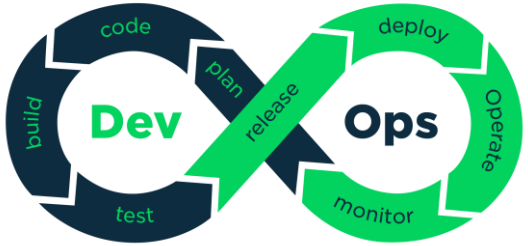
DevSecOps



Raman Khanna

Introduction

Name

Background – Development / Infrastructure / Database / Network

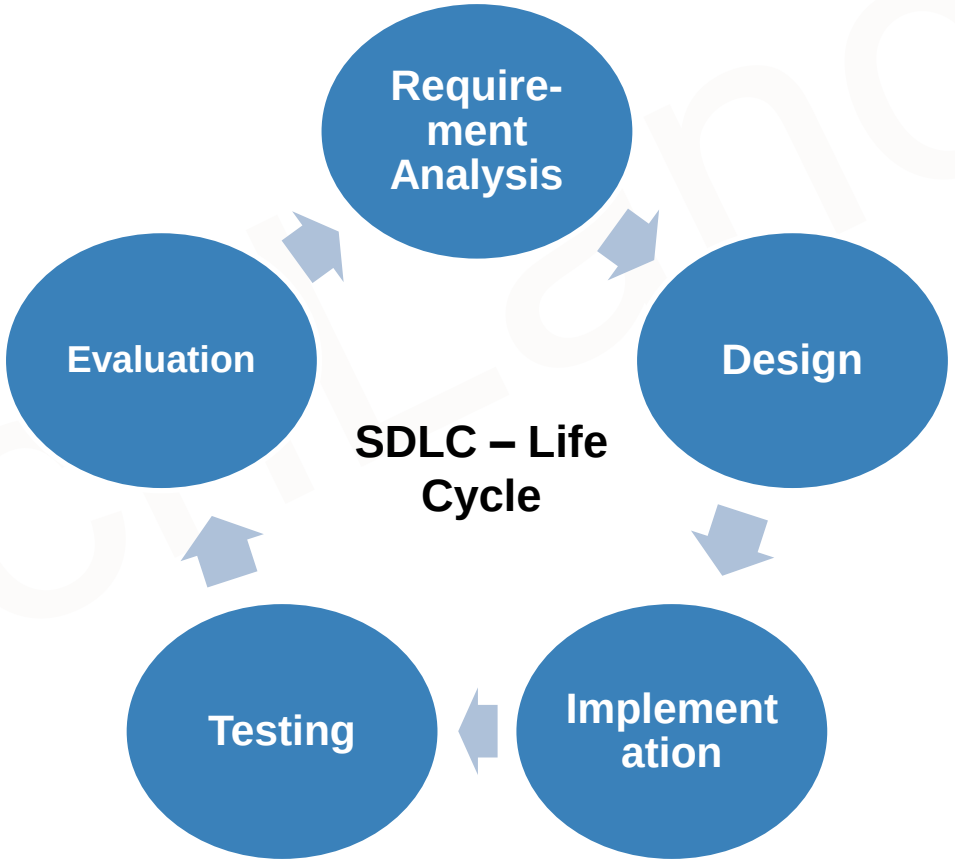
Experience : AWS/Git/Docker/Kubernetes/Jenkins/Terraform

DevOps

What is DevOps?

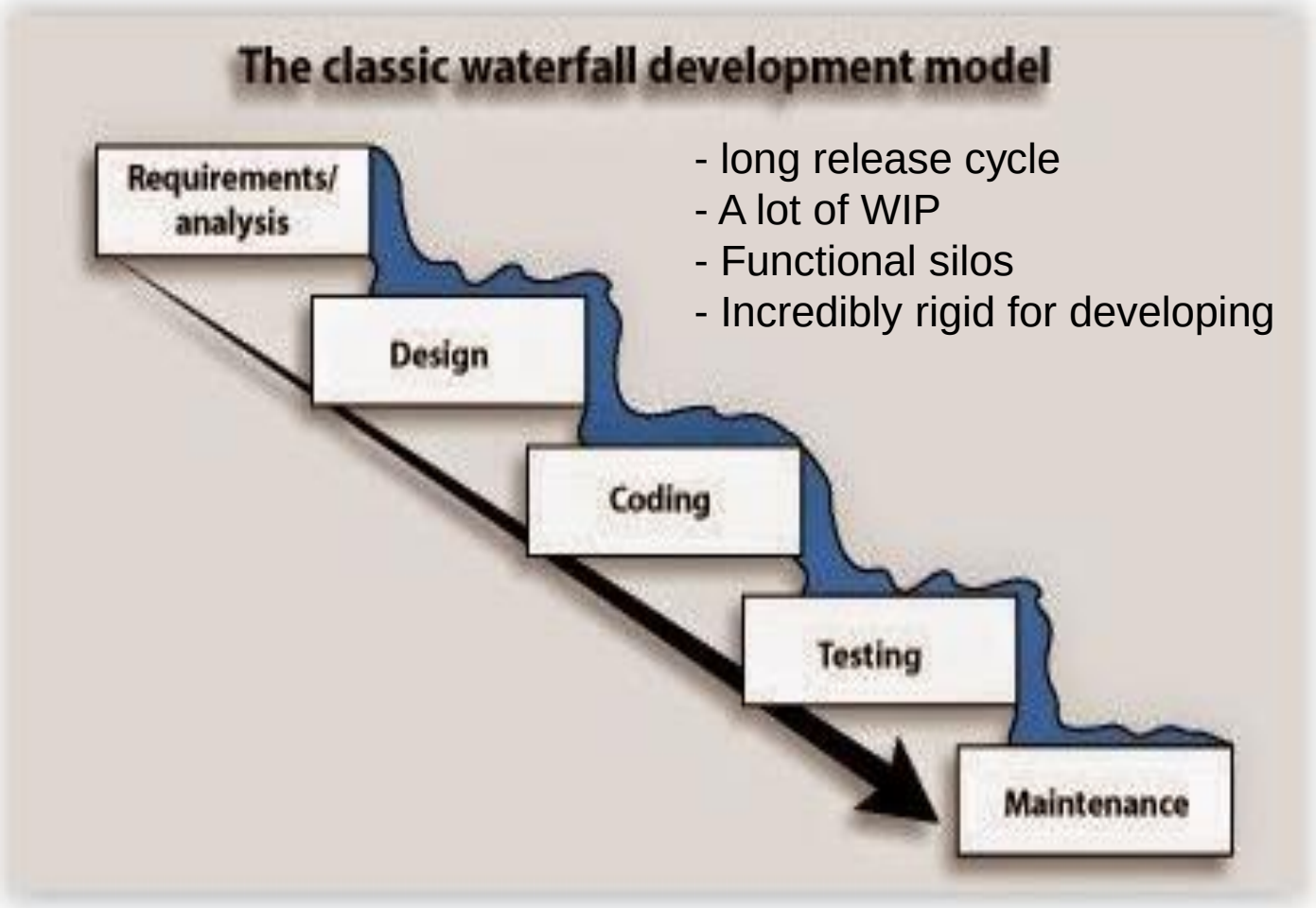
SDLC Model

- A systems development life cycle is composed of **several clearly defined and distinct work phases** which are used by systems engineers and systems developers to plan for, design, build, test, and deliver information systems

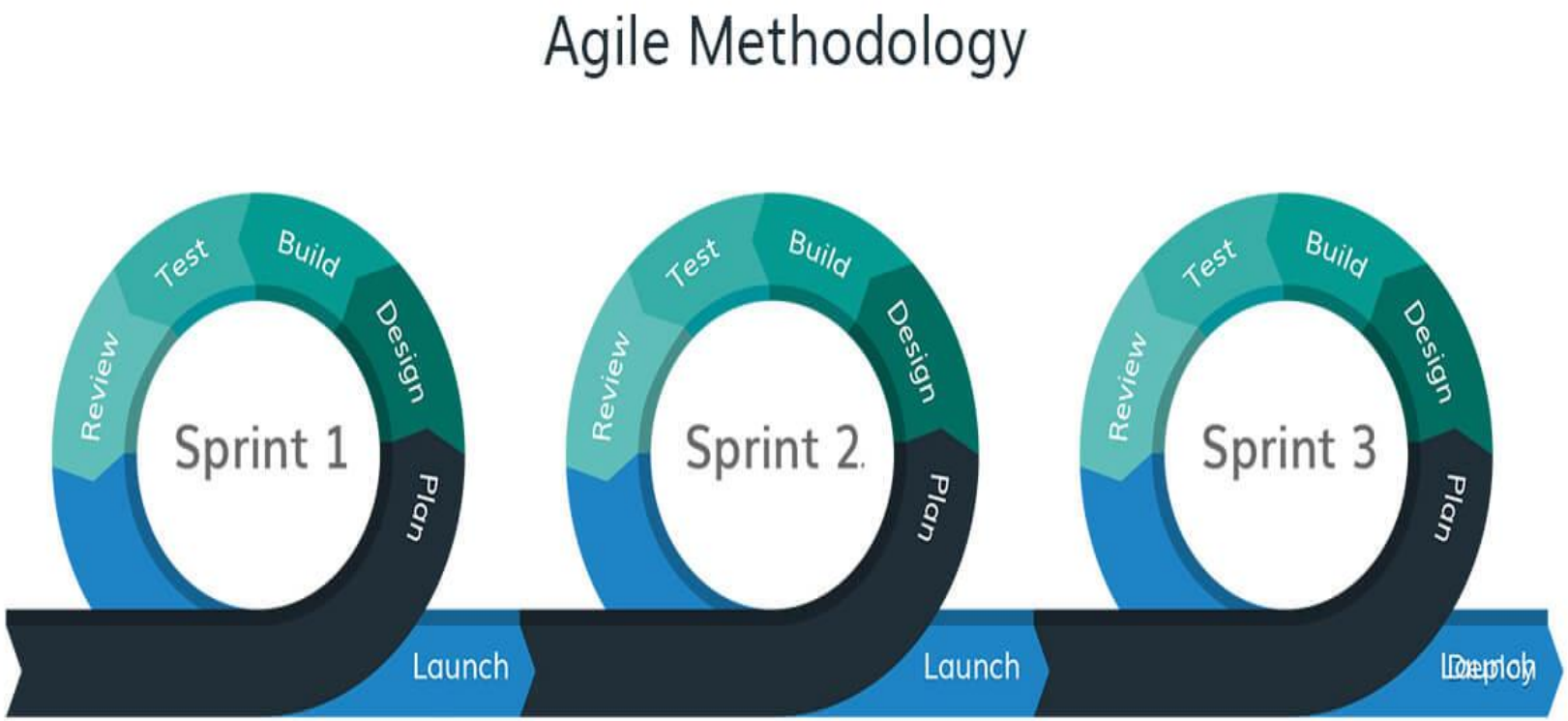


Waterfall Model

1. Determine the Requirements
2. Complete the design
3. Do the coding and testing (unit tests)
4. Perform other tests (functional tests, non-functional tests, Performance testing, bug fixes etc.)
5. At last deploy and maintain

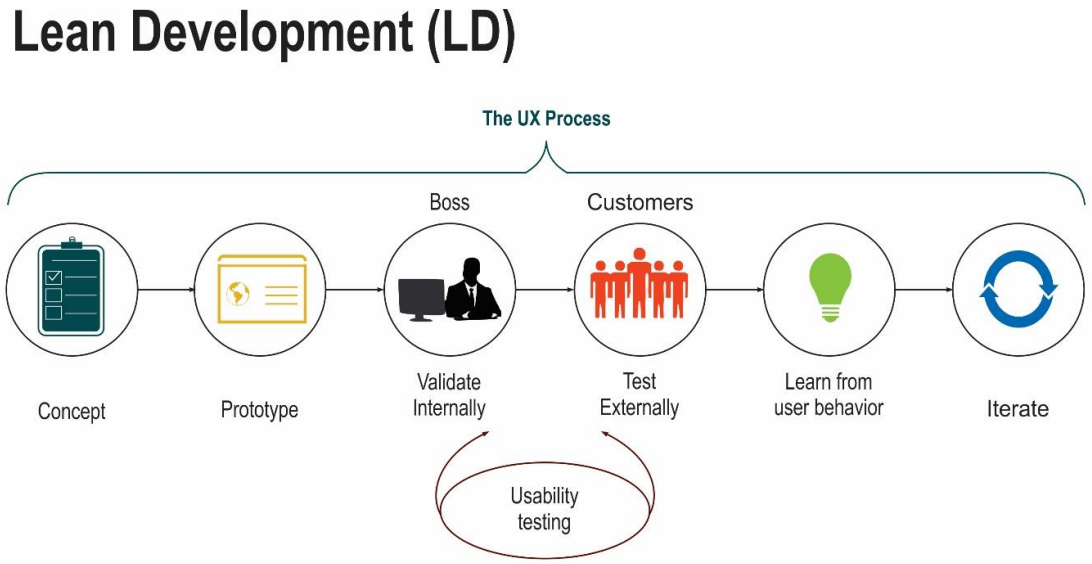


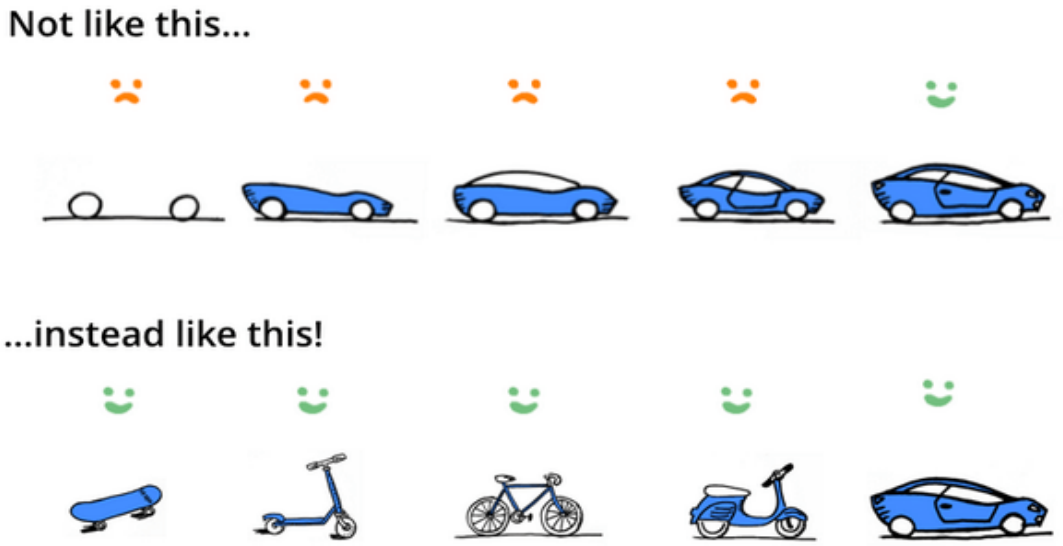
Agile



- Shorter release cycle
- Small batch sizes (MVP)
- Cross-functional teams
- Incredibly agile

