CPSC 436C
Cloud Computing for Data Science

Cloud Computing Service Models
Serverless Computing and Containerization

Maryam R.Aliabadi

mraiyata@cs.ubc.ca

Fall 2023

Last class' review

- Why a computer is not enough?
- Data center as a computer
- Cloud Characteristics
- Cloud Service Models

Todays' Agenda

- Cloud Deployment Models
- Introduction to Computing Services
- Serverless Computing

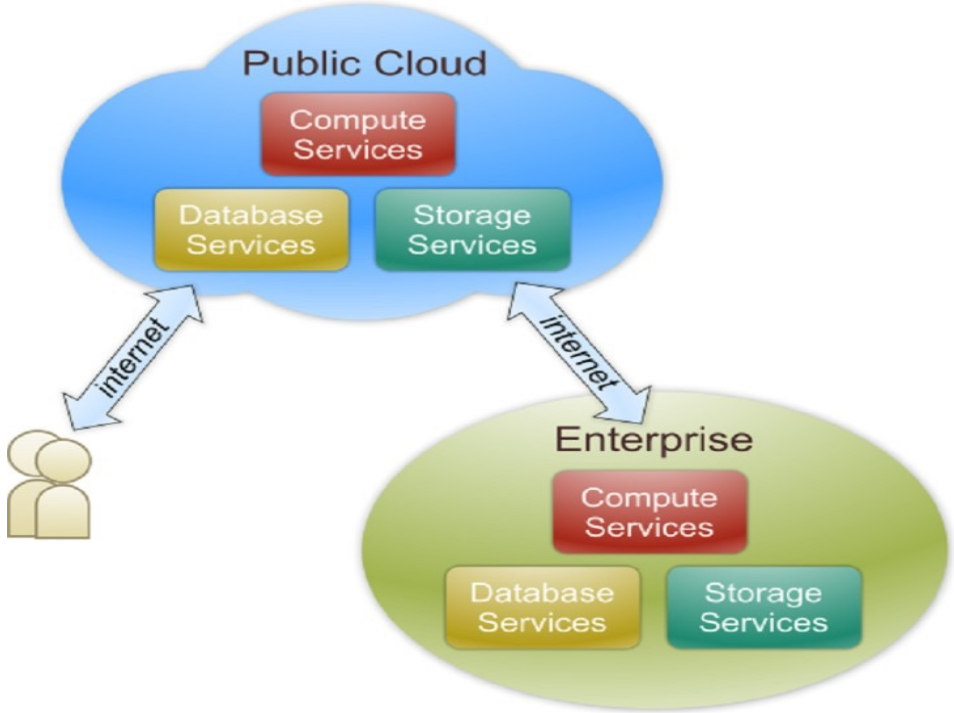
Cloud Deployment Models

Cloud Deployment models

- Public cloud
- Private cloud
- Hybrid cloud
- Community cloud

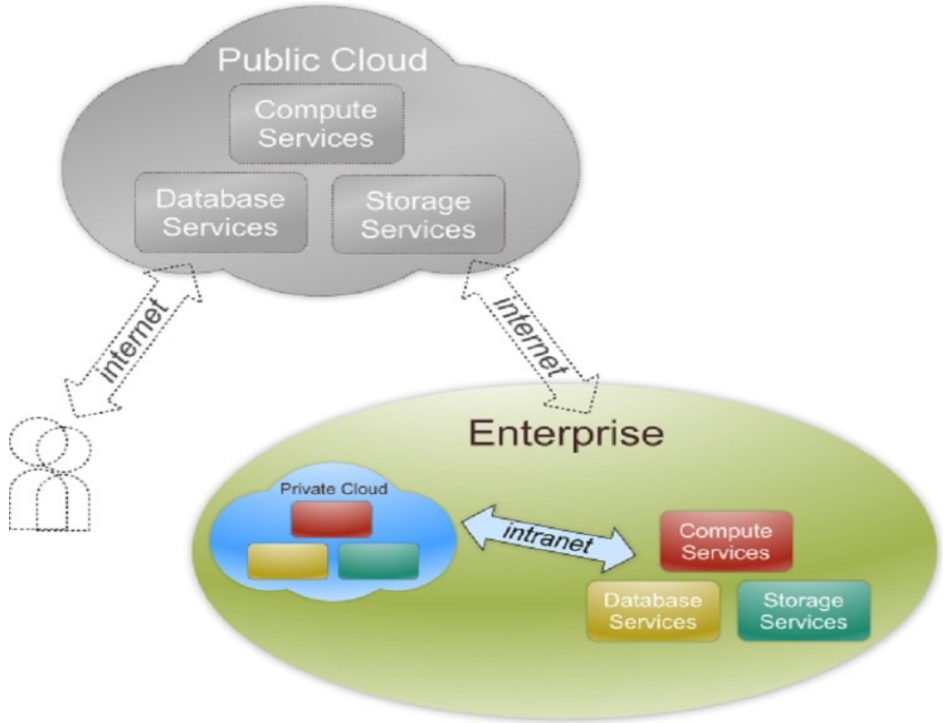
Public Cloud

- Infrastructure is made available to the general public.
- Owned by an organization selling cloud services.



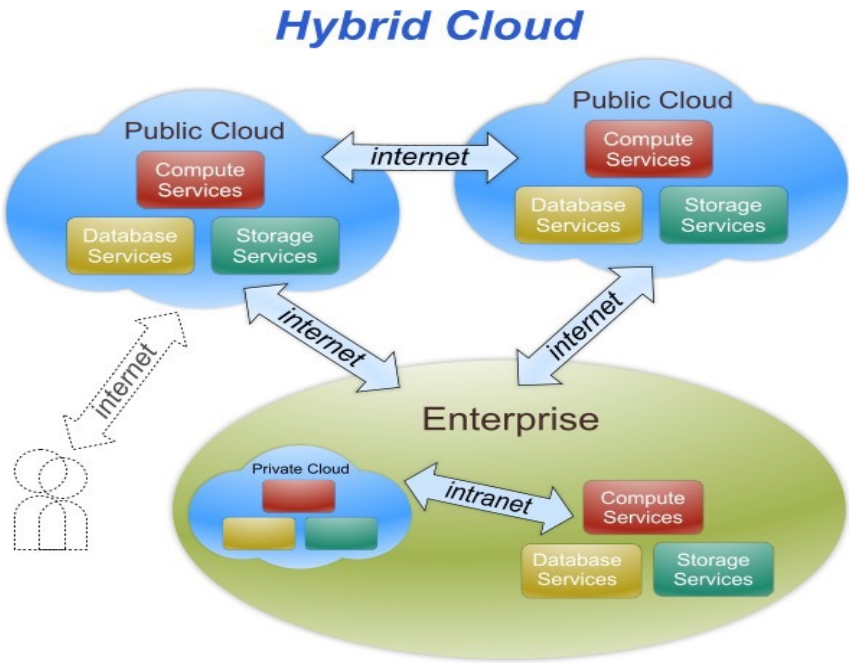
Private cloud

- Infrastructure is operated solely for an organization.
- Managed by the organization or by a third party.



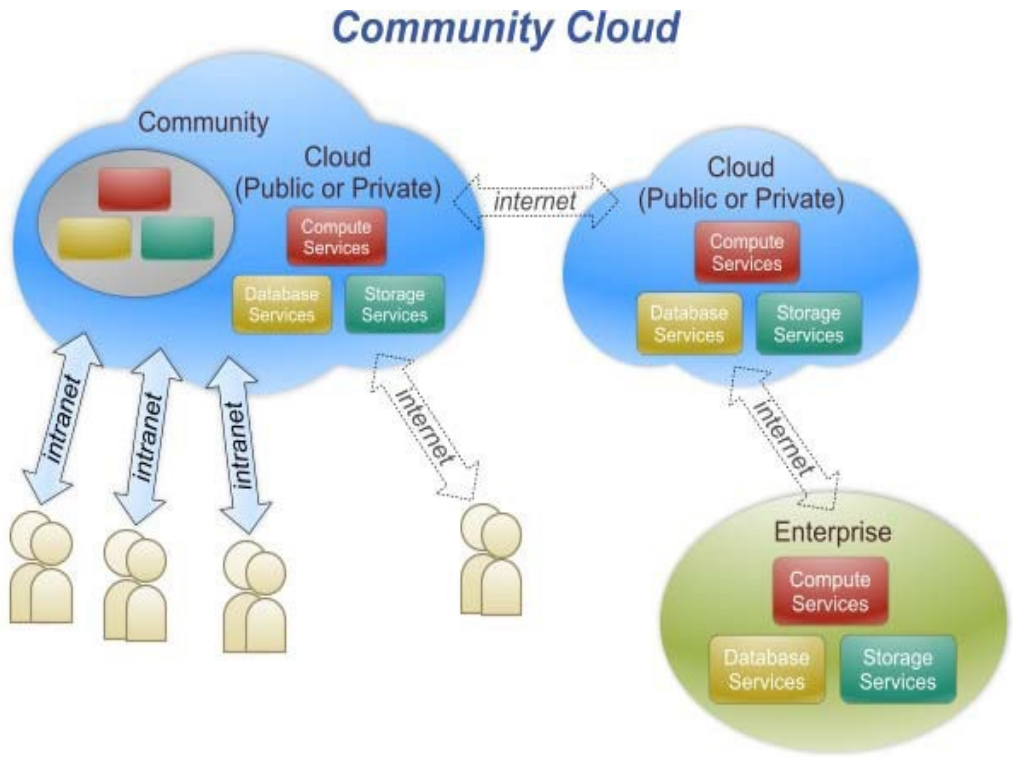
Hybrid Cloud

- Infrastructure is a composition of two or more clouds deployment models.
- Enables data and application portability.



Community Cloud

- Supports a specific community with common.
- Infrastructure is shared by several organizations.



Cloud Main Services

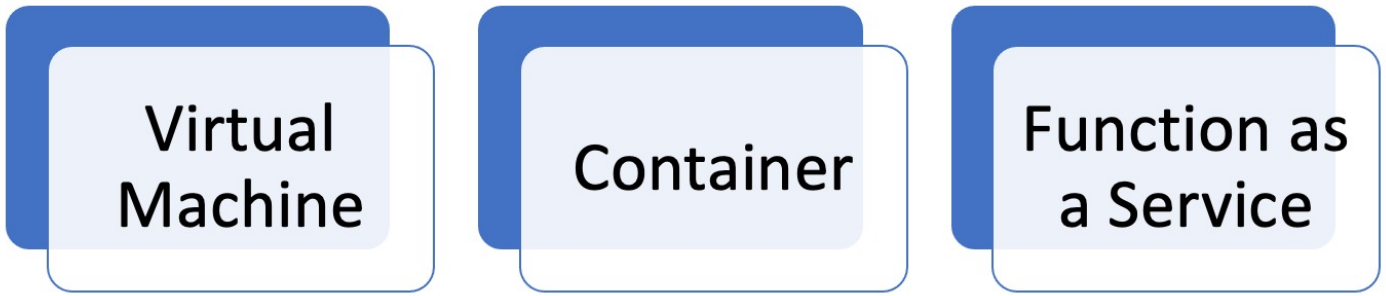
- Computing (virtual machines, containers, serverless, …)
- Storage (file, block, object, …)
- Database (RDBMS, NoSQL, …)
- Big data analytics
- …

Cloud Computing Services

How to deliver a computing service?

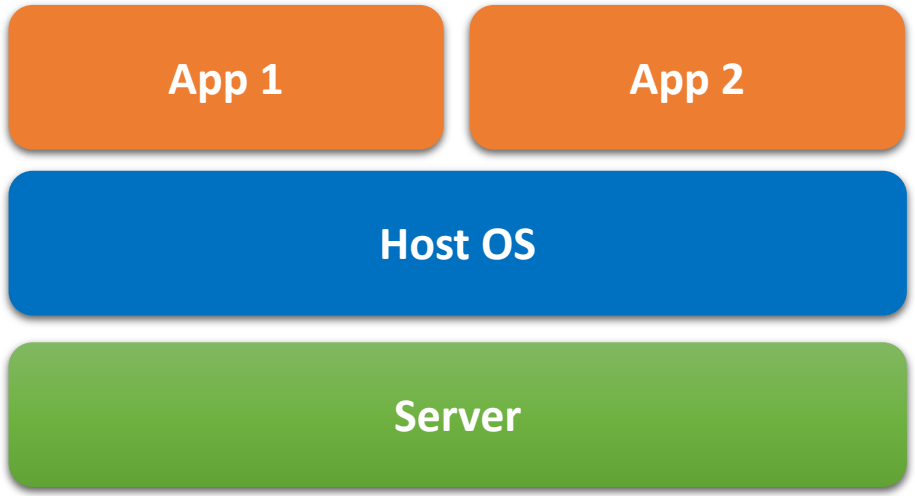
- Traditional:

- Cloud-based:



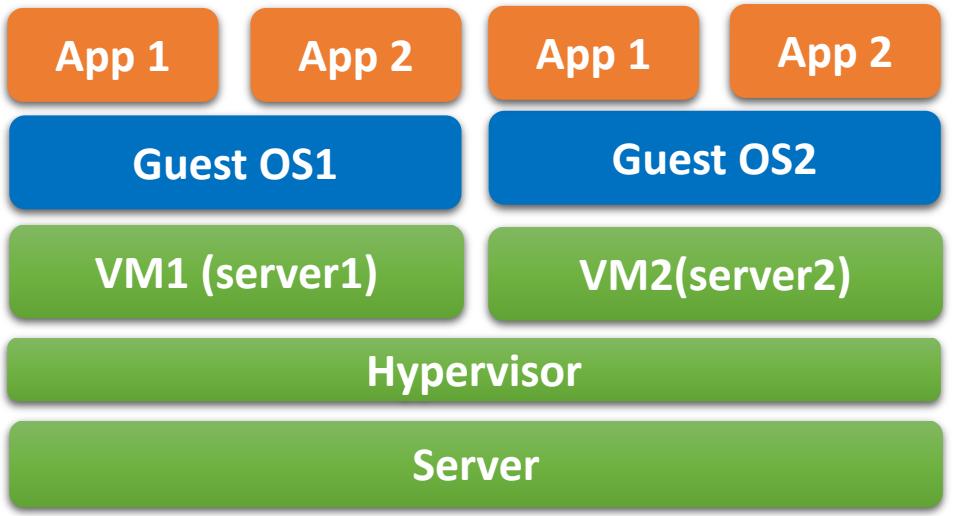
What is Bare Metal?

- Bare metal means a **physical machine/server**.
- One entire server was dedicated to a **single** Operating System.
- It was impossible to partition a server for running multiple OSes, so it wasted many server resources.



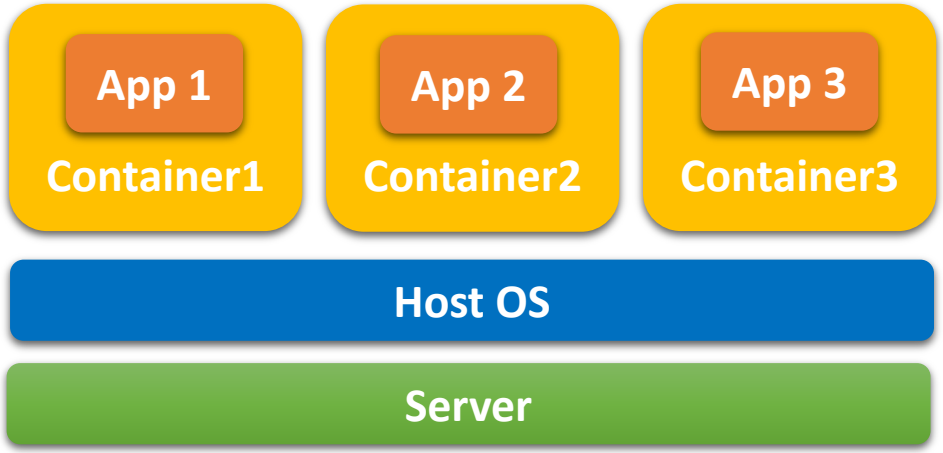
What is a Virtual Machine (VM)?

- A virtual machine (VM) is a **software implementation of a physical server** that runs an operating system and applications.
- VM uses the **hypervisor.**
- One physical server can turn into multiple servers, each having its **dedicated resources**.



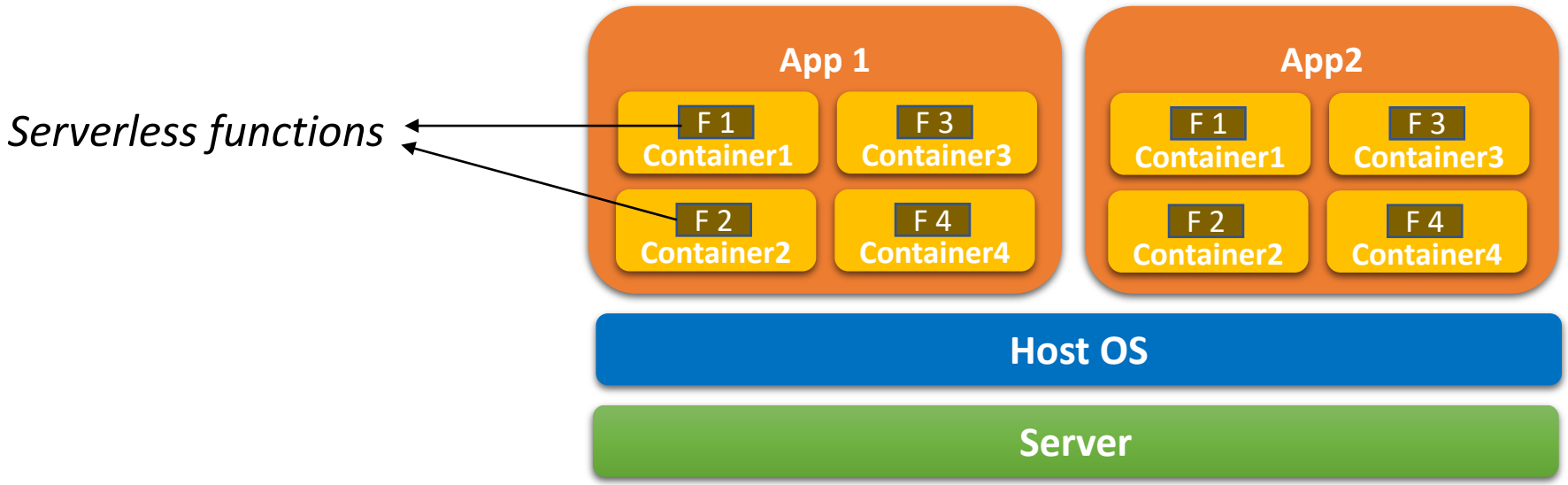
What is a Container?

- Container is a method of **operating system virtualization** that can be used to run an application and its dependencies in resource-isolated processes.
- A container is a **lightweight alternative** to a VM that does not need any Guest OS to run or hypervisor. It makes use of the host operating system.

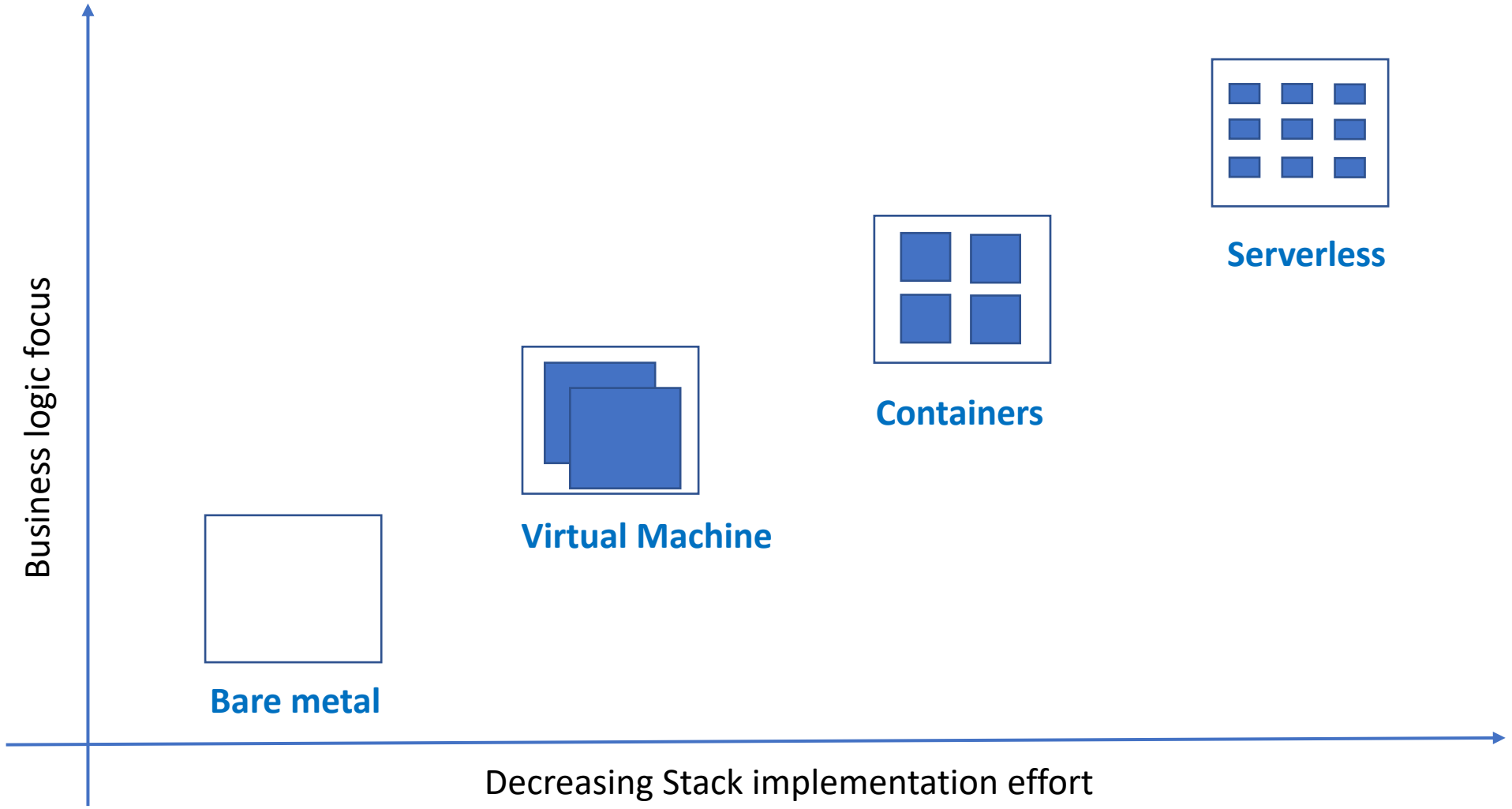


What is Serverless?

- Serverless is a cloud computing model where cloud providers dynamically manage the allocation and provisioning of computing resources needed to execute code functions.
- Organizations essentially outsource their servers instead of owning and they leverage external, cloud-based servers to run **serverless functions.**



From Bare metal to Serverless



Serverless Computing

Serverless Computing

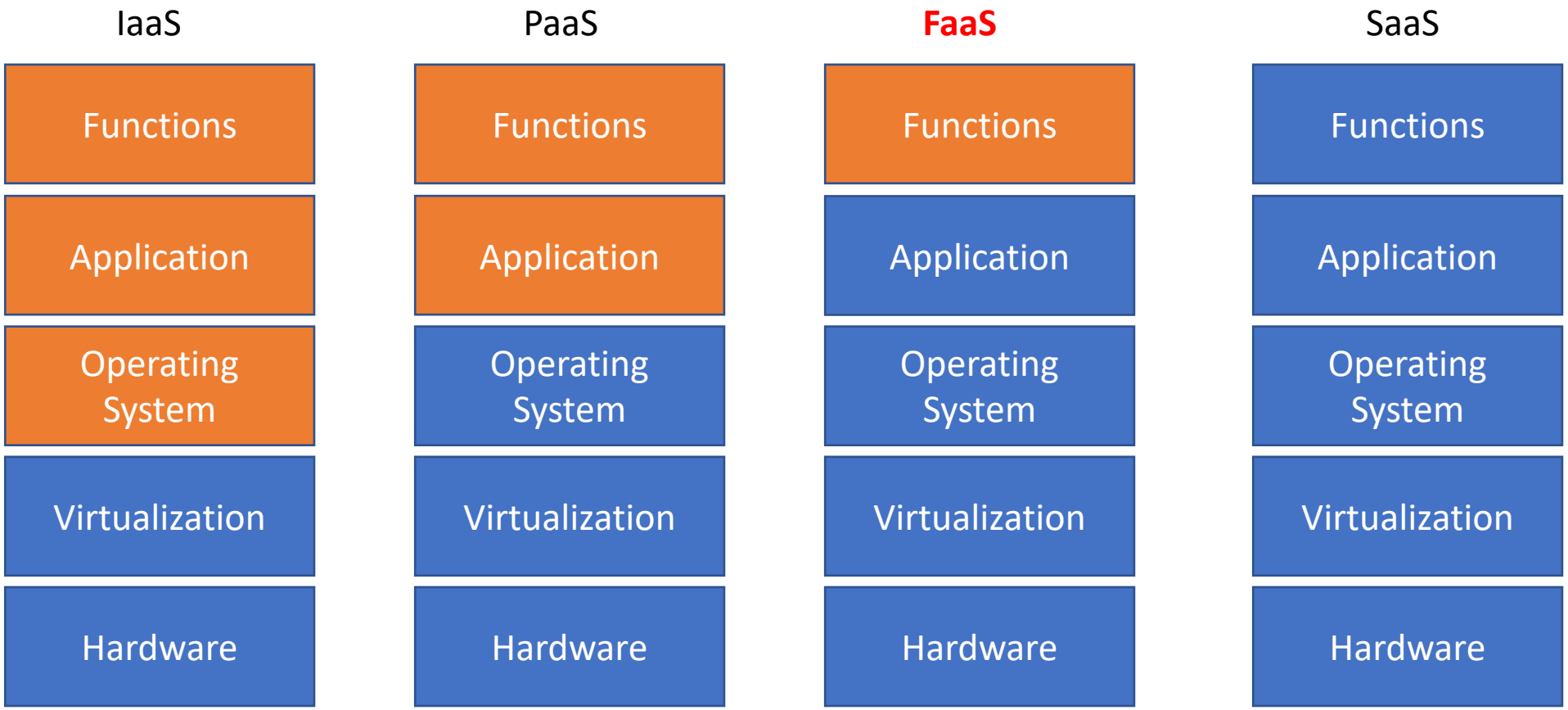
- **Serverless computing** is a cloud computing service model in which the cloud provider allocates machine resources on demand, taking care of the servers on behalf of their customers.

- "Serverless" is a misnomer in the sense that servers are still used by cloud service providers to execute code for developers. However, developers of serverless applications are not concerned with capacity planning, configuration, management, maintenance, fault tolerance, or scaling of containers, VMs, or physical servers.

- Serverless uses functions as the deployment unit, so it is also called **Function-as-a-Service (FaaS)**.

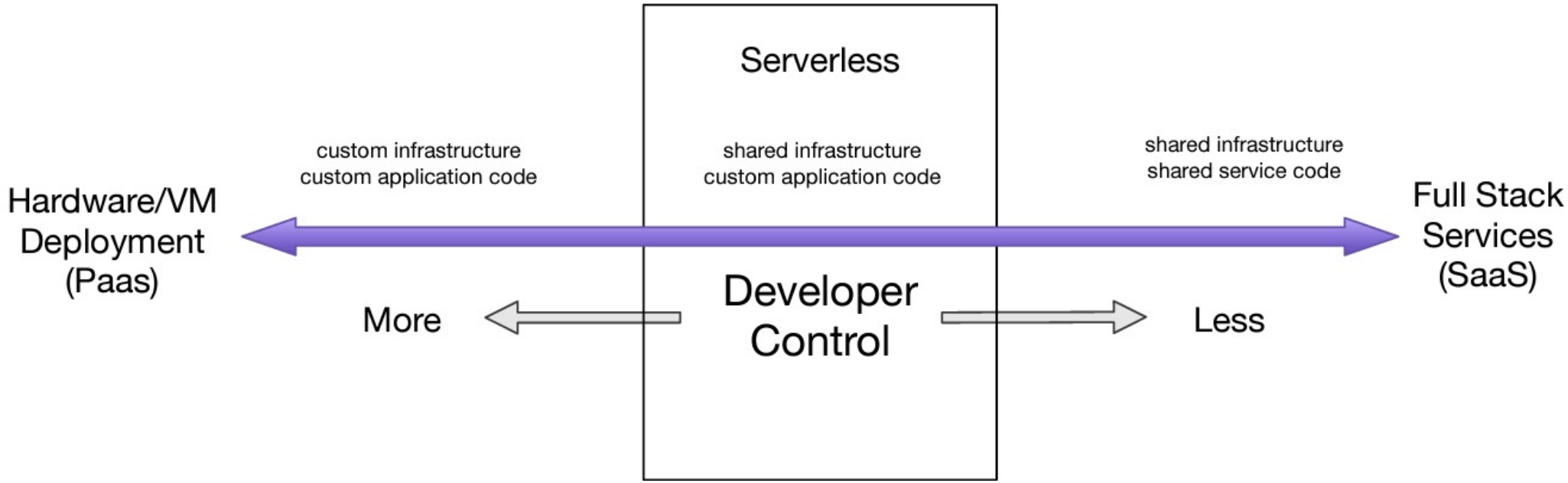
https://en.wikipedia.org/wiki/Serverless_computing

FaaS Vs. Other Service Delivery Models



Developer Control in Serverless

- Serverless can be explained by varying level of developer control over the cloud infrastructure.



An Analogy

Scenario: everyday, dinner party at your home, number of guests could be between 1-20, no one RSVPs!

- **Traditional Data Center**
  - You own the kitchen
  - You own the appliances
  - You pay for the electricity and ingredients
  - You make food for 20 people everyday
  - **Lots od wasted food**
- **Virtual server (VM)**
  - You do NOT own kitchen
  - you get food from takeout place who delivers instantly
  - Take out place only delivers to 5 people at a time
  - If 3 people show up, you waste food of two people
  - If 7 people show up, you place two orders, still 3 wastes.
  - **Better than traditional one but still little wasteful**
- **Serverless**
  - You do NOT own kitchen
  - you get food from takeout place who delivers instantly
  - Take out place accepts order for any number of people showed up
  - **Best cost optimized solution**

An Analogy

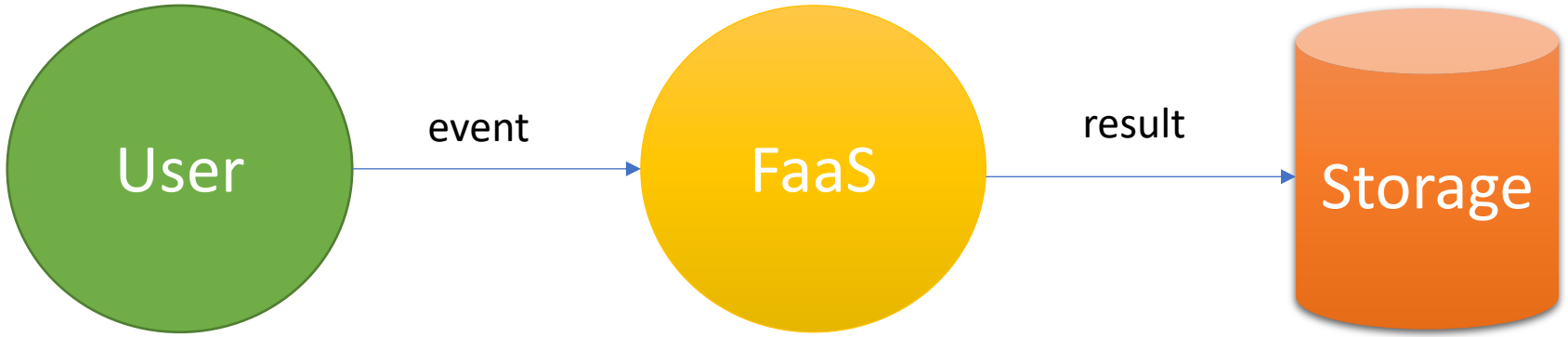
Scenario: everyday, traffic hits your website, and traffic rate is not predictable during the day, but you know the pick amount of traffic!

- **Traditional Data Centre**
  - You own the data center
  - You spend money to buy each physical server
  - You pay for the electricity, AC, ..
  - You buy enough servers to accommodate the pick of traffic
  - **Lots od wasted $$$**
- **Virtual server (VM)**
  - You do NOT own data center
  - You provision VM instantly
  - Each VM comes with fixed processing power and memory
  - If VM reaches its capacity you provision other VMs via autoscaling
  - **Better than traditional data center but still little wasteful**
- **Serverless**
  - You do NOT own data center
  - You utilize serverless functions
  - If more traffic hits it will scale automatically
  - **Best cost optimized solution**

Alexa

Event-driven Architecture

- Runs code in response to an event

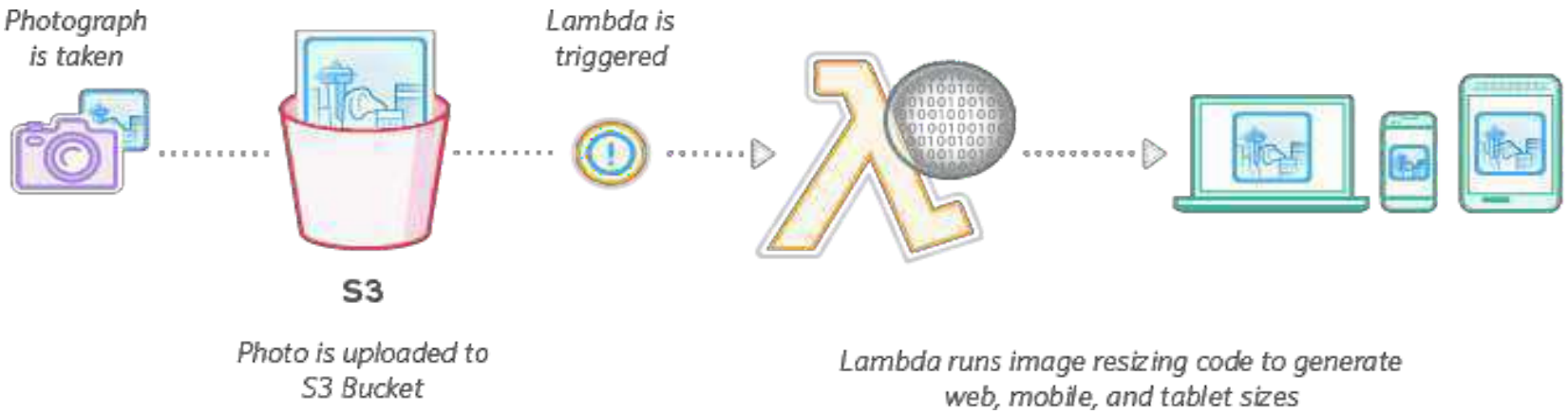


What is an Event?

- Changes in data state
  - E.g., place a new order, post a new message,…
- Request to endpoints
  - E.g., send an HTTP request
- Changes in resource states
  - E.g., upload a new file, delete a file,…
- Call serverless-function from your code

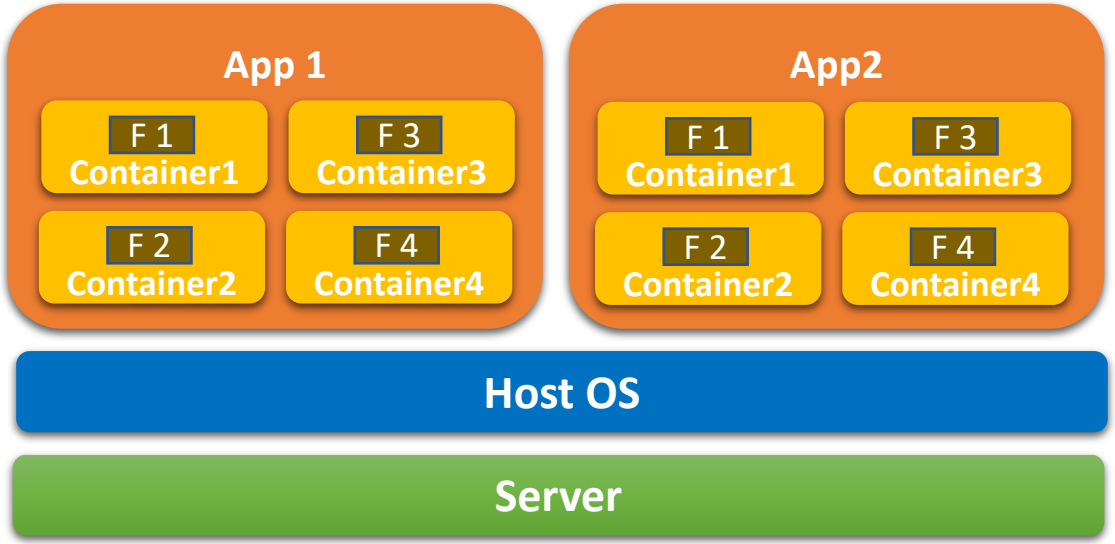
Example - File upload system

- Image processing event handler function by AWS
  - The function is connected to a data store, that emits change events
  - New image file is uploaded, an event is generated.
- In the case of failure , the function can be executed again with no side effects.



Serverless execution environment (1)

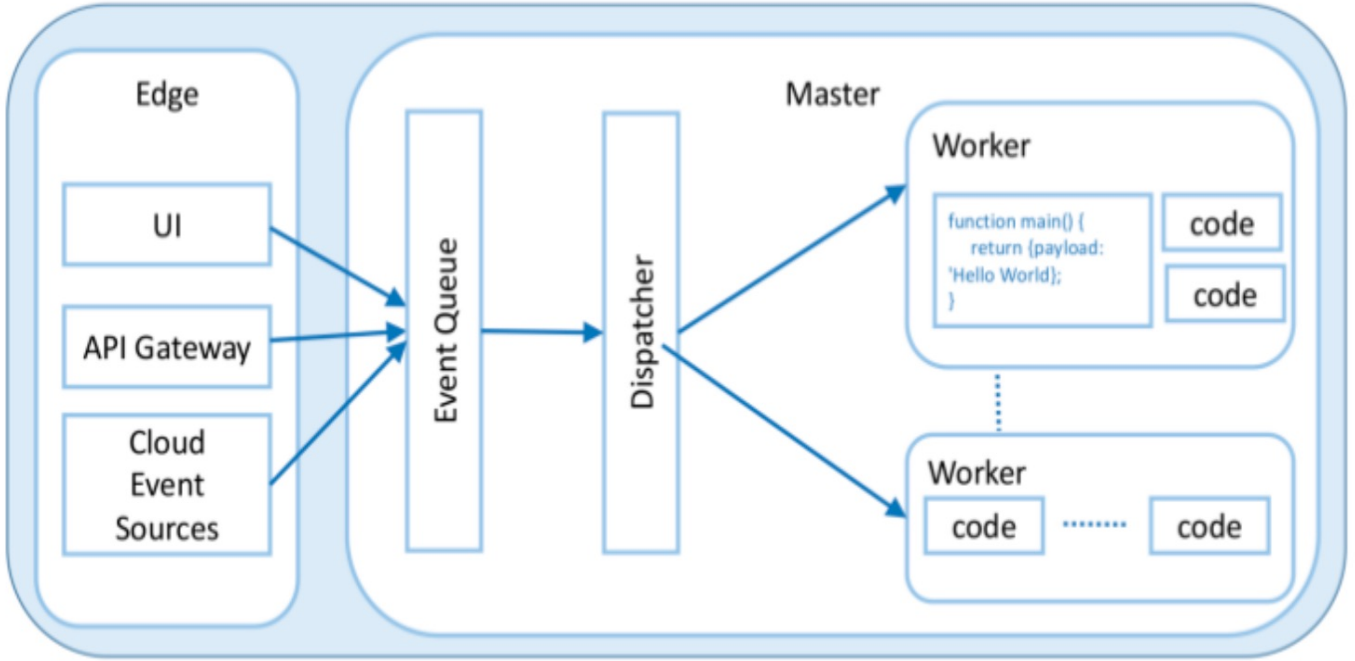
- Function runs inside an operating system container
  - Single OS runs multiple containers
  - Isolation ensures that only processes inside container are visible
  - Has its own file system /tmp
  - Runtimes: C#, Node.js, Java, Python



Serverless execution environment (2)

- A container handles a single function/event at a time
- Concurrent execution by container replication
- Multiples containers of the same function by different events
- Containers may be reused for subsequent function executions
- Containers may be terminated at any time
  - Stateless
  - Store all persistent state outside of container
    - application configuration, databases, files, or session data

Generic FaaS Architecture

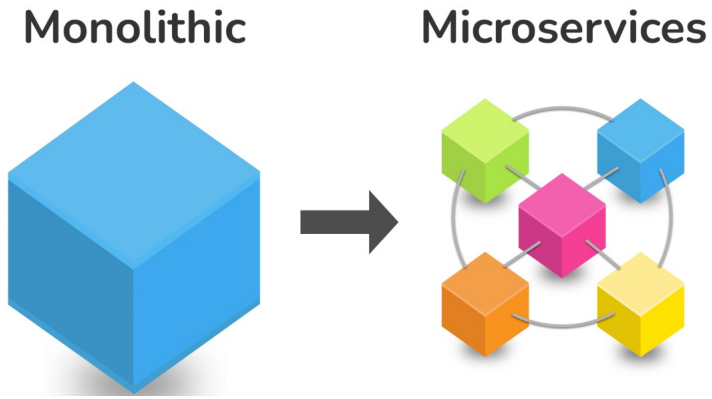


Generic FaaS Architecture

Generic FaaS Architecture

- Edge
  - UI – An UI for the management of functions
  - API Gateway – The general API for the implemented functions
- Event Queue/Dispatcher
  - Event Queue – Manages the triggered Events
  - Dispatcher – Manages the scaling of invocations
- Worker
  - Worker Processes – Execute the function invocations

Serverless Applications



WEB APPLICATIONS

MOBILE APPS

IOT APPLICATIONS

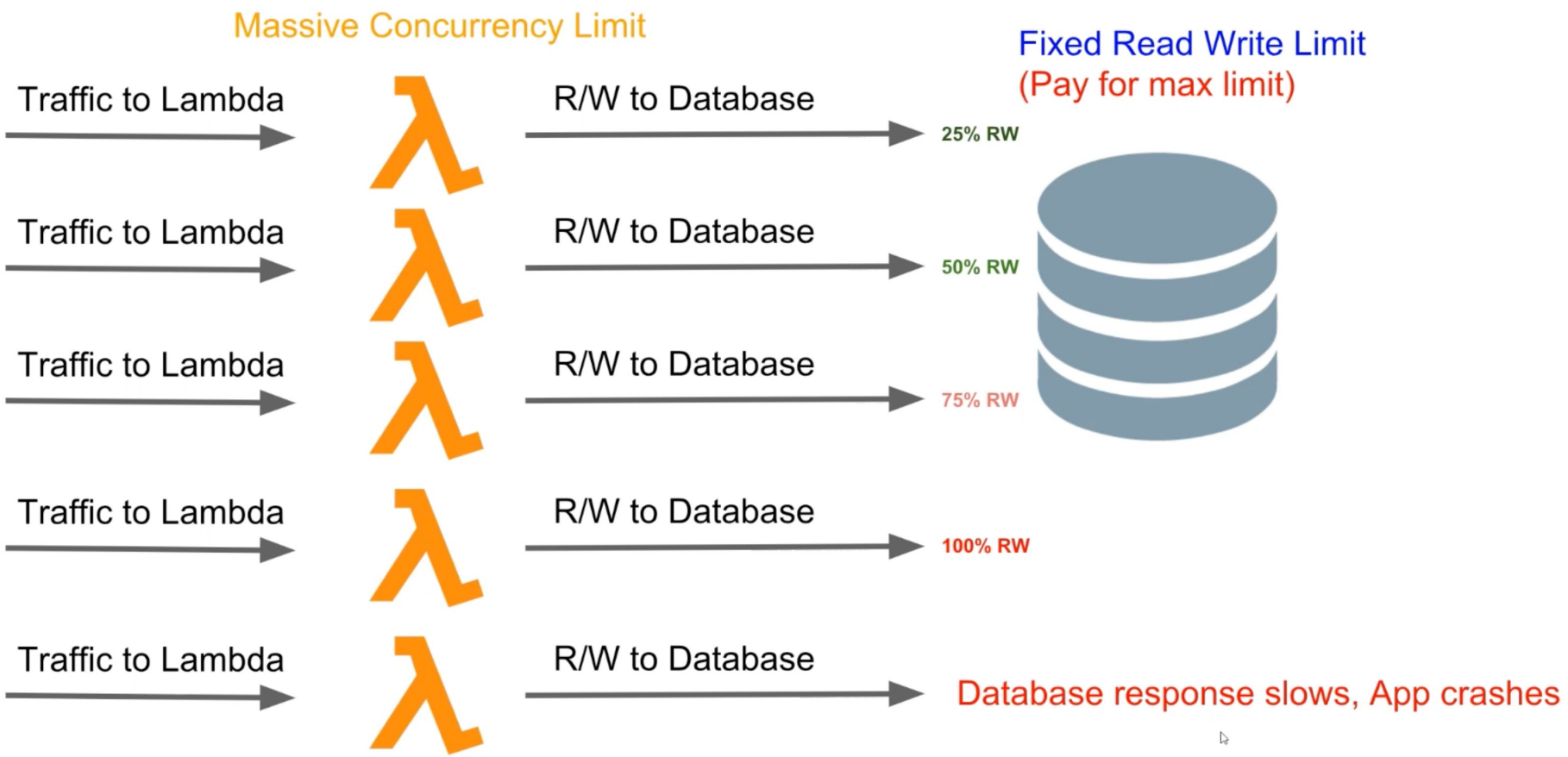
REAL-TIME PROCESSING

BIG DATA PROCESSING

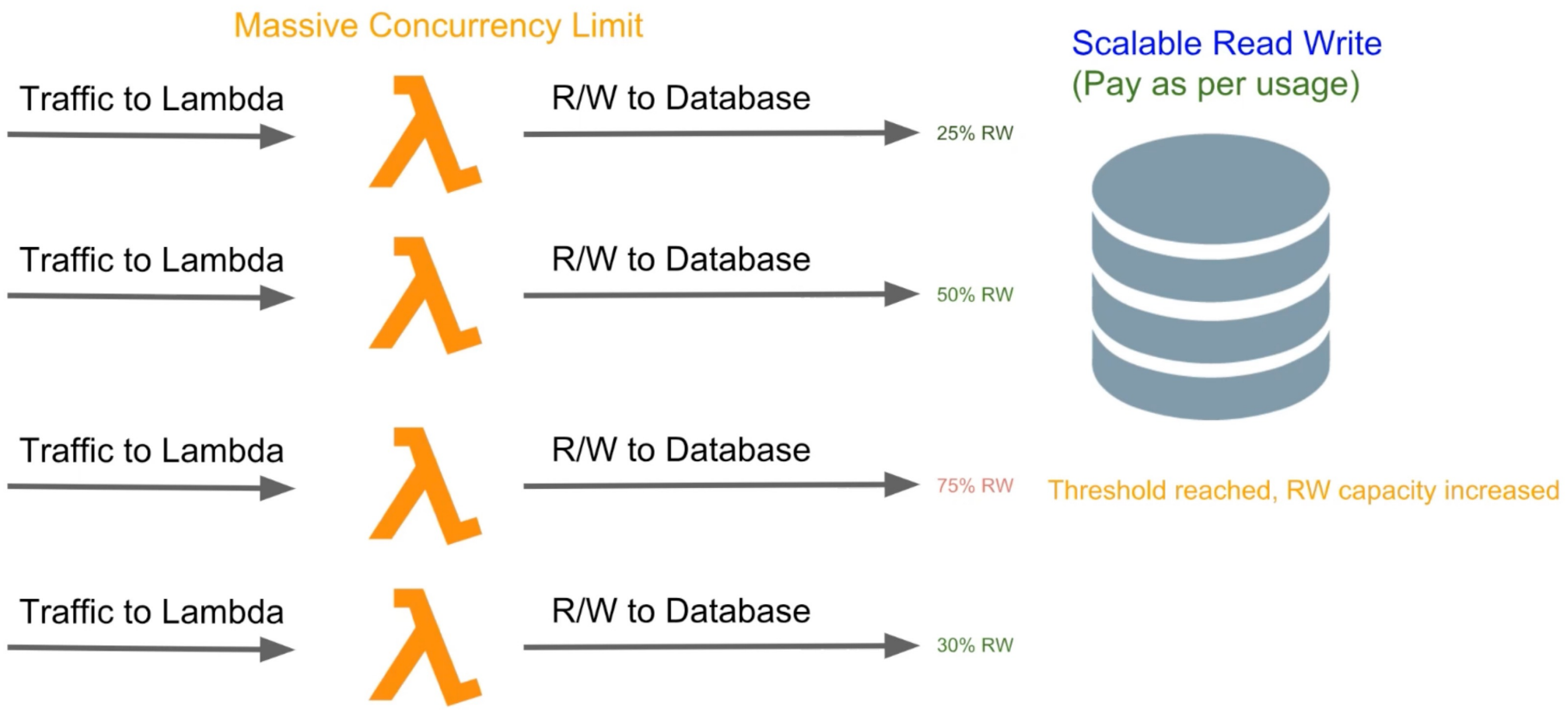
Commercial Serverless Platforms

- Amazon's AWS Lambda
- Google's Cloud Functions
- Microsoft Azure Functions
- IBM Cloud Functions (OpenWhisk)

Scalability issues with traditional database



Ideal Lambda and database behavior



Serverless Security Questions

1. Is my APIs open to the world?
2. Can it be abused without authentication?
3. Is my data encrypted and protected in transit and in rest?
4. How do I access to a trusted entity through my API?
5. How to manage my secret keys?

Serverless Security Considerations

1. Is my APIs open to the world?
   - *Ensure Proper Access Controls*
2. Can it be abused without authentication?
   - *Implement Authentication and Authorization*
3. Is my data encrypted and protected in transit and in rest?
   - *Data Encryption*
4. How do I access to a trusted entity through my API?
   - *Secure Communication with Trusted Services*
5. How to manage my secret keys?
   - *Secret Management like AWS Secrets Manager or Azure Key Vault*

Additional Security Considerations..

1.Least Privilege Principle

2.Implement Function-Based Access Control

3.Secure Environment Variables

4.Implement Function Timeout and Memory Limits

5.Log and monitor

6.Incident Response Plan

7.…

Serverless Recap

- Serverless is an evolution of the trend towards higher levels of abstractions in cloud programming models.
- Independent, logical and event-triggered functions
- Cost : Typically its Pay As You Go
- Stateless and Ephemeral
- Polyglot environment
- Serverless Security considerations

Next Topic:

Containerization