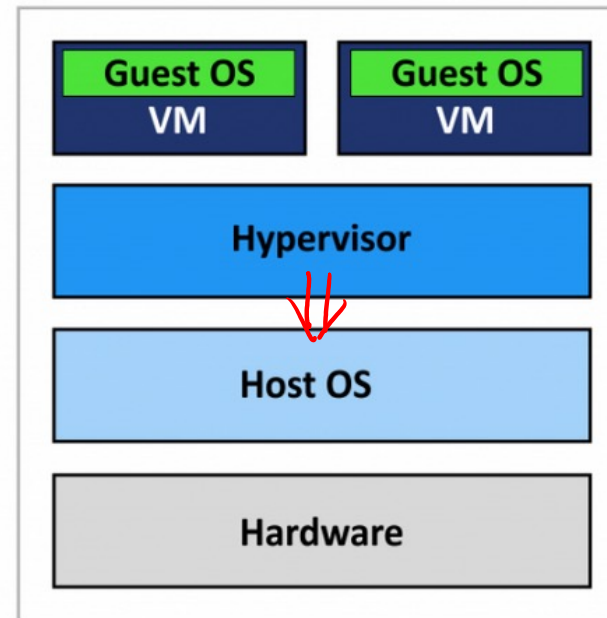
Virtualization Technology (2/5) — Virtualization Types

Native



Type 1 Hypervisor
(Bare-Metal Architecture)

Examples:
Hyper-V
Xen
VMware ESX



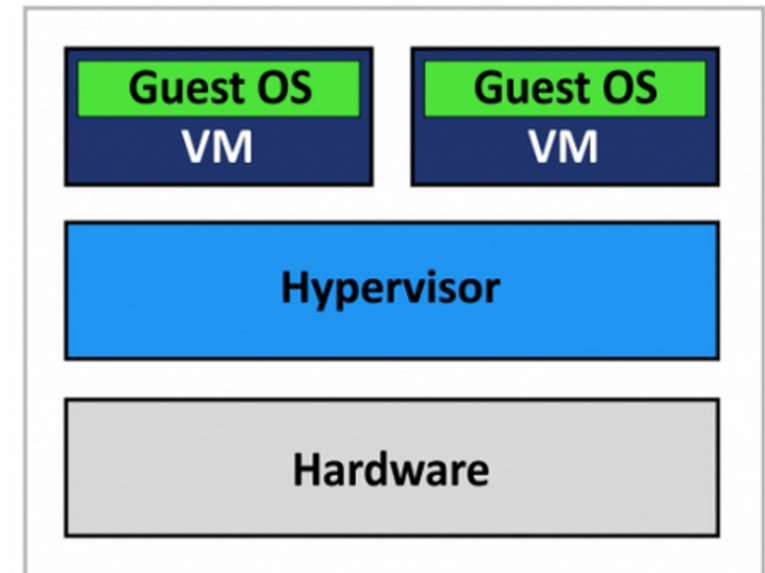
Type 2 Hypervisor
(Hosted Architecture)

Examples:
Virtual PC & Virtual Server
VMware Workstation
KVM

Virtualization Technology (3/5) — Virtualization Types (Bare-Metal)

- "This option represents the installation of virtualization software directly on the physical host hardware to bypass the host operating system."^[1]
- The virtual machine management layer is referred to as the **hypervisor**, this thin layer handles hardware management functions
- Pros:
 - Shed the IT resource consumption and license fees of host OS
 - More efficient by interacting with hardware directly
- Cons:
 - Concerns of compatibility with hardware devices, hence reduced driver support (mitigated by commodity hardware)

Native



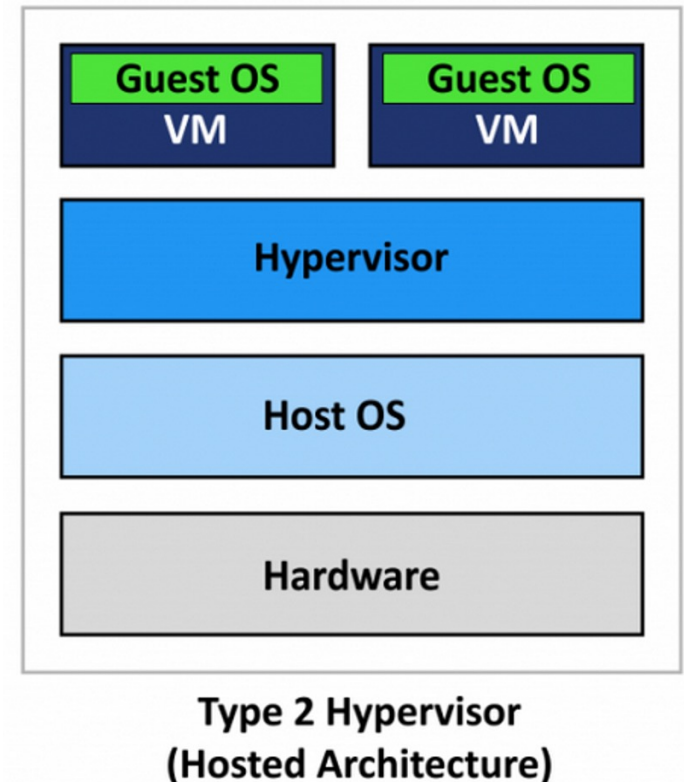
**Type 1 Hypervisor
(Bare-Metal Architecture)**

[1]: Cloud Computing (T. Erl, Z. Mahmoud, R. Puttini), pg. 101

Figure from: <https://www.nakivo.com/blog/hyper-v-virtualbox-one-choose-infrastructure/>

Virtualization Technology (4/5) — Virtualization Types (Hosted)

- "Operating system-based virtualization is the installation of virtualization software in a pre-existing operating system, which is called the host operating system."^[1]
- Pros:
 - Backup and recovery
 - Integration with directory services (roles & permissions)
 - Security management
- Cons:
 - Host OS consumes resources
 - Hardware related operations have increased latency as they traverse layers
 - OS licenses \$\$\$



[1]: Cloud Computing (T. Erl, Z. Mahmoud, R. Puttini), pg. 101

Figure from: <https://www.nakivo.com/blog/hyper-v-virtualbox-one-choose-infrastructure/>

Virtualization Technology (5/5) — Comparison

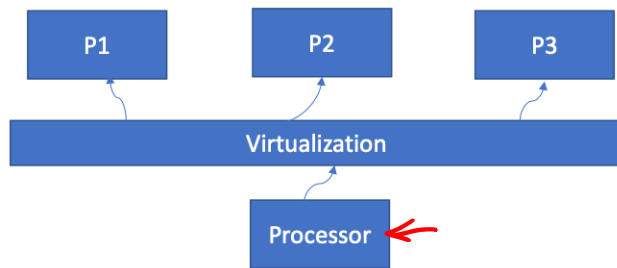
Hypervisor Type 1 vs. Type 2 in Tabular Form

Criteria	Type 1 Hypervisor	Type 2 Hypervisor
AKA	<u>Bare-metal</u> or Native	<u>Hosted</u>
Definition	Runs directly on the system with VMs running on them	Runs on a conventional Operating System
Virtualization	<u>Hardware</u> Virtualization	<u>OS</u> Virtualization
Operation	<u>Guest OS and applications run on the hypervisor</u>	<u>Runs as an application on the host OS</u>
Scalability	Better Scalability	Not so much, because of its reliance on the underlying OS.
Setup/Installation	Simple, as long as you have the necessary hardware support	Lot simpler setup, as you already have an Operating System.
System Independence	Has direct access to hardware along with virtual machines it hosts	Are not allowed to directly access the host hardware and its resources
Speed	<u>Faster</u>	<u>Slower</u> because of the system's dependency
Performance	<u>Higher-performance</u> as there's no middle layer	Comparatively has reduced performance rate as it runs with extra overhead
Security	More Secure	Less Secure, as any problem in the base operating system affects the entire system including the protected Hypervisor
Examples	<ul style="list-style-type: none"> • VMware ESXi • Microsoft Hyper-V • Citrix XenServer 	<ul style="list-style-type: none"> • VMware Workstation Player • Microsoft Virtual PC • Sun's VirtualBox

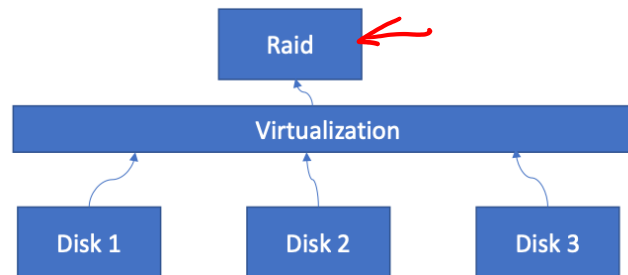
Table from: <https://www.hitechnectar.com/blogs/hypervisor-type-1-vs-type-2/>

- **Type 1 (bare-metal)**
- For enterprise applications and cloud computing, type 1 is preferable, primarily because of its independence from the host operating system.
- For the same reason, type 1 generates lesser overhead, and any malfunction in an individual VM does not harm the rest of the system.
- **Type 2 (hosted)**
- Type 2 does not have direct access to the host hardware and resources, so this may make a certain degree of latency inevitable. The already present OS manages the requirements for memory, storage, and network resources.
- Hosted Hypervisors are still popular for personal use, some developer environments, like where access to multiple OSs and their variants is required, Type 2 hypervisors are a better option. On devices not dedicated to the VMs Host role, hosted hypervisors are recommended.

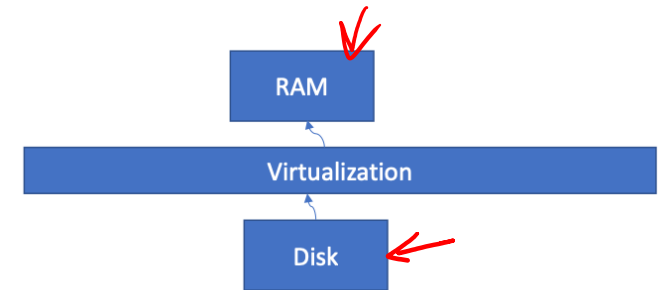
What techniques make virtualization possible?



- **Multiplexing:** $1-M$
 - Creates multiple virtual objects from one instance of a physical object
 - In the example above a single processor is multiplexed and time shared among a number of processes or threads



- **Aggregation:** $M-1$
 - Creates one virtual object from multiple physical objects
 - In the example above many hard drives are virtualized into a single RAID disk



- **Emulation:**
 - Constructs a virtual object from a different type of physical object
 - In the example above a physical disk emulates Random Access Memory (RAM)

These techniques can be combined! For example, virtual memory with paging multiplexes real memory and disks, a virtual address emulates a real address.

Deploying Web Apps with Virtual Machines

1. Build the virtual machine:
 - 1) Install guest operating system (usually this is done for you by the cloud provider)
 - 2) Connect to virtual machine remotely (RDP, SSH, etc.)
 - 3) Install OS updates ← *automate*
 - 4) Configure and install dependencies (IIS, Apache Tomcat, programming framework, etc...)
 - 5) Install app
 - 6) Test
 - 7) Save

IaaS

Containerization

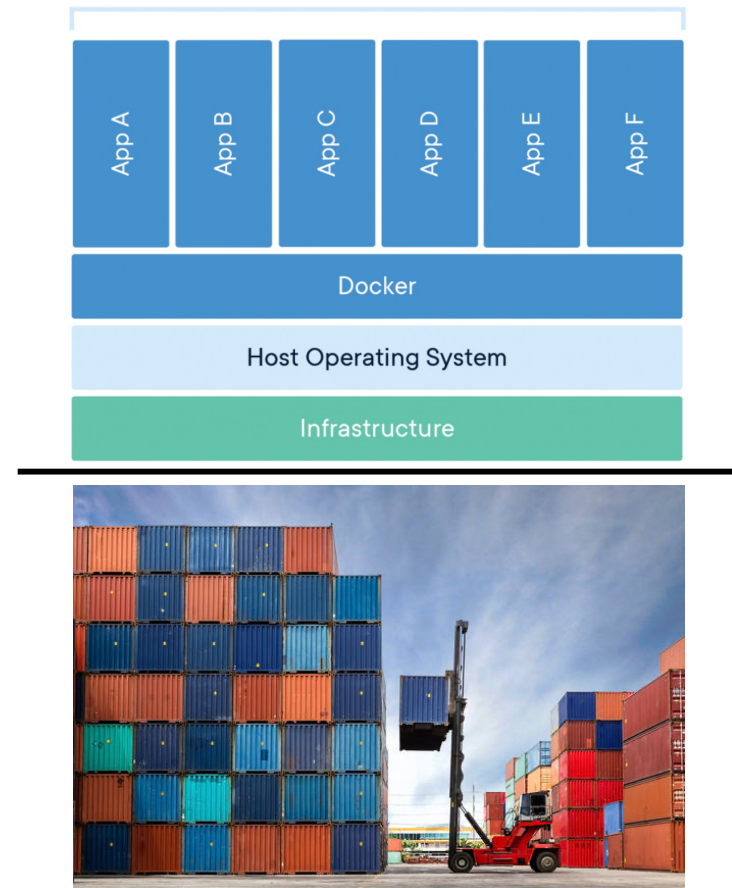


- "A **container** is a standard unit of software that packages up code and **all** its dependencies so the application runs quickly and reliably from one computing environment to another."^[1]
- Virtual machines need a guest operating system, containers do not, this makes them very small and lightweight
- Secure: Each container is a read-only sandbox, whatever you put in the container stays exactly how you arranged it, massively increasing reliability and testability

[1]: <https://www.docker.com/resources/what-container>

Photo: <https://www.thesafetymag.com/ca/topics/safety-and-ppe/worksafebc-warns-against-using-shipping-containers-for-other-purposes-/183992>

Containerized Applications



Just like real shipping containers, docker containers can be easily moved around and stored, and when you fill them up, they are just the same when you open them.

Deploying Web Apps with Containers

1. Get it working on your machine *local*
2. Test
3. Configure the container (e.g. dockerfile) *images*
4. Build the image
5. Upload image to a repository, this is fast because the image is as lightweight as possible
6. Provision IT resources (CPU, memory, storage)
7. Deploy the container to compute resources (e.g. EC2, Beanstalk, ECS, EKS)
8. ~~Turn it on~~
9. Test

Why is this better than a virtual machine?

- Virtual machines can be modified
- Once built the container is fixed, this provides additional security to the organization. What you run is what you tested.
- Containers are lightweight, it's easier to move them around
 - Cheaper storage costs (MB vs GB)
 - Faster scaling
- Faster boot time
- Easier to build with infrastructure as code (more on this later), allowing automation and significantly reducing deploy times which increases organizational agility

Differences of Virtual Machines and Containers

Feature	Virtual Machine	Container
Isolation	Provides complete isolation from the host operating system and other VMs. This is useful when a strong security boundary is critical, such as hosting apps from competing companies on the same server or cluster.	Typically provides lightweight isolation from the host and other containers, but doesn't provide as strong a security boundary as a VM. (You can increase the security by using Hyper-V isolation mode to isolate each container in a lightweight VM).
OS	Runs a complete operating system including the kernel, thus requiring more system resources (CPU, memory, and storage).	Runs the user mode portion of an operating system, and can be tailored to contain just the needed services for your app, using fewer system resources.
Deployment	Deploy individual VMs or multiple VMs by tools like PowerShell or System Center Virtual Machine Manager.	Deploy individual containers by using Docker via command line; deploy multiple containers by using an orchestrator such as Kubernetes.
Memory on disk	Complete OS plus apps	App requirement only
Time taken to start up	Substantially longer because it requires boot of OS plus app loading	Substantially shorter because only apps need to start as kernel is already running
Portability	Limited and need proper preparation	More portable than VMs and can be deployed across Public Clouds, Private Cloud and Traditional Data Centers without any conversion

<https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm>



2

World Wide Web



World Wide Web (1/2)

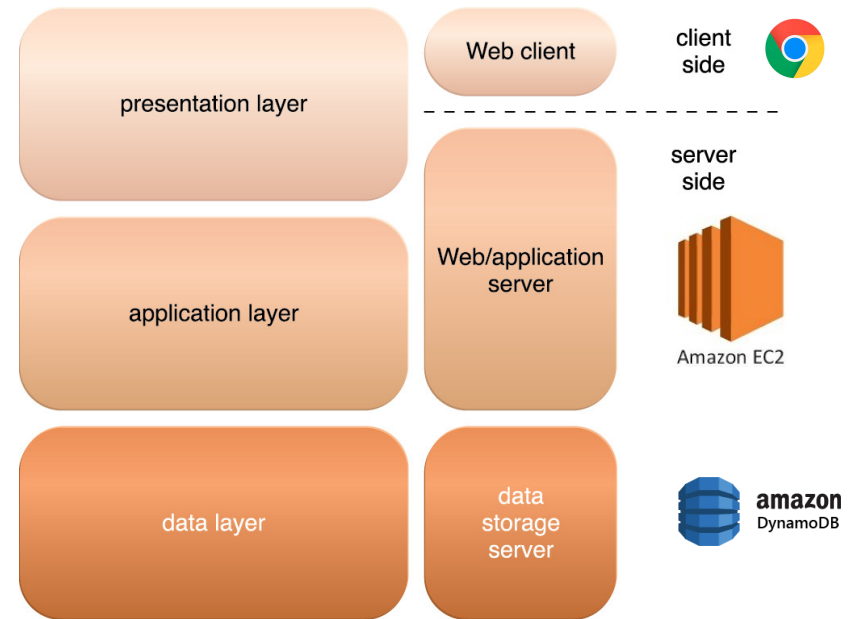
Is WWW the same as the Internet?[1]

1	Internet is a <u>global network of networks</u> .	WWW stands for <u>World wide Web</u> .
2	Internet is a means of <u>connecting a computer to any other computer</u> anywhere in the world.	World Wide Web which is a <u>collection of information which is accessed via the Internet</u> .
3	Internet is <u>infrastructure</u> .	WWW is <u>service on top of that infrastructure</u> .
4	Internet can be viewed as a <u>big book-store</u> .	Web can be viewed as <u>collection of books on that store</u> .
5	At some advanced level, to understand we can think of the <u>Internet as hardware</u> .	At some advanced level, to understand we can think of the <u>WWW as software</u> .
6	Internet is primarily <u>hardware-based</u> .	WWW is more <u>software-oriented</u> as compared to the Internet.
7	It is originated sometimes in <u>late 1960s</u> .	English scientist Tim Berners-Lee invented the World Wide Web in <u>1989</u> .
8	Internet is <u>superset</u> of WWW.	WWW is a <u>subset</u> of the Internet.
9	The first version of the Internet was known as <u>ARPANET</u> .	In the beginning WWW was known as <u>NSFNET</u> .
10	Internet uses <u>IP address</u> .	WWW uses <u>HTTP</u> .

[1]: <https://www.geeksforgeeks.org/difference-between-internet-and-www/>

World Wide Web (2/2)

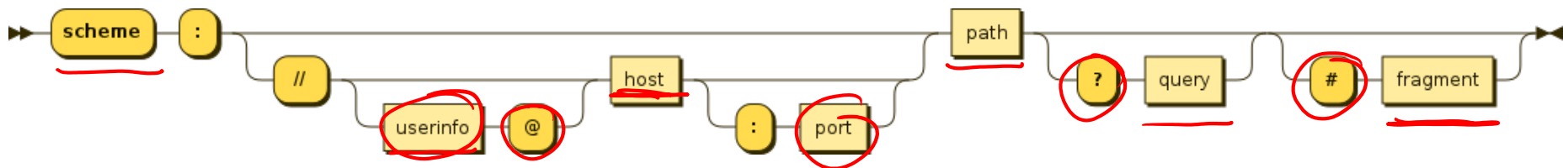
- "Due to cloud computing's fundamental reliance on internetworking, web browser universality, and the ease of web-based service development, web technology is generally used as both the implementation medium and the management interface for cloud services."^[1]
- So, how about web development?
Using clouding computing as a web app development tool users can access any software, applications, images or any kind of data from wherever they are. Offering much faster innovation and flexible resources it can deliver all the computing services over the internet.



The three basic architectural tiers of Web applications.

HTTP & URLs

- Hypertext Transfer Protocol (HTTP):
 - Primary communications protocol used to exchange content and data throughout the WWW
 - Requests are either PUSH or GET, sending data or requesting resources
- Uniform Resource Locator (URL):
 - The "address" of things on the internet, how we uniquely identify resources in web computing
 - Every HTTP URL conforms to the syntax of a generic uniform resource identifier^[1]:

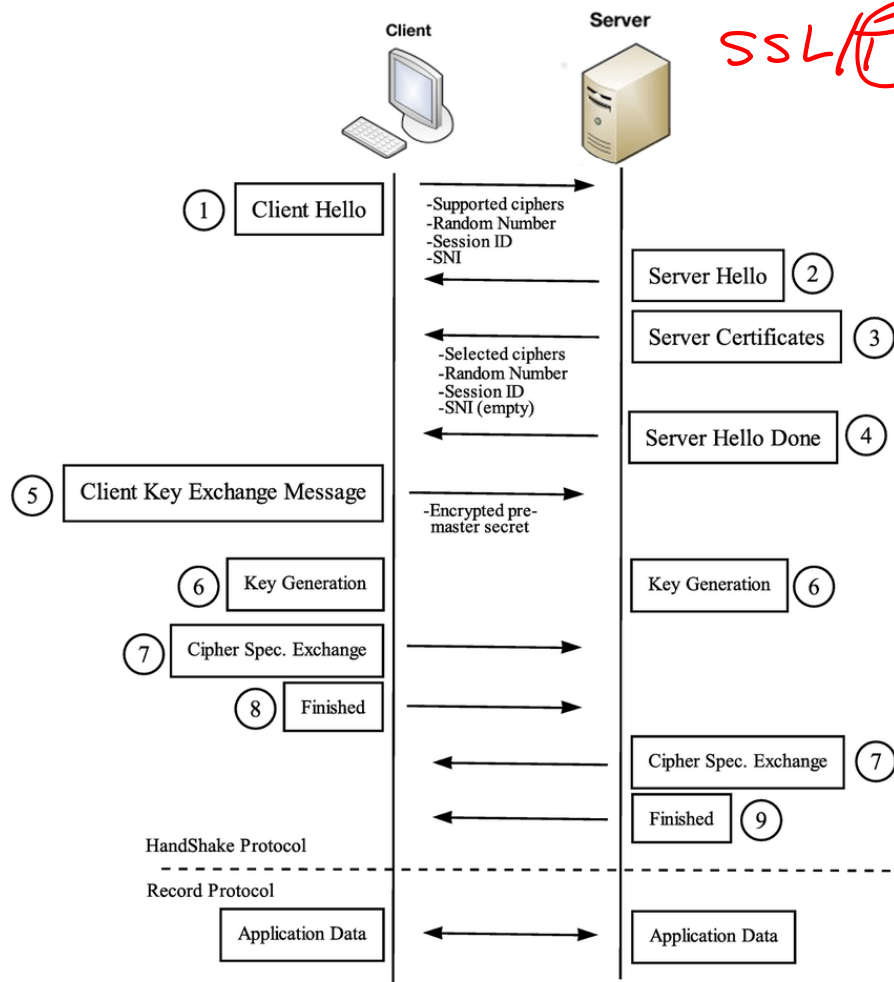


- Examples
 - <https://robusername@website.com:80/givemesomeresource?password=NEVERDOTHIS>

[1]: <https://en.wikipedia.org/wiki/URL>

Transport Layer Security (TLS)

~~SSL~~ / TLS



- Transport Layer Security (TLS) replaces the now-deprecated predecessor Secure Sockets Layer (SSL)^[1]
 - A cryptographic protocol designed to provide communications security over a computer network
 - There are several versions of the protocol, but **HTTPS** is the most publicly visible
- This is the mechanism we use to secure data as it leaves the cloud consumer organization's trust boundary, crosses the open internet and lands in the cloud provider's systems
- TLS uses the Public Key Infrastructure and digital certificates to secure web transactions^[2]

[1]: https://en.wikipedia.org/wiki/Transport_Layer_Security

[2]: <https://www.ssl2buy.com/wiki/ssh-vs-ssl-tls>

Figure: https://www.researchgate.net/figure/TLS-handshake-protocol_fig1_298065605

Data-Interchange Formats

- Often our system architectures require communication between web services we create, we require a mechanism to interchange data:
 - Serialize your object from its binary in-memory format to text
 - Send the text to the other side
 - De-serialize the text back into its binary format
- JSON: JavaScript Object Notation ← Use this
 - Uses human-readable text to store and transmit data objects consisting of attribute-value pairs and array data types^[1]
 - It's **FAST**
- XML: Extensible Markup Language ← Not this, only use if legacy forces you
 - A markup language that defines rules for encoding documents in a format that is both human-readable and machine-readable, emphasizes simplicity, generality and usability^[2]
 - It's **SLOW**

[1]: <https://en.wikipedia.org/wiki/JSON>

[2]: <https://en.wikipedia.org/wiki/XML>

Web Services (Little Programs in the Cloud)

- "A **web service** is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format. Other systems interact with the web service in a manner prescribed by its description, typically conveyed using HTTP with a data interchange format."^[1]
- SOAP: Simple Object Access Protocol^[2]
 - SOAP is on top of HTTP
 - XML-based messaging protocol
 - You send XML that describes your request, the service processes the request and returns an XML response
- REST: Representational State Transfer^[3]
 - Standard for service-oriented architecture
 - ~~stateless~~ protocol (meaning you start from scratch on every request)
 - You send request, receive response (HTML, XML, JSON or something else could be anything)
- We'll learn more about services when we study service-oriented architecture later on

[1]: https://en.wikipedia.org/wiki/Web_service

[2]: <https://en.wikipedia.org/wiki/SOAP>

[3]: https://en.wikipedia.org/wiki/Representational_state_transfer

SOA

The background of the image is a stylized world map divided into four quadrants by a vertical and a horizontal line. The top-left quadrant is red, the top-right is blue, the bottom-left is yellow, and the bottom-right is green. The word "Kahoot!" is written in a large, white, bold, sans-serif font across the center of the image, spanning all four quadrants.

Kahoot!