Lean Development

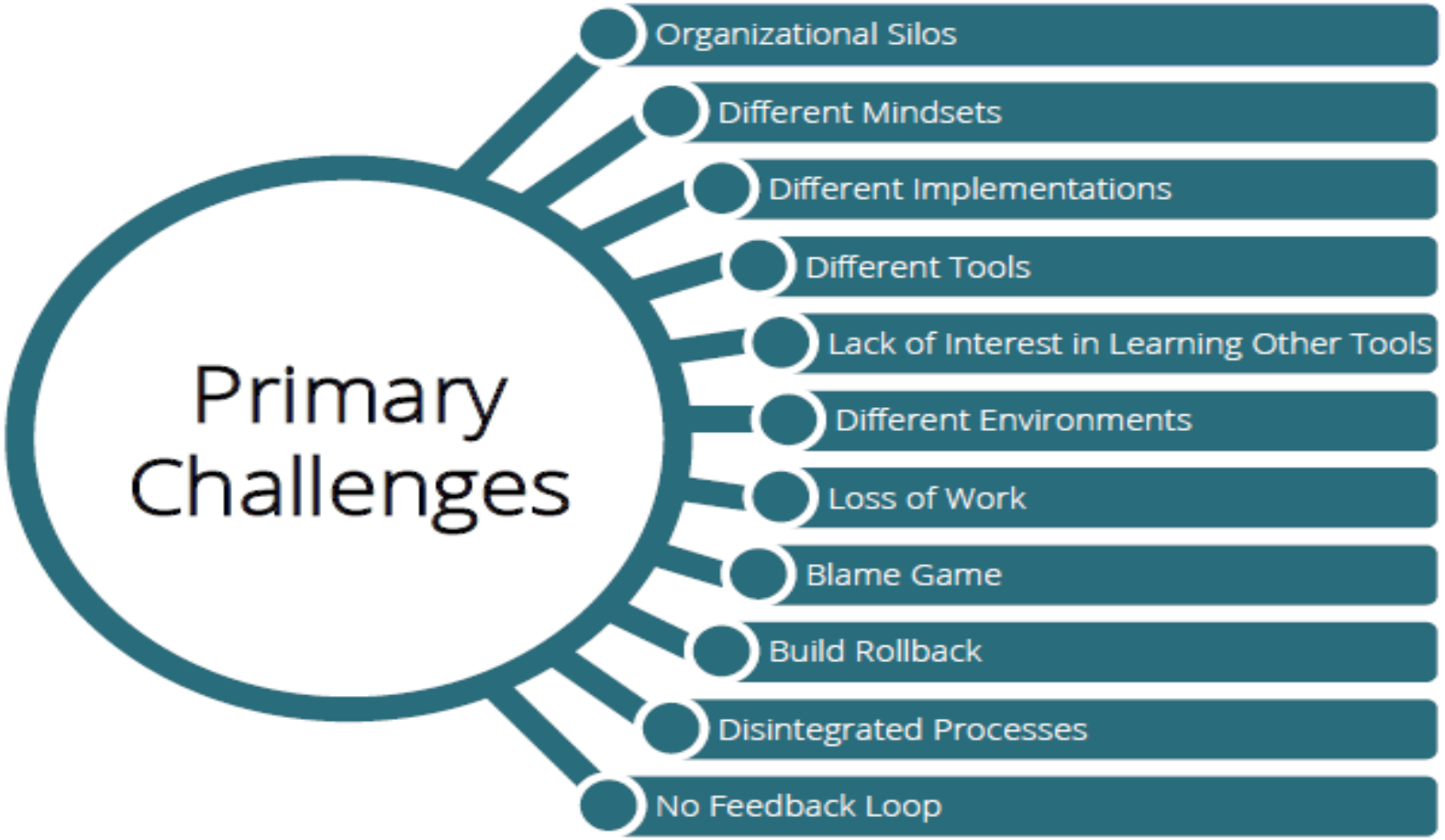




- Suddenly ops was the bottleneck (more release less people), again WIP is more!

Challenges

Some of the challenges with the traditional teams of Development and Operations are:



A Typical Case Study

Development Team:

- Monday Morning, the writing of code done, unit tests completed, code delivered to the Integration teams to get the code included in CI builds.
- To get the services tested, a ticket is opened for QA teams

Build/Release/Testing/Integration Team:

- Tuesday Morning, ticket accepted, a tester put an email to the developer asking deployment instructions. There is not automated deployments, developer updated to the tester, lets come online and we will deploy the services to the QA environment together.
- Call started, developer identified the “test environment” is not compatible.
- Tuesday afternoon, a ticket raised in Ops Team with new specifications.

Ops Team:

- Wednesday morning, ticket accepted, specifications checked , a new port open request was identified.
- Ticket raised for Security team, ticket accepted, change approved, port opened, email received by the Ops team the work is done.

A Typical Case Study

Ops Team:

- Identified the provisioning requirements again and started work on building the environment.

Build/Release/Testing/Integration Team:

- Thursday Morning, updates received - the environment is ready. Developer and Tester again on call to deploy new services. Services deployed; tester is running test scripts. Next phase is to run regression test cases. Again a new ticket is raised for new test data with production teams and day ends.

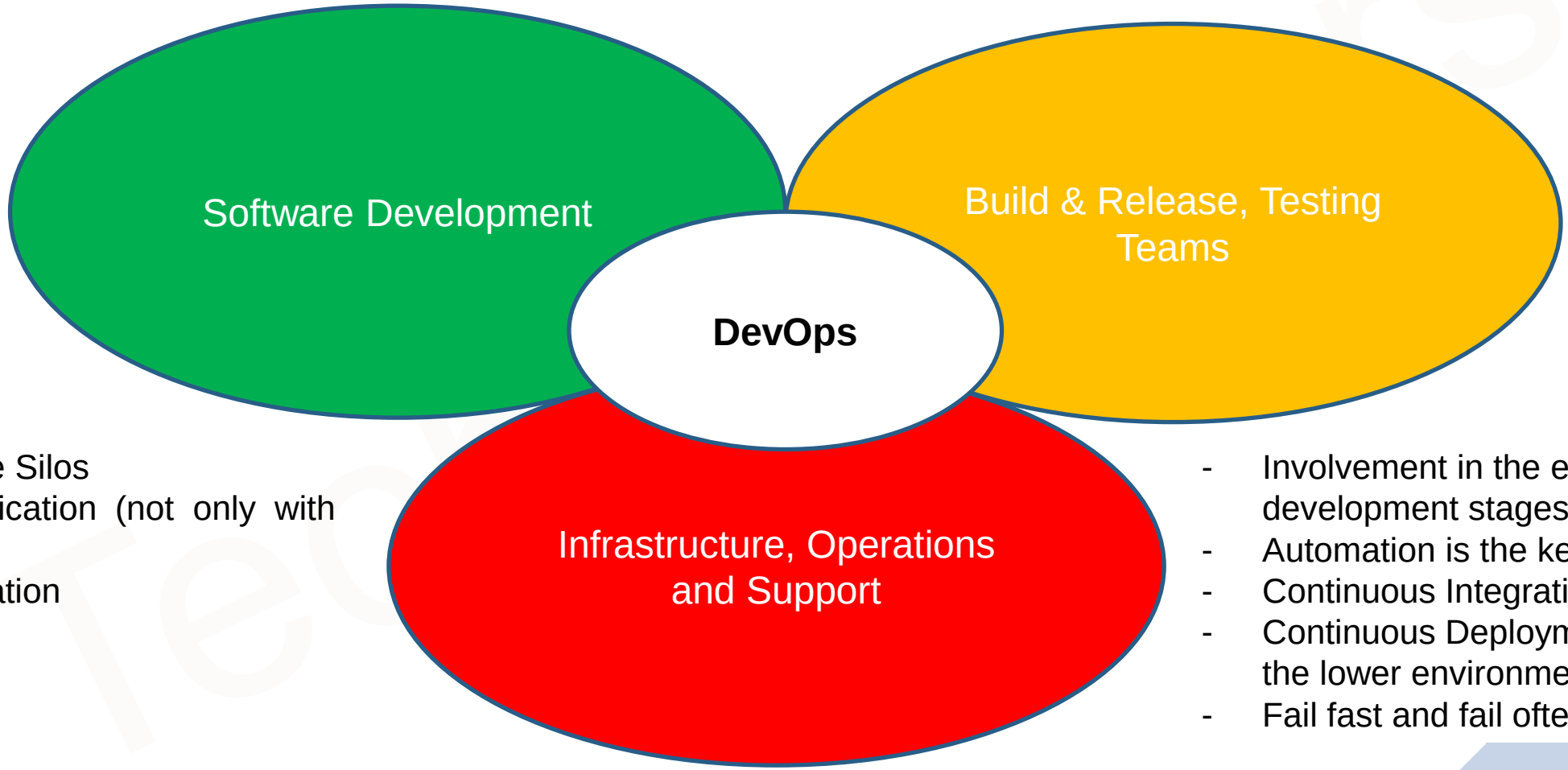
Ops Team:

- Its Friday and the work is not on full swing, ticket accepted but not worked as production team has to complete rest of the works. Somehow the test data is gathered by Friday Evening.

Build/Release/Testing/Integration Team:

- Monday morning, tester gets the data, regression tests run, a defect found, and ticket returned to the development team.

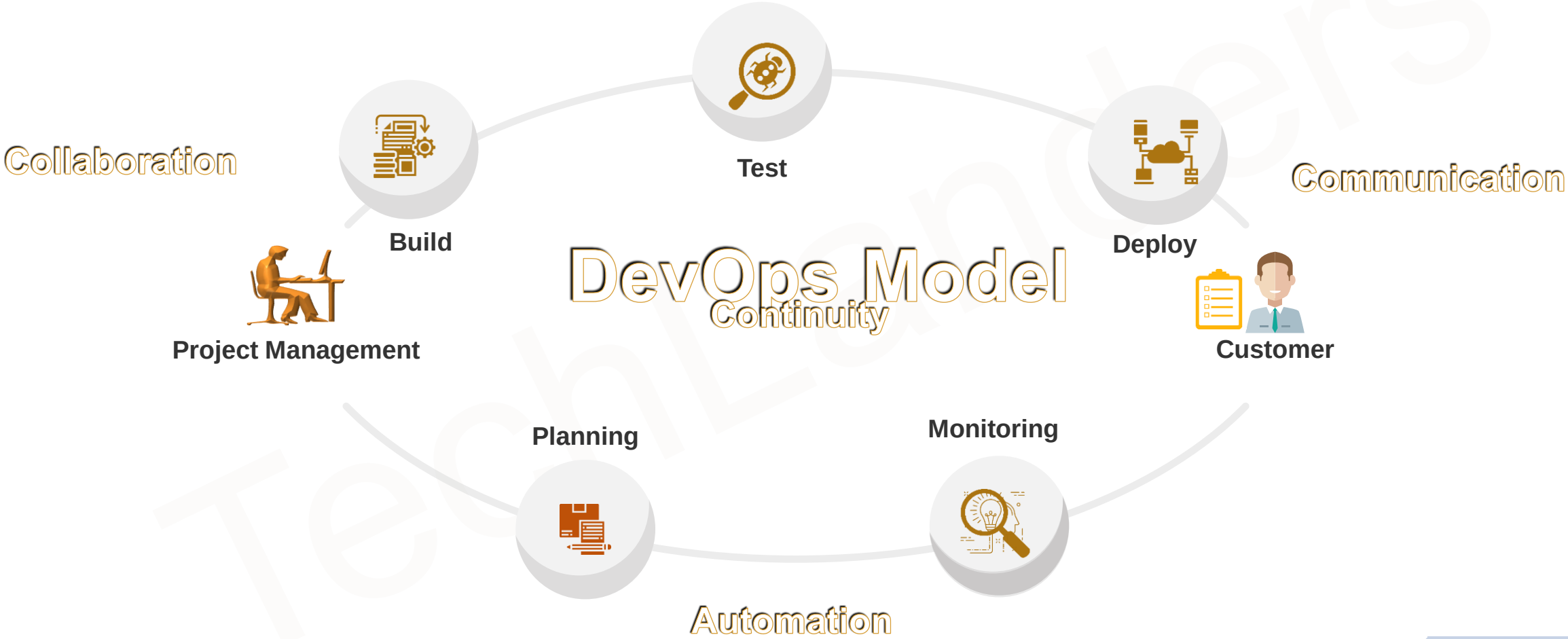
DevOps



- Break the Silos
- Communication (not only with emails)
- Collaboration
- Trust

- Involvement in the early development stages
- Automation is the key
- Continuous Integration
- Continuous Deployments in the lower environments
- Fail fast and fail often

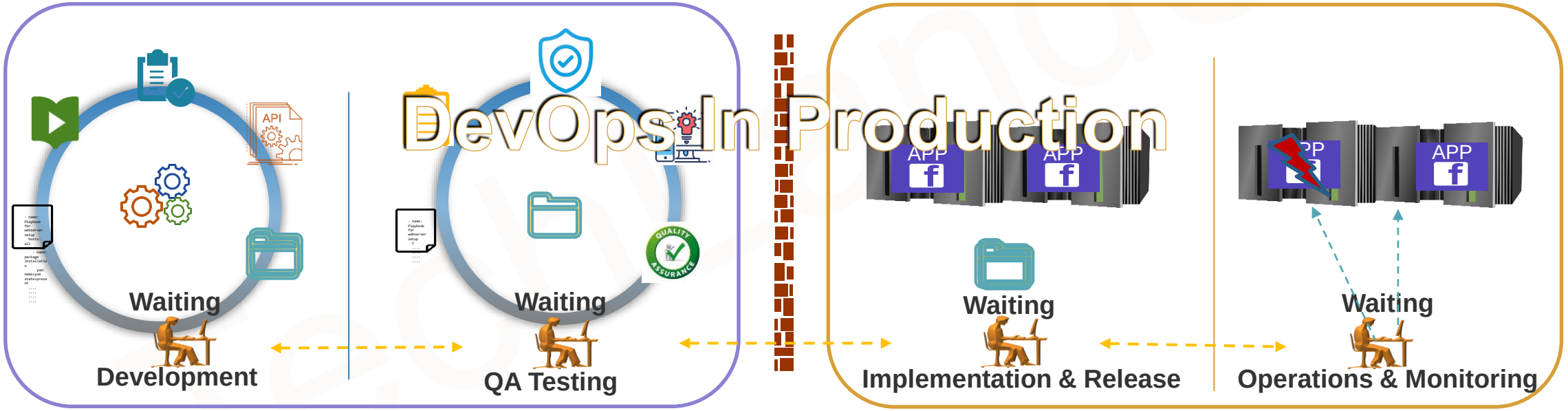
DevOps



DevOps in Action

Continuous Feedback Continuous Improvement Continuous Planning

Continuous Integration Continuous Delivery Continuous Deployment Continuous Monitoring



DevOps Essence

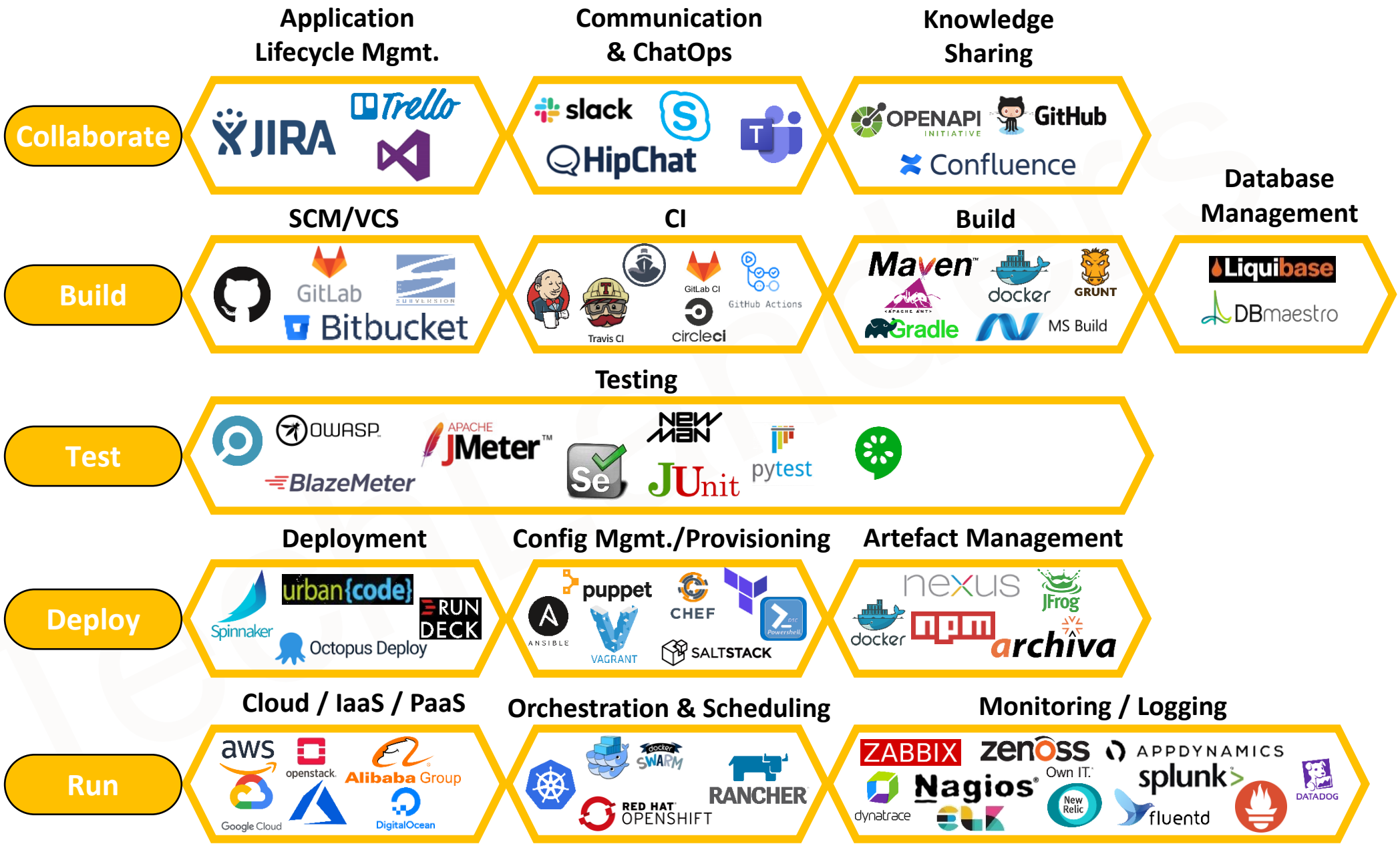
Efficiency - Faster time to market

Predictability - Lower failure rate of new releases

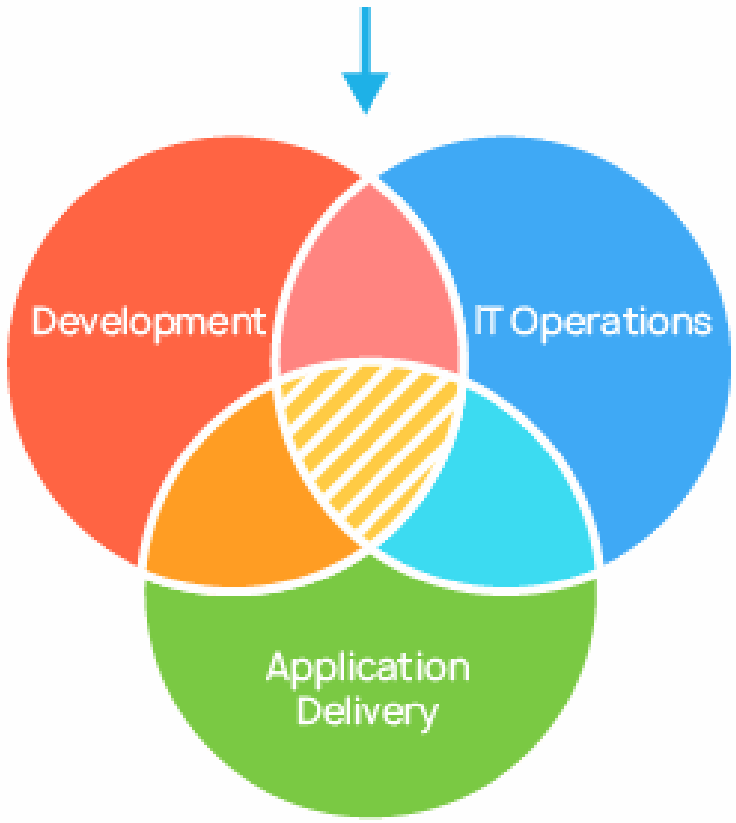
Reproducibility – Version everything

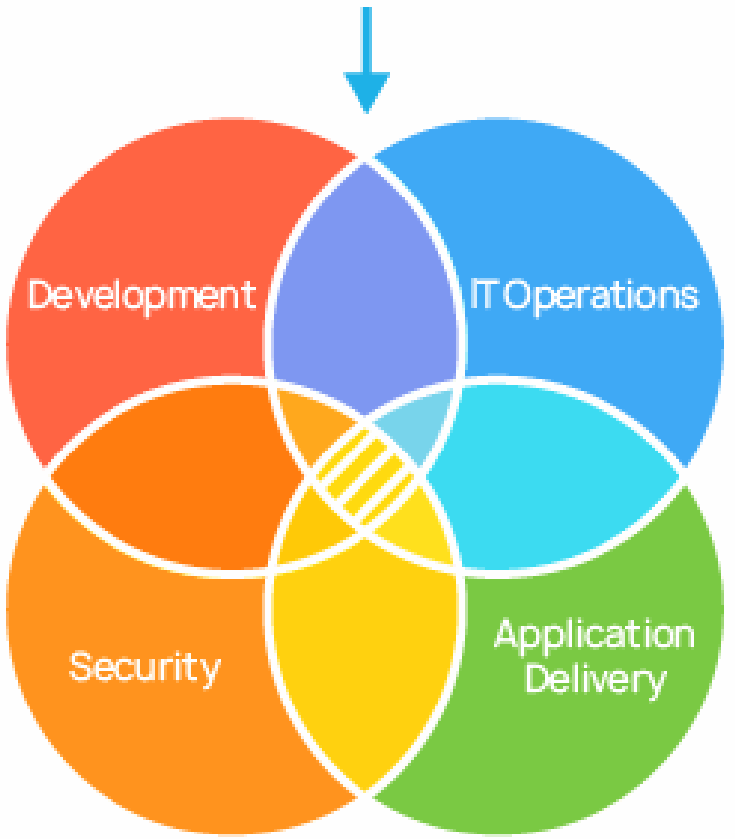
Maintainability - Faster time to recovery in the event of a new release crashing or otherwise disabling the current system

DevOps Toolsets

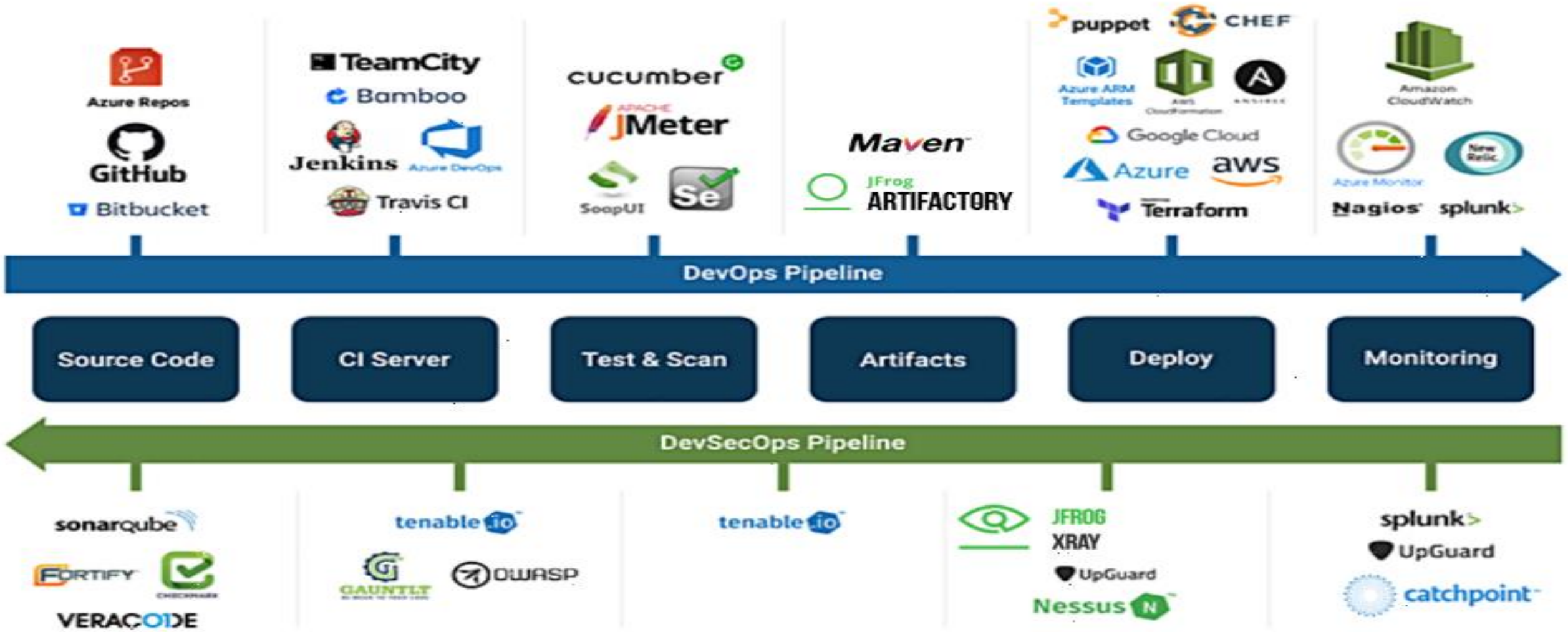


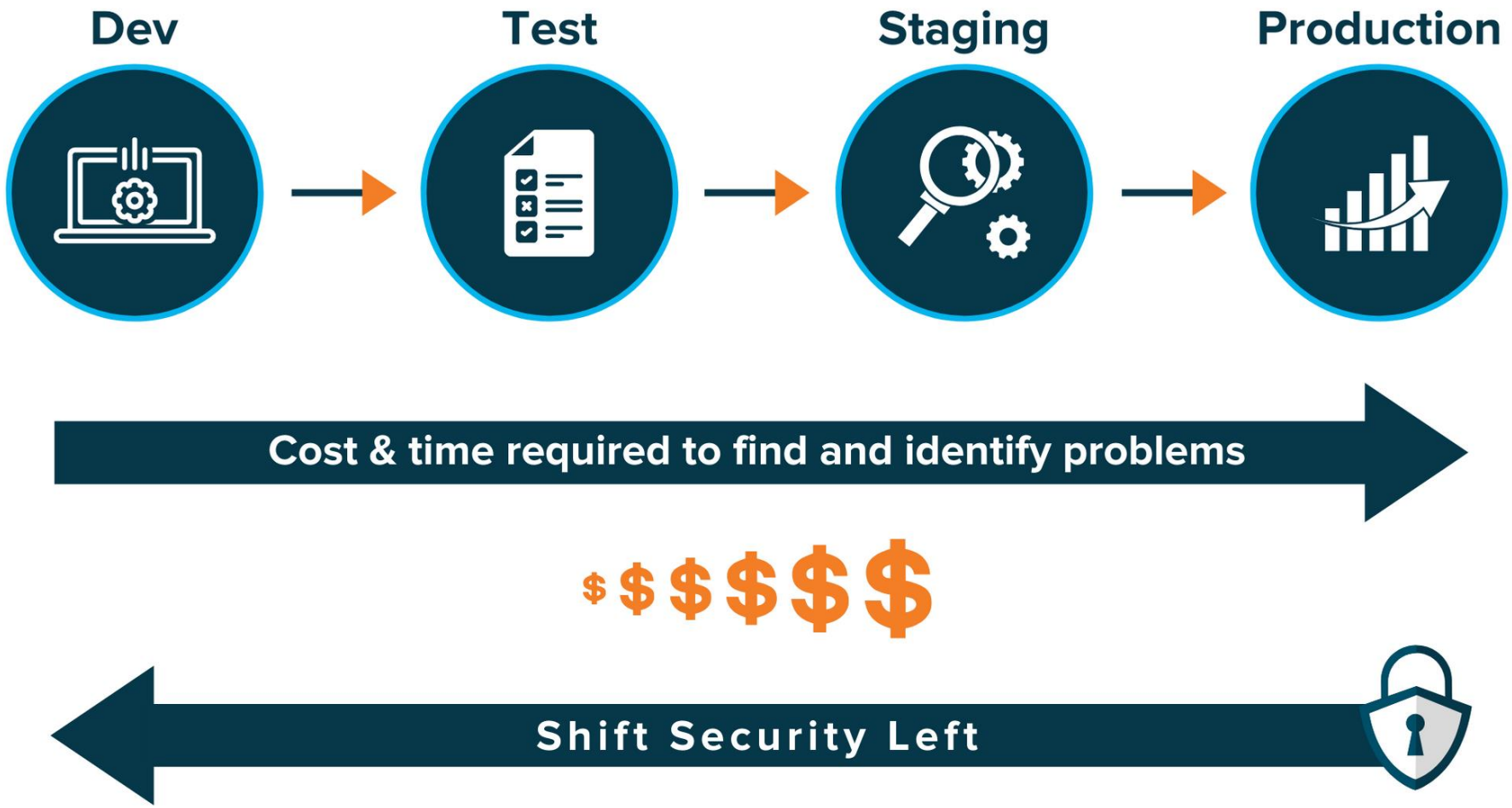
DevOps VS DevSecOps





DevOps


Azure Repos
GitHub
Bitbucket
TeamCity
Bamboo
Jenkins
Azure DevOps
Travis CI
cucumber
JMeter
SoapUI
Maven
JFrog
ARTIFACTORY
puppet
CHEF
Azure ARM Templates
Google Cloud
Azure
aws
Terraform
Amazon CloudWatch
Azure Monitor
Nagios
splunk>
DevOps Pipeline
Source Code
CI Server
Test & Scan
Artifacts
Deploy
Monitoring
DevSecOps Pipeline
sonarqube
FORTIFY
VERACODE
tenable.io
GAUNTLT
OWASP
tenable.io
JFROG
XRAY
UpGuard
Nessus
splunk>
UpGuard
catchpoint


Dev
Test
Staging
Production
Cost & time required to find and identify problems
$ $ $ $ $ $
Shift Security Left

NVD, CVE, and CVSS

- NVD == National Vulnerability Database
  - SCAP == Security Content Automation Protocol
- CVE == Common Vulnerabilities and Exposure
- CVSS == Common Vulnerability Scoring System

1. **NVD (National Vulnerability Database)**:
    1. The NVD is a repository of standards-based vulnerability management data represented using the Security Content Automation Protocol (SCAP).
    2. It provides information about vulnerabilities, including their severity, affected products, and fixes.
    3. NVD is curated by the National Institute of Standards and Technology (NIST) in the United States.
2. **CVE (Common Vulnerabilities and Exposures)**:
    1. CVE is a dictionary of common identifiers for publicly known cybersecurity vulnerabilities.
    2. Each CVE entry includes a unique identifier (CVE ID) along with a description of the vulnerability.
    3. CVE IDs are used as references in security advisories, vulnerability databases, and security tools.
    4. The CVE List is maintained by the MITRE Corporation, a not-for-profit organization that operates various research and development centers funded by the U.S. government.
3. **CPE (Common Platform Enumeration)**:
    1. CPE is a structured naming scheme for information technology systems, platforms, and packages.
    2. It provides a standardized format to describe and identify hardware, software, and other IT assets.
    3. CPE entries include vendor names, product names, version numbers, and other attributes to uniquely identify specific configurations or instances of software/hardware.
    4. CPE is used in vulnerability databases and scanning tools to associate vulnerabilities with specific IT assets.

NVD

- https://nvd.nist.gov/
- Allows searching of Vulnerability Database
  - https://nvd.nist.gov/vuln/search
- Uses SCAP
- Reports “publicly known” vulnerabilities
- Some criticism that it lags actual vulnerability discovery

CVE

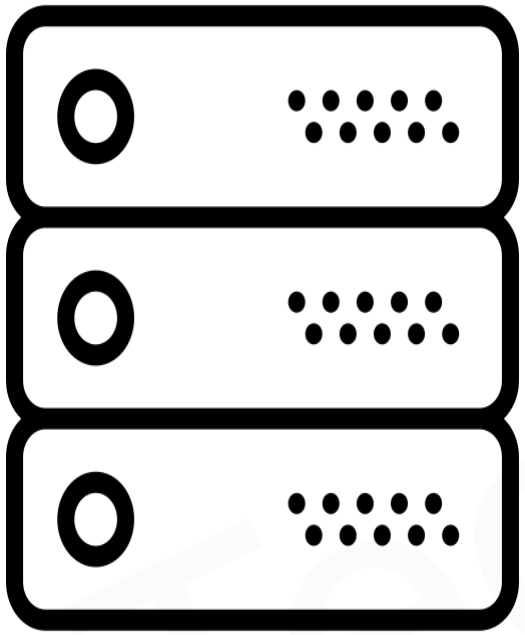
- https://cve.mitre.org/
- Is a listing for publicly known vulnerabilities

What are the Differences?

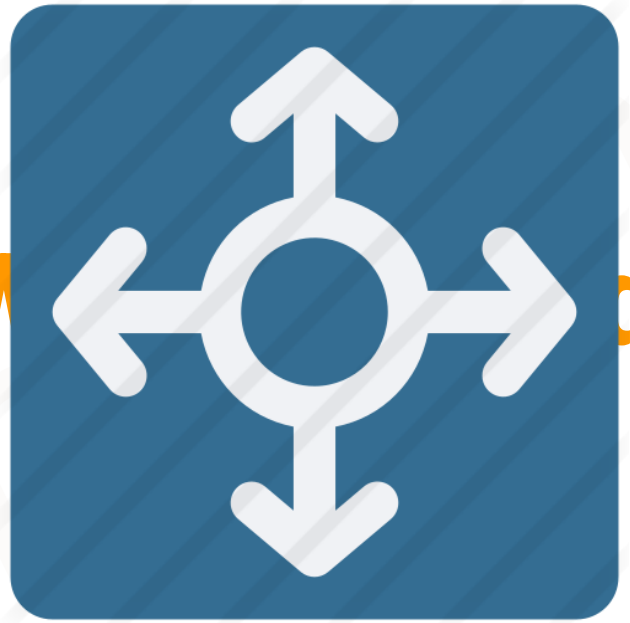
- NVD and CVE:
  - CVE is a part of NVD.
  - NVD allows more analysis and search capabilities
  - NVD and CVE are synchronized.
- CVE and CVSS
  - CVE is a free and open source standard and does not provide severity scoring as does CVSS

Cloud Fundamentals

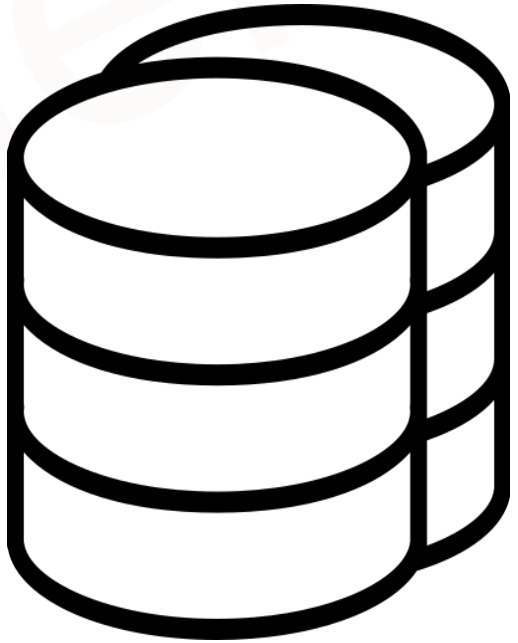
Traditional Datacenters



Servers

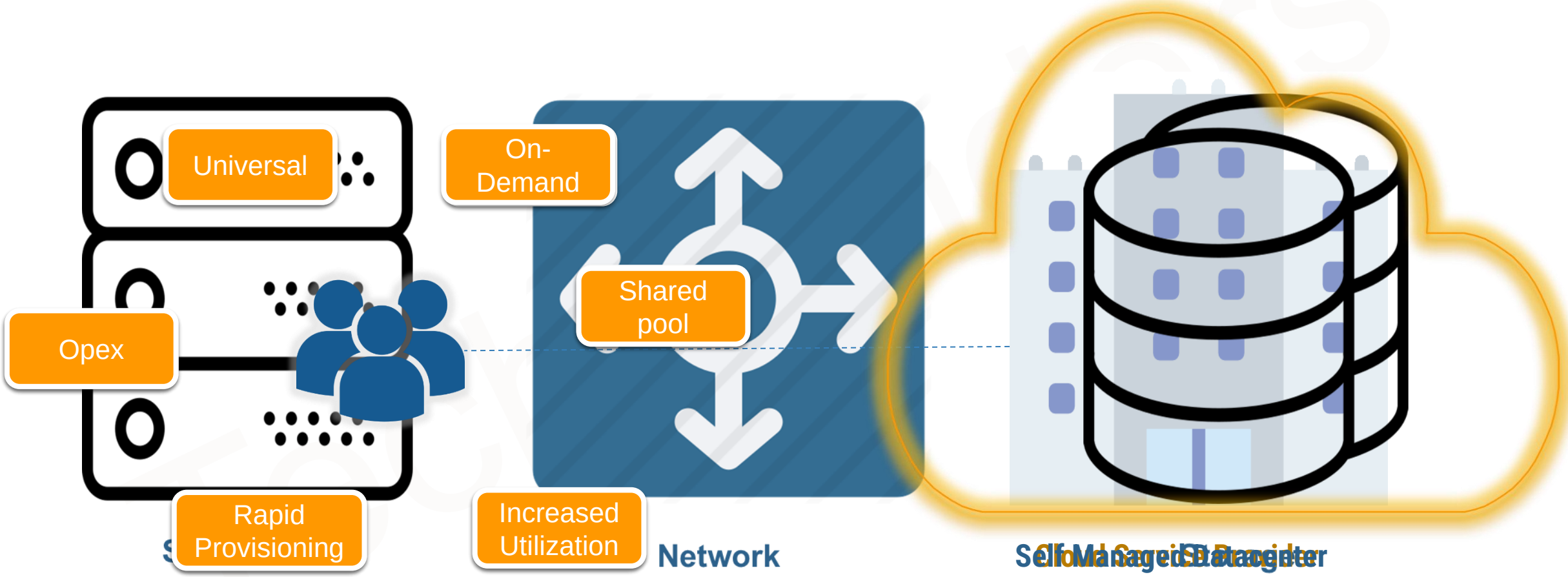


Network

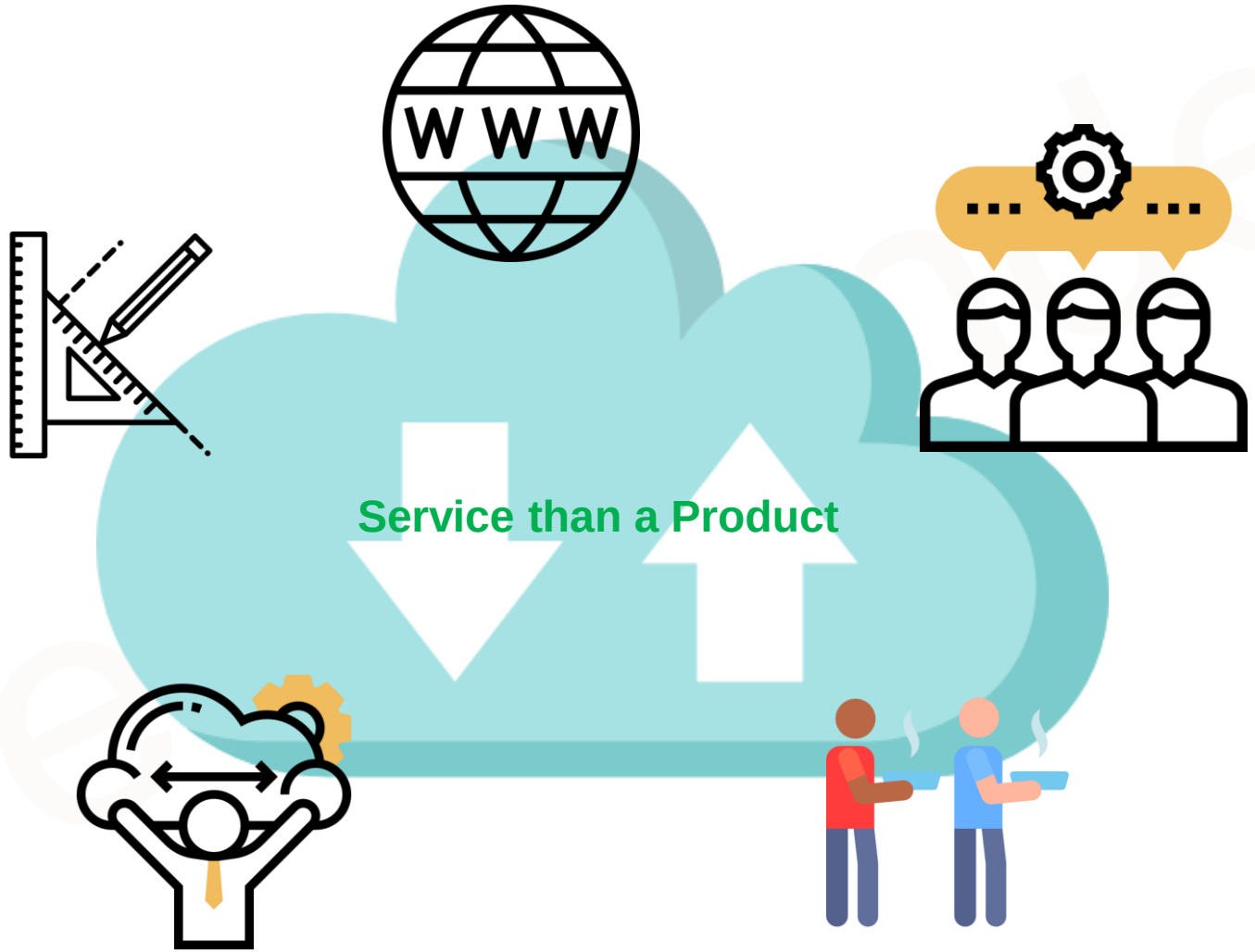


Storage

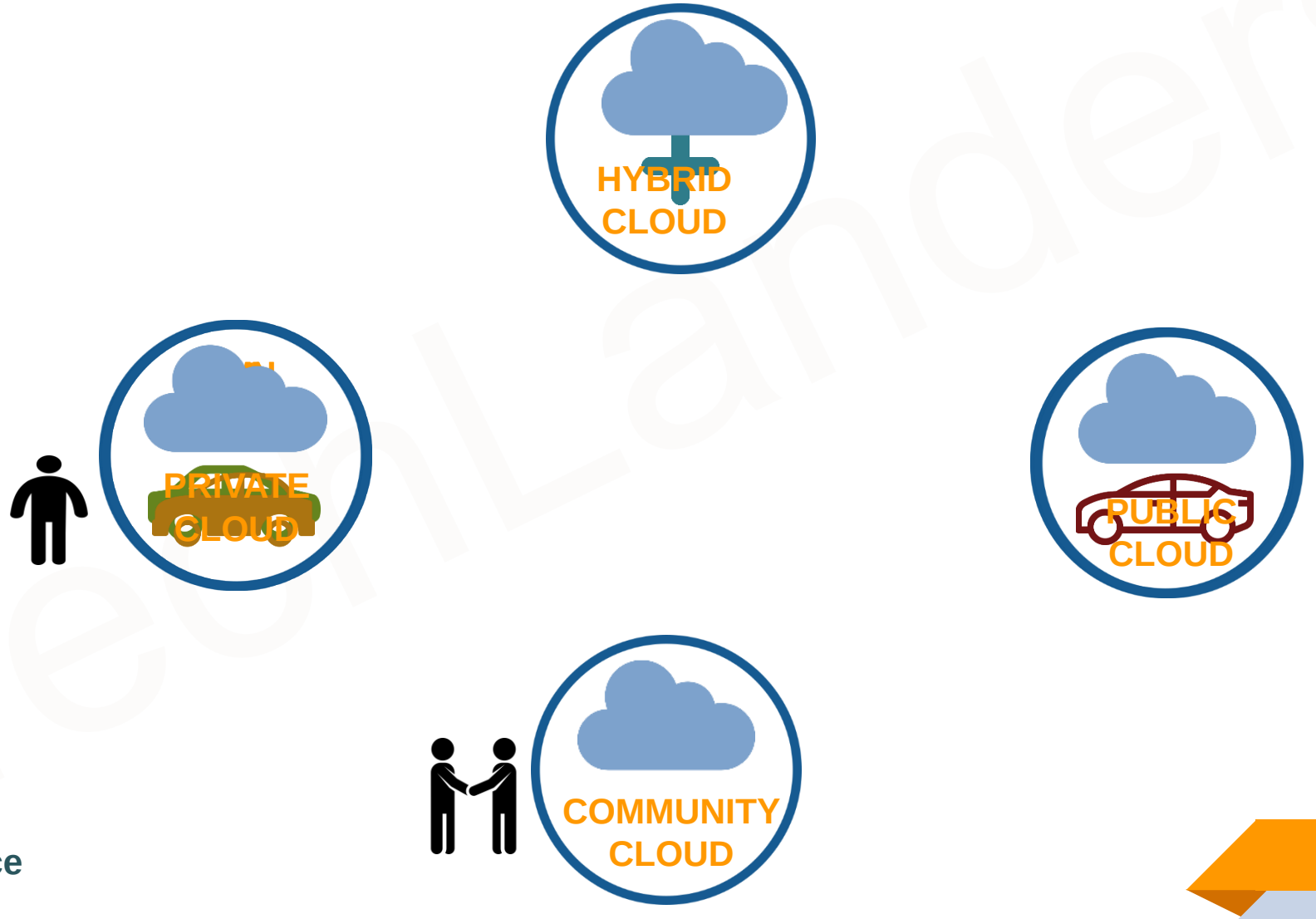
Traditional vs Cloud Datacenters



Characteristics of Cloud

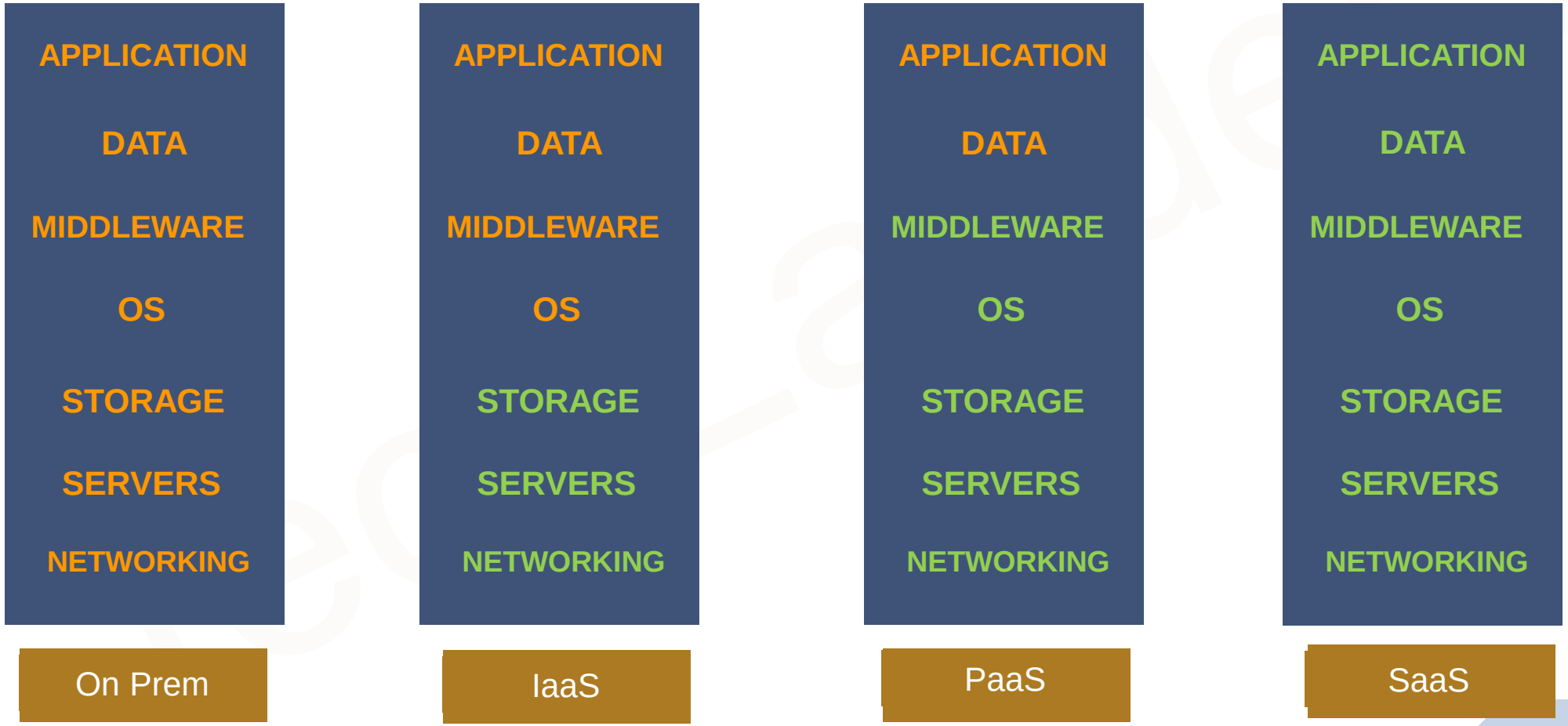


Cloud Deployment Types



Service Models - CARaaS

Service Models



Cloud Benefits

Cloud Major Use Cases

On and Off

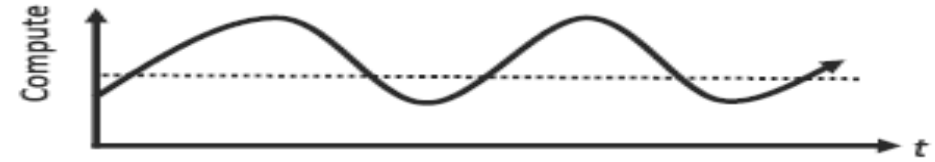
On and off workloads (e.g. batch job)
Over provisioned capacity is wasted
Time to market can be cumbersome

Growing Fast

Successful services needs to grow/scale
Keeping up with growth is a big IT challenge
Cannot provision hardware fast enough

Unpredictable Bursting

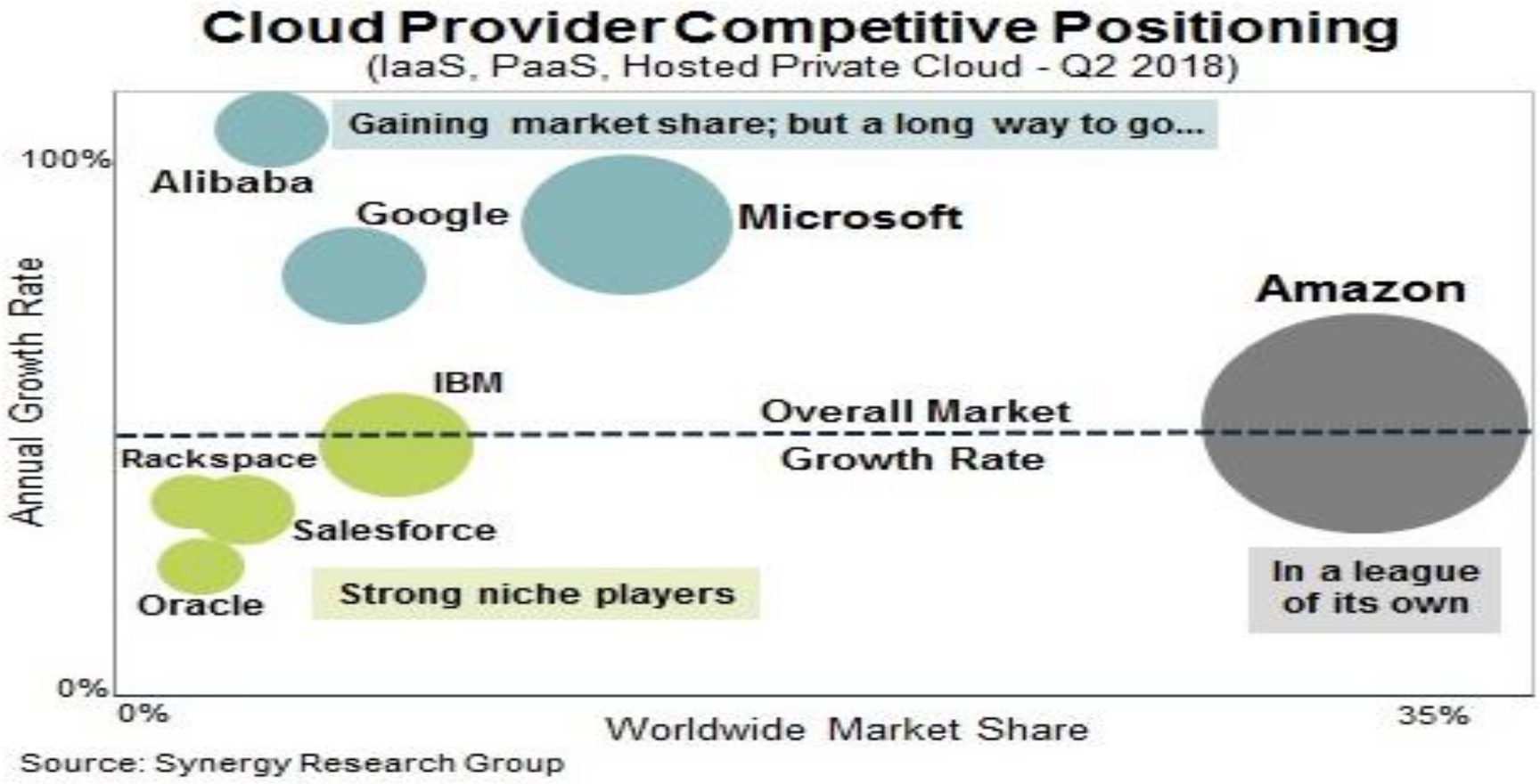
Unexpected/unplanned peak in demand
Sudden spike impacts performance
Cannot over provision for extreme cases



Predictable Bursting

Services with micro seasonality trends
Peaks due to periodic increased demand
IT complexity and wasted capacity

Cloud Players



AWS

Amazon Web Services

AWS (Amazon Web Services) is a group of web services (also known as cloud services) being provided by Amazon since 2006.

AWS provides huge list of services starting from basic IT infrastructure like CPU, Storage as a service, to advance services like Database as a service, Serverless applications, IOT, Machine Learning services etc..

Hundreds of instances can be build and use in few minutes as and when required, which saves ample amount of hardware cost for any organizations and make them efficient to focus on their core business areas.

Currently AWS is present and providing cloud services in more than 190 countries.

Well-known for IaaS, but now growing fast in PaaS and SaaS.

Why AWS?

Low Cost: AWS offers, pay as you go pricing. AWS models are usually cheapest among other service providers in the market.

Instant Elasticity: You need 1 server or 1000's of servers, AWS has a massive infrastructure at backend to serve almost any kind of infrastructure demands, with pay for what you use policy.

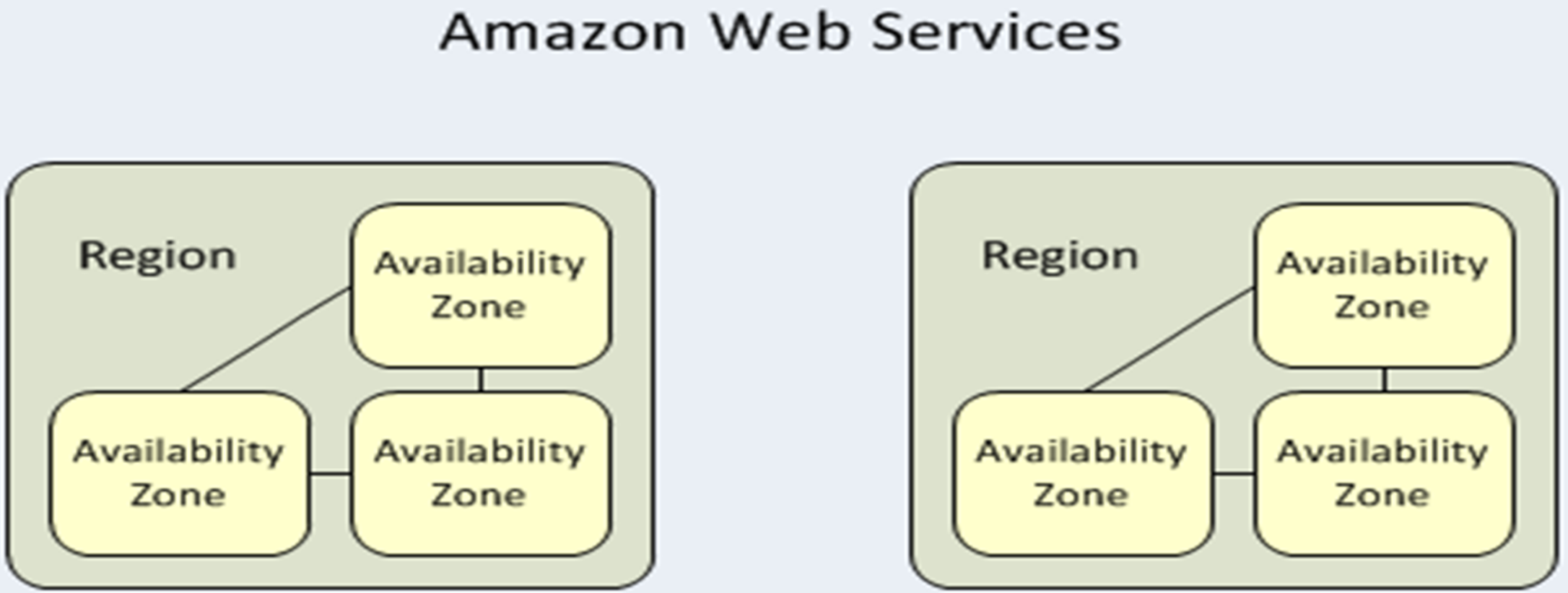
Scalability: Facing some resource issues, no problem within seconds you can scale up the resources and improve your application performance. This cannot be compared with traditional IT datacenters.

Multiple OS's: Choice and use any supported Operating systems.

Multiple Storage Options: Choice of high I/O storage, low cost storage. All is available in AWS, use and pay what you want to use with almost any scalability.

Secure: AWS is PCI DSS Level1, ISO 27001, FISMA Moderate, HIPAA, SAS 70 Type II passed. In-fact systems based on AWS are usually more secure than in-house IT infrastructure systems.

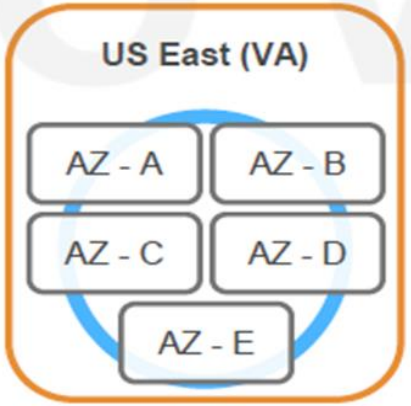
Amazon Web Services

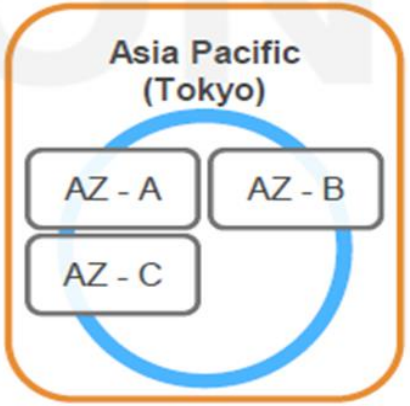


Amazon Web Services

At least 2 AZs per region.

- Examples:
  - US East (N. Virginia)
    - us-east-1a
    - us-east-1b
    - us-east-1c
    - us-east-1d
    - us-east-1e
  - Asia Pacific (Tokyo)
    - ap-northeast-1a
    - ap-northeast-1b
    - ap-northeast-1c





Note: Conceptual drawing only. The number of Availability Zones (AZ) may vary.

Amazon Web Services

AWS Regions:
-Geographic Locations
-Consists of at least two Availability Zones(AZs)
-All of the regions are completely independent of each other with separate Power Sources, Cooling and Internet connectivity.

AWS Availability Zones
-AZ is a distinct location within a region
-Each Availability Zone is isolated, but the Availability Zones in a Region are connected through low-latency links.
-Each Region has minimum two AZ’s
-Most of the services/resources are replicated across AZs for HA/DR purpose.

Amazon Web Services

Current:

22 AWS Regions

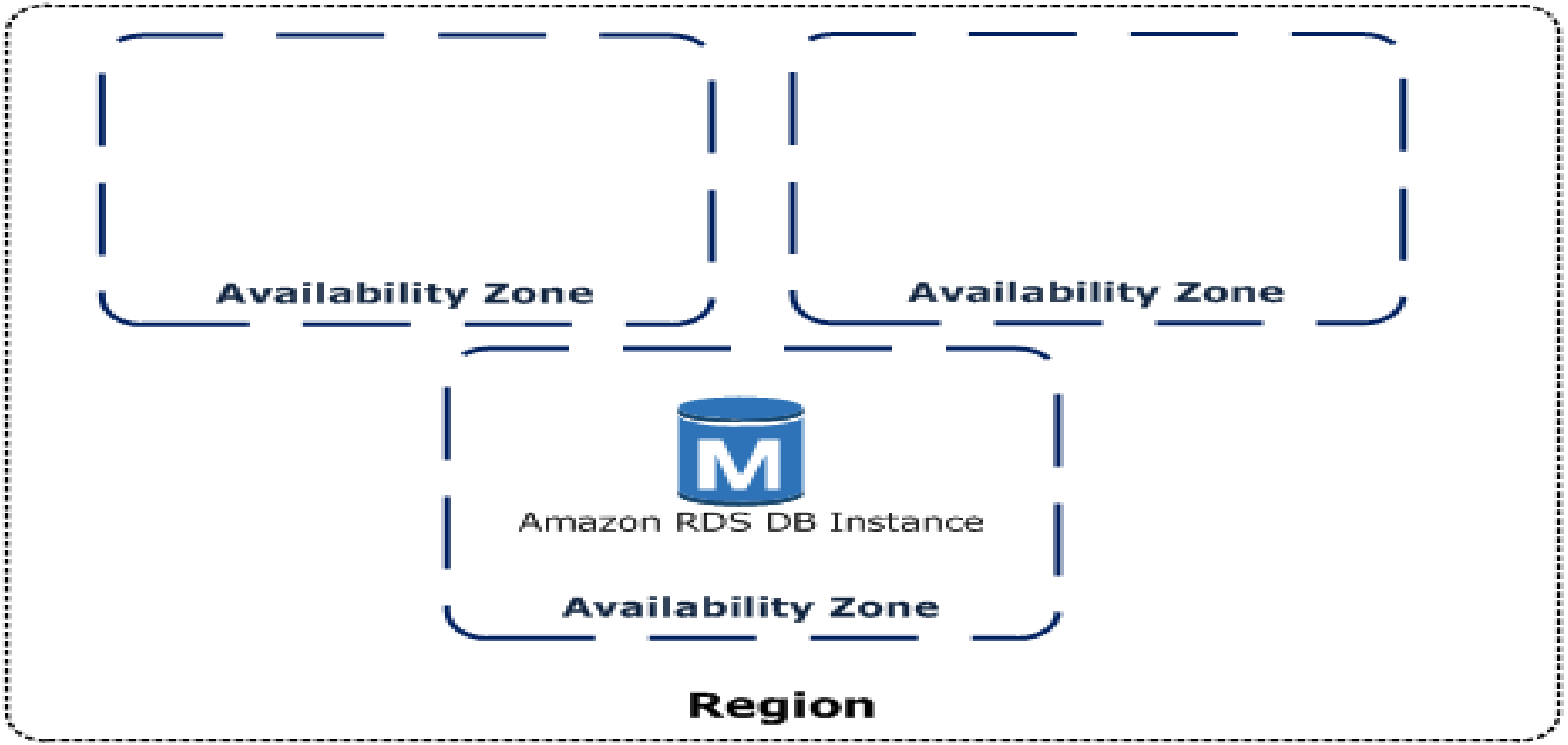
69 AZs

Upcoming:

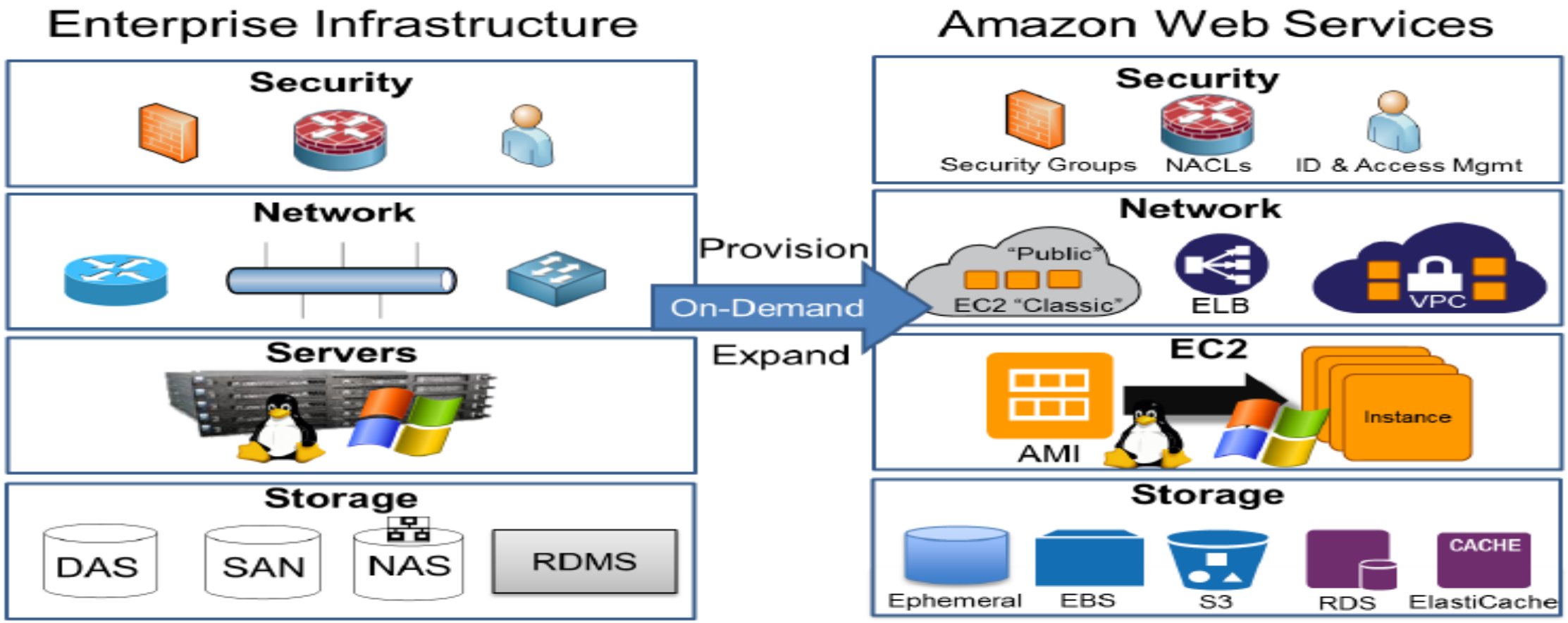
4 Regions

13 AZs

Amazon Web Services



AWS


Enterprise Infrastructure
Security
Network
Servers
Storage
DAS
SAN
NAS
RDMS
Provision
On-Demand
Expand
Amazon Web Services
Security
Security Groups
NACLs
ID & Access Mgmt
Network
"Public"
EC2 "Classic"
ELB
VPC
EC2
AMI
Instance
Storage
Ephemeral
EBS
S3
RDS
CACHE
ElastiCache

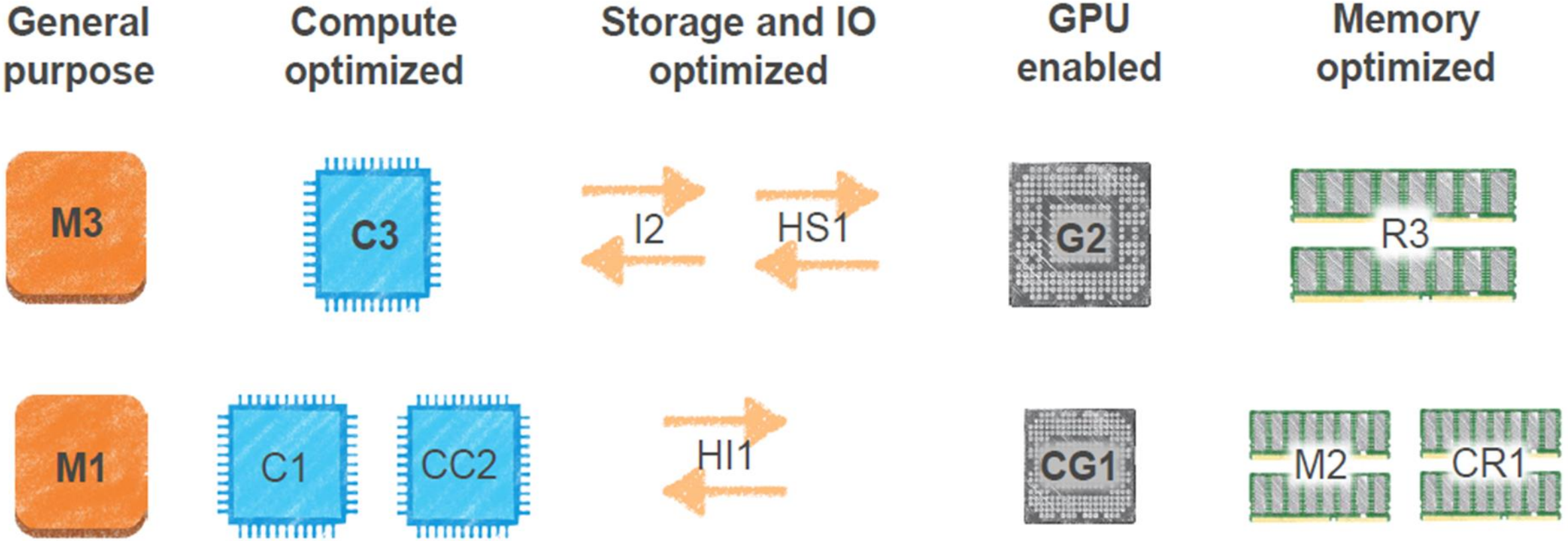
AWS
Compute Services

AWS Elastic Compute Cloud

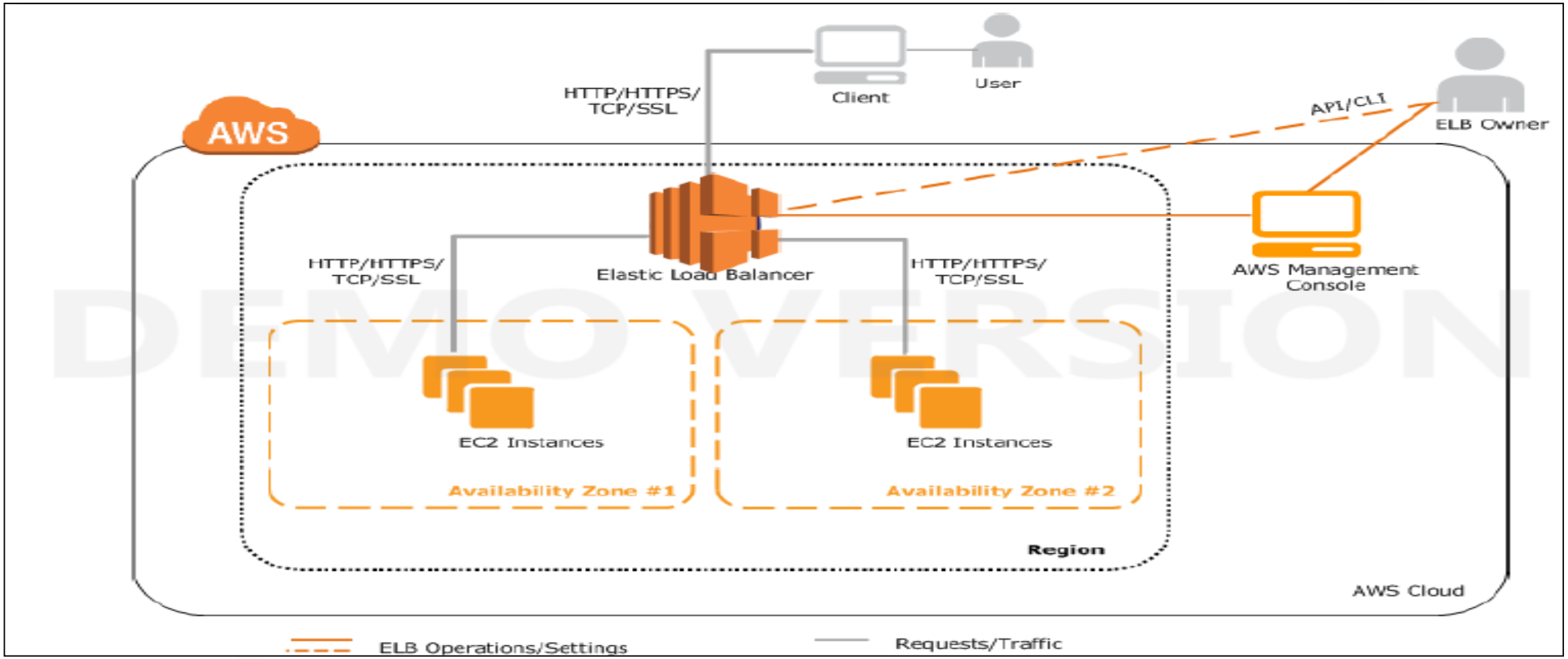
- Amazon EC2 stands for Elastic Compute Cloud, and is the Primary AWS web service.
- Provides Resizable compute capacity
- Reduces the time required to obtain and boot new server instances to minutes
- There are two key concepts to Launch instances in AWS:
  - Instance Type
  - AMI

- EC2 Facts:
  - Scale capacity as your computing requirements change
  - Pay only for capacity that you actually use
  - Choose Linux or Windows OS as per need. You have to Manage the OS and Security of same.
  - Deploy across AWS Regions and Availability Zones for reliability/HA

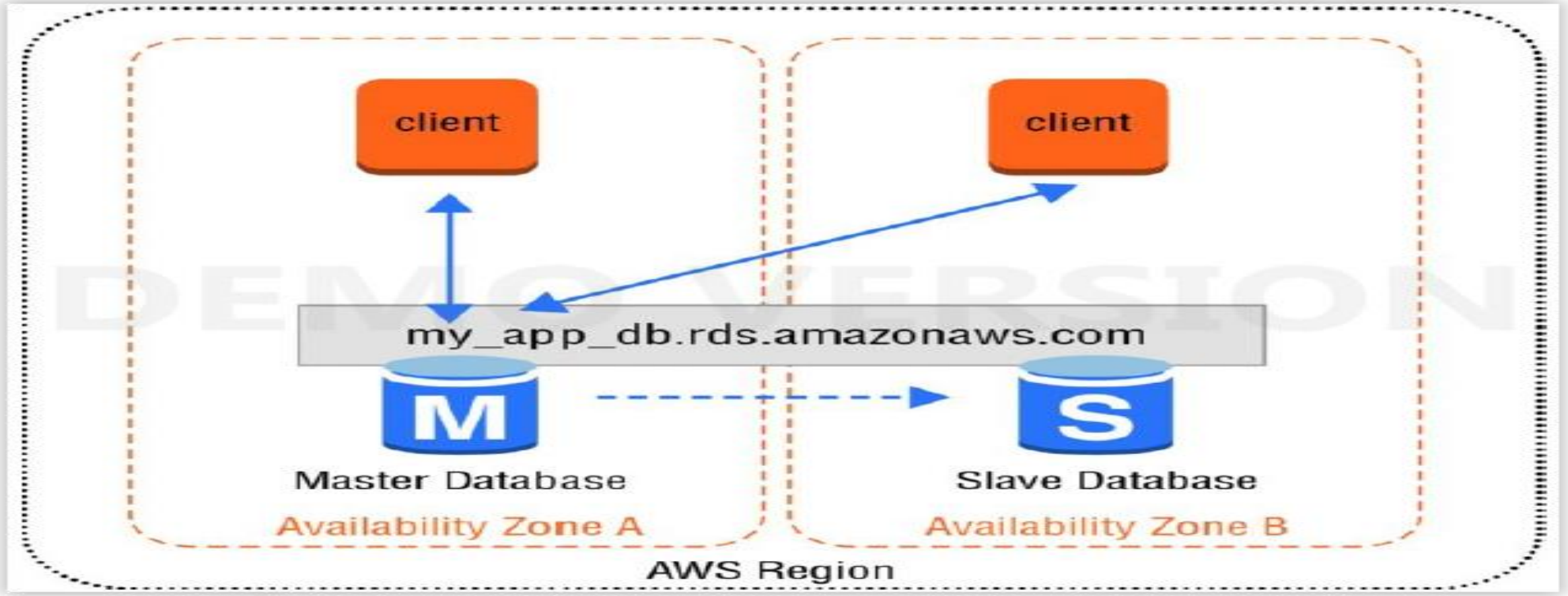
AWS EC2


General purpose
Compute optimized
Storage and IO optimized
GPU enabled
Memory optimized
M3
C3
I2
HS1
G2
R3
M1
C1
CC2
HI1
CG1
M2
CR1

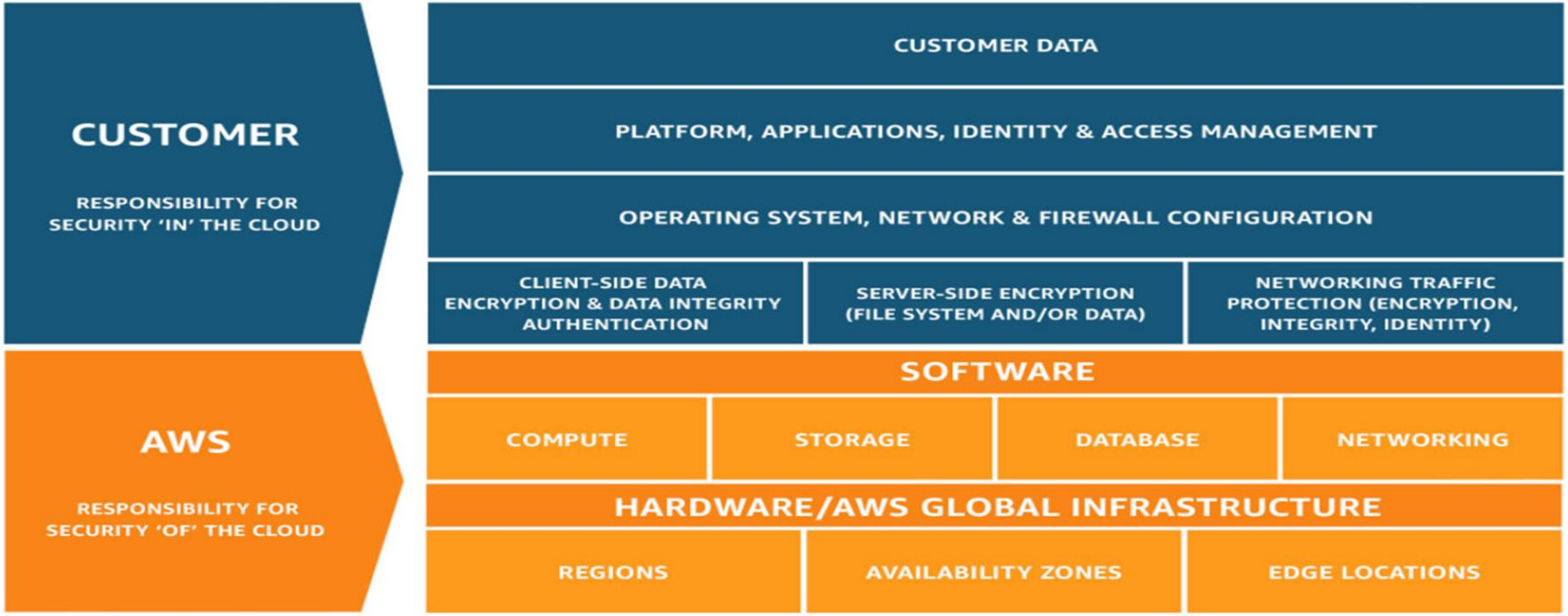
ELB



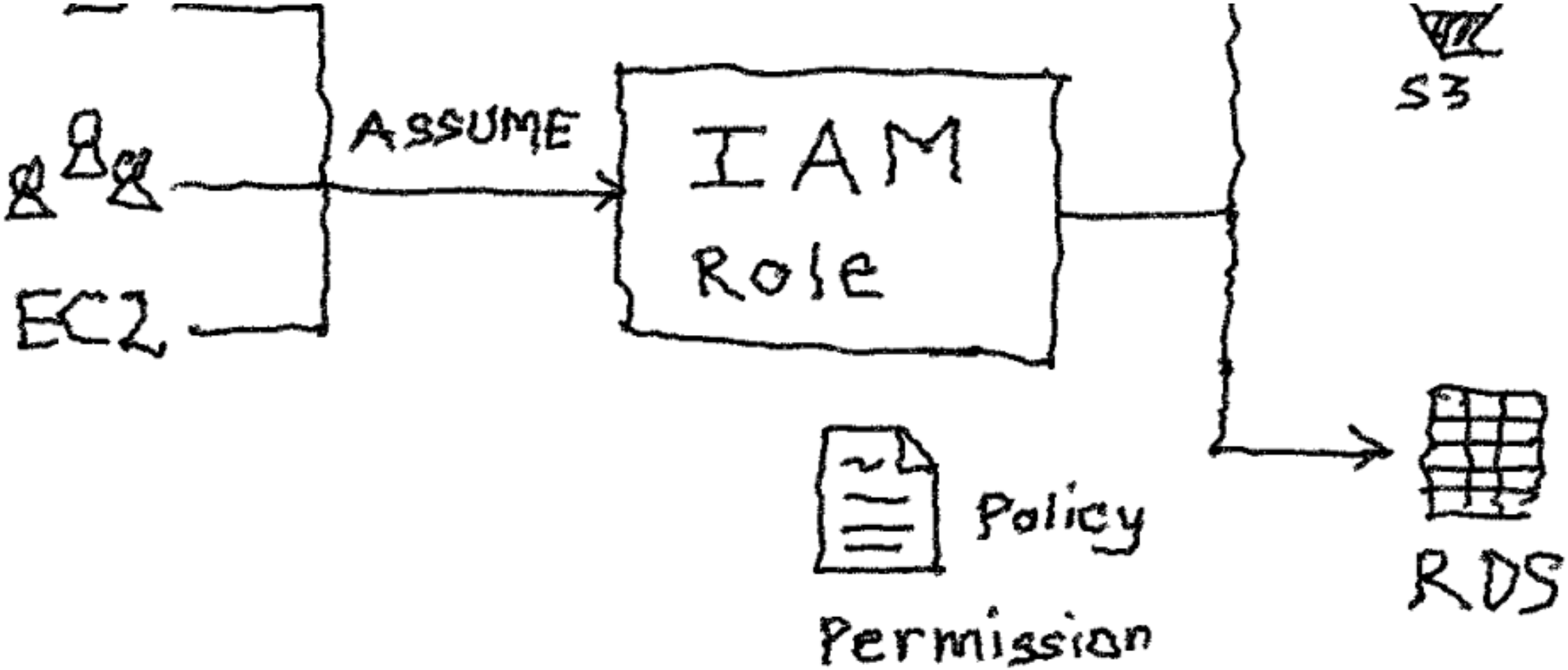
PaaS Example



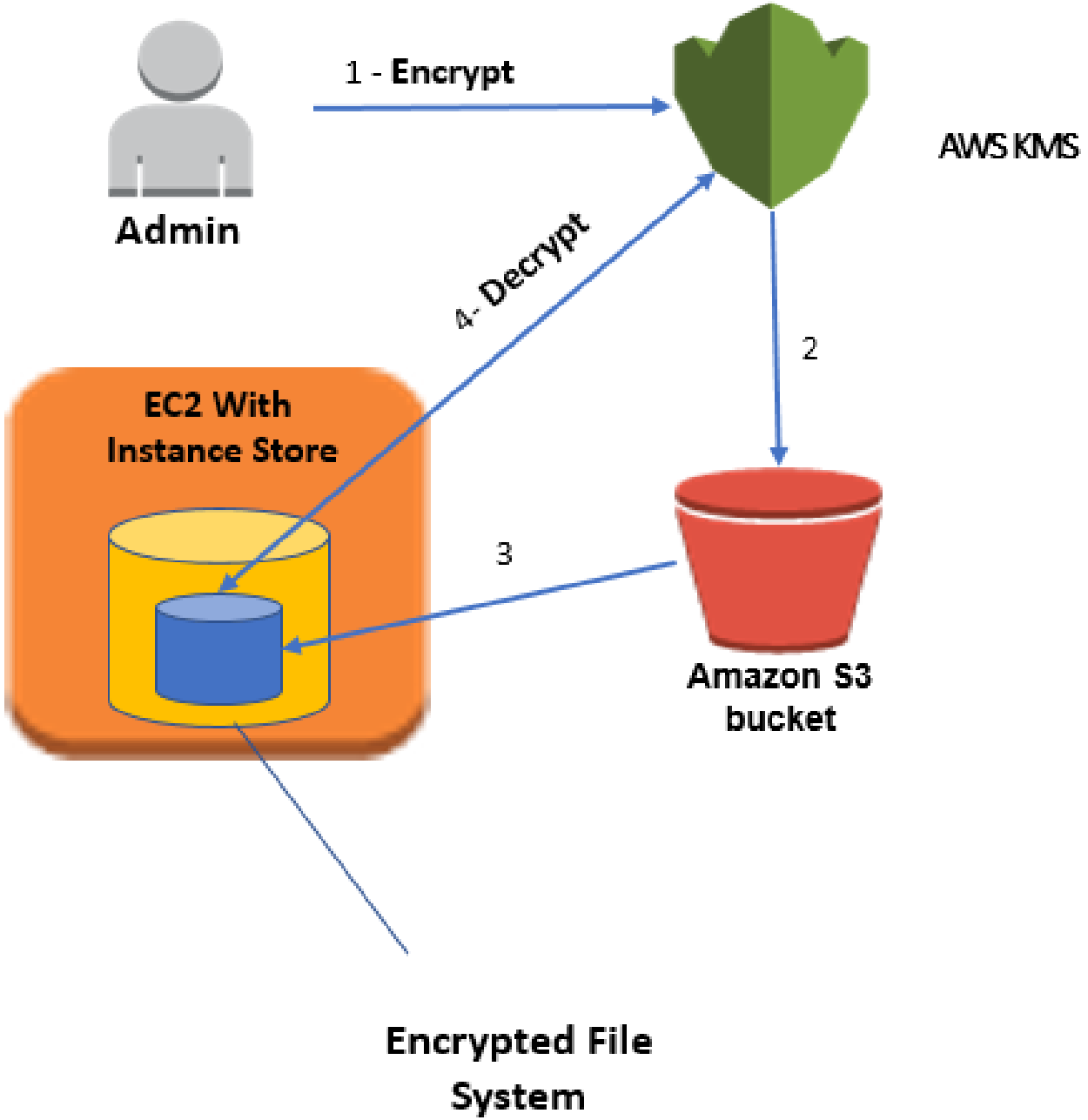
Shared Responsibility



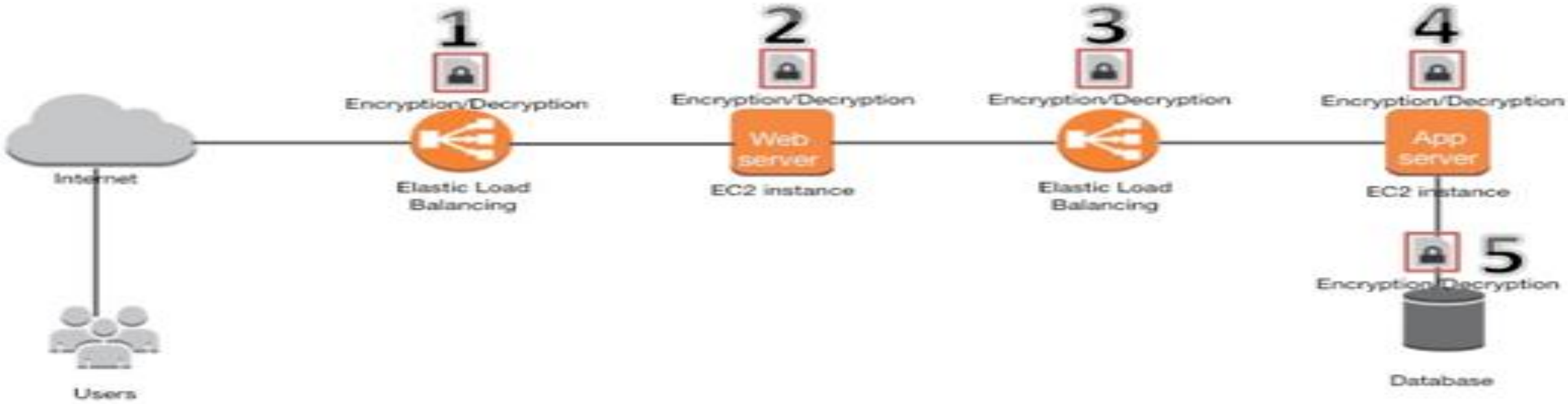
> *LAB : Creation of an ubuntu server and demonstation of security at instance level.*


ASSUME
IAM
Role
EC2
S3
RDS
Policy
Permission

LAB : Handling security credentials and IAM roles for security purpose.


1 - Encrypt
AWS KMS
Admin
4- Decrypt
2
EC2 With
Instance Store
3
Amazon S3
bucket
Encrypted File
System

End-to-end encryption in a standard web application



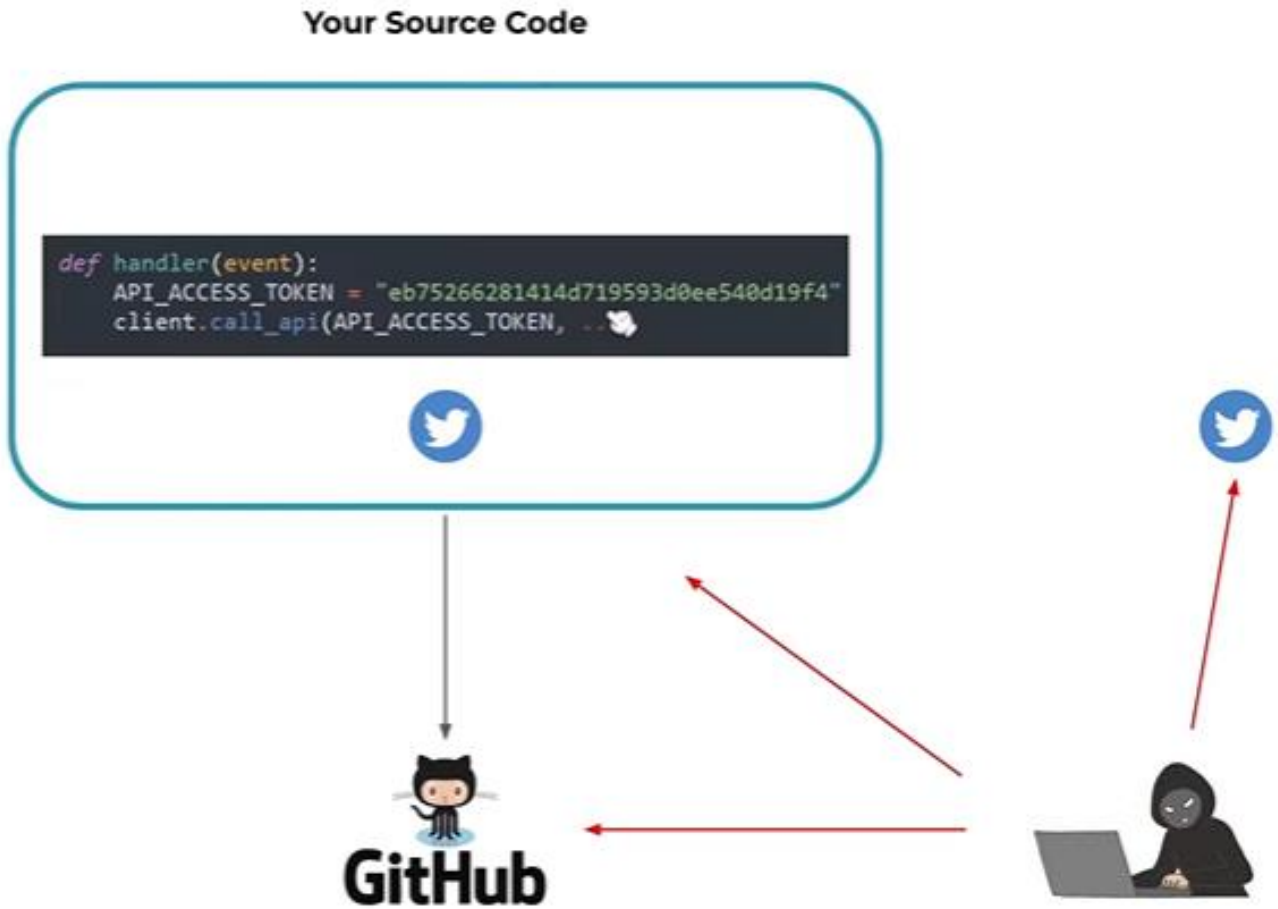
1. **Security Groups and NACLs**: Demonstrate how to use security groups and network access control lists (NACLs) to control inbound and outbound traffic to EC2 instances. Implement security best practices such as least privilege access.
2. **Identity and Access Management (IAM)**: Set up IAM roles and policies to control access to EC2 instances and other AWS resources. Use IAM roles for EC2 instances to grant permissions to interact with AWS services securely.
3. **Encryption**: Implement encryption for data at rest and data in transit. Use AWS Key Management Service (KMS) to manage encryption keys and encrypt EBS volumes or S3 buckets used by EC2 instances.
4. **Monitoring and Logging**: Configure CloudWatch alarms to monitor EC2 instance metrics such as CPU utilization, network traffic, and disk usage. Set up CloudWatch Logs to centralize logs from EC2 instances for troubleshooting and analysis.

4

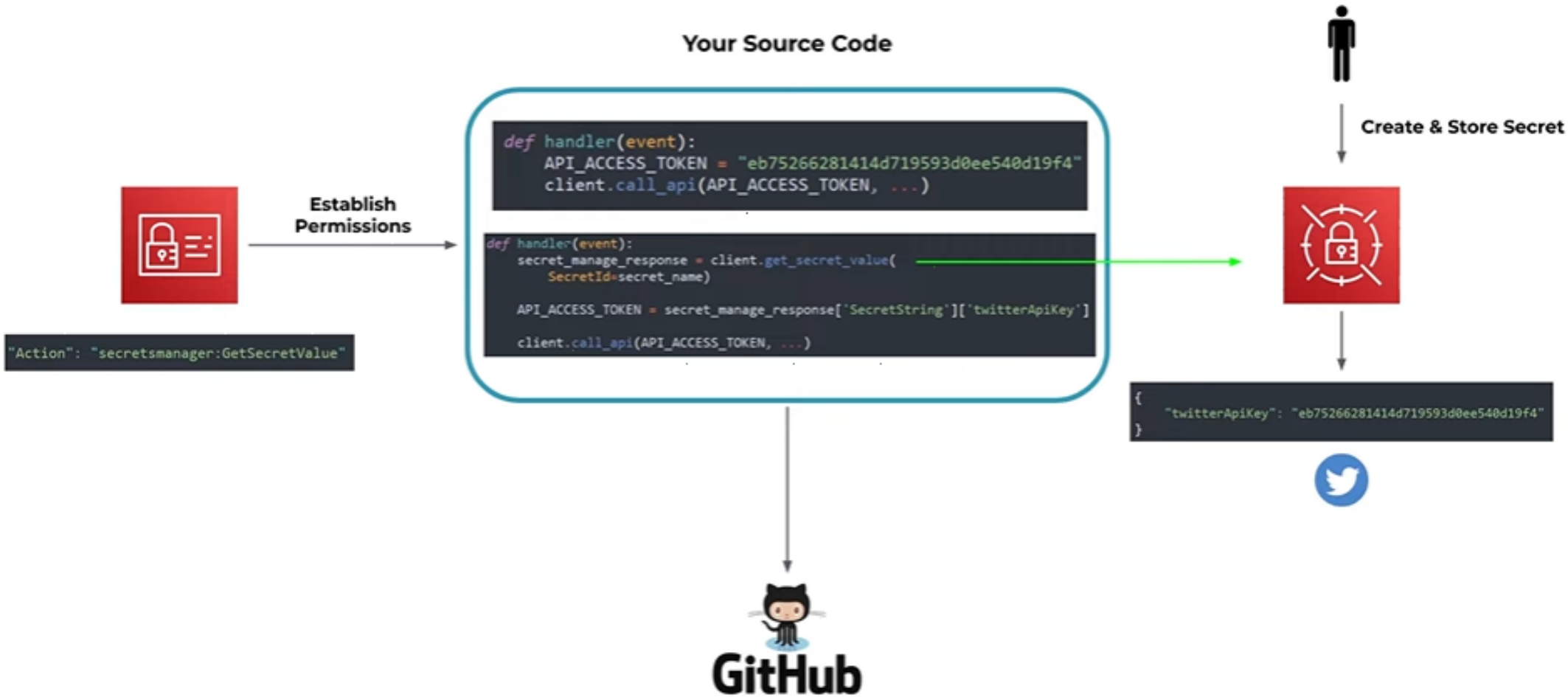
What is Secrets Manager?

- Managed AWS Service that **protects** api keys, tokens, database creds, ...
- Centralizes secret storage and **encrypts** your data
- Access is managed through IAM and monitored via Cloudwatch
- Allows you to control the **lifecycle** of secrets including **rotation**

An Example


Your Source Code
def handler(event):
API_ACCESS_TOKEN = "eb75266281414d719593d0ee540d19f4"
client.call_api(API_ACCESS_TOKEN, ...
GitHub

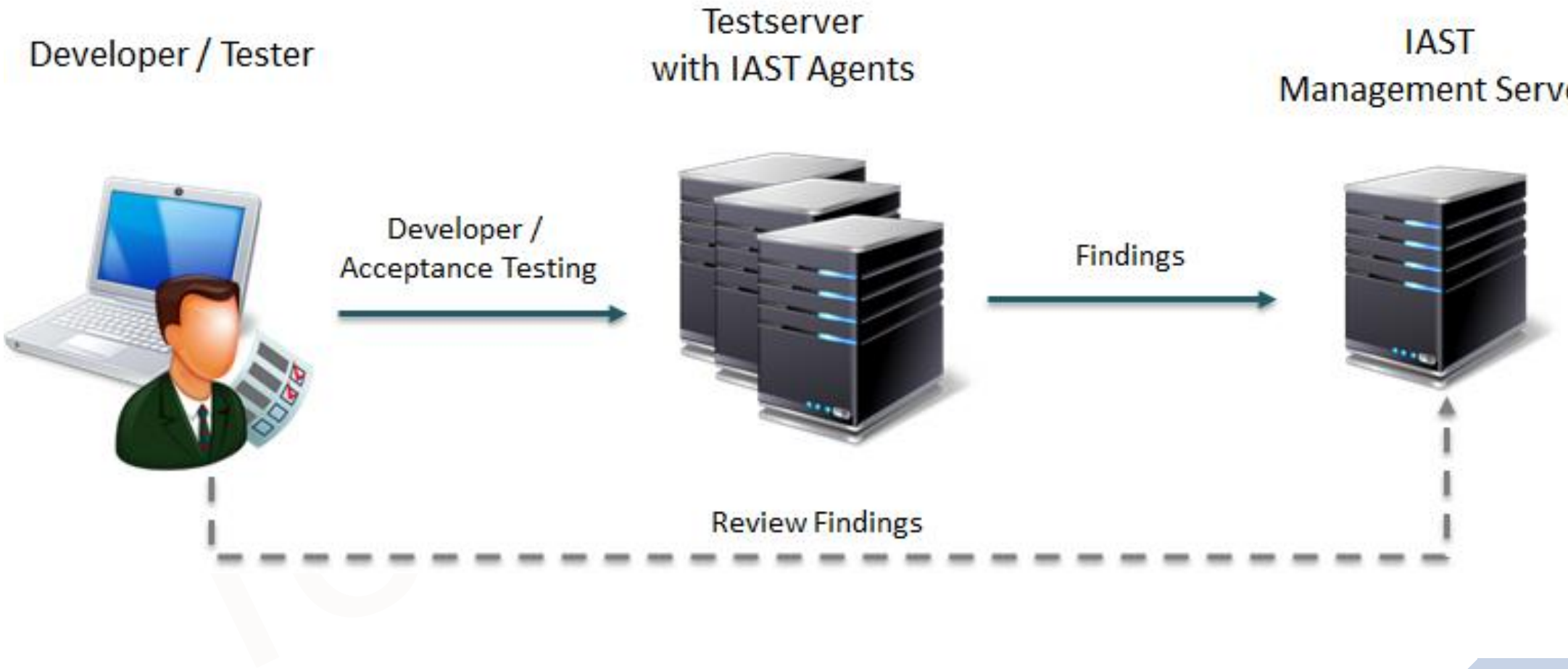
How Secrets Manager Helps



SAST vs. DAST

Static application security testing (SAST) and dynamic application security testing (DAST) are both methods of testing for security vulnerabilities, but they're used very differently.

SAST	DAST
White box security testing The tester has access to the underlying framework, design, and implementation. The application is tested from the inside out. This type of testing represents the developer approach.	**Black box security testing**The tester has no knowledge of the technologies or frameworks that the application is built on. The application is tested from the outside in. This type of testing represents the hacker approach.
Requires source codeSAST doesn't require a deployed application. It analyzes the sources code or binary without executing the application.	**Requires a running application**DAST doesn't require source code or binaries. It analyzes by executing the application.
Finds vulnerabilities earlier in the SDLC The scan can be executed as soon as code is deemed feature-complete.	**Finds vulnerabilities toward the end of the SDLC**Vulnerabilities can be discovered after the development cycle is complete.
Less expensive to fix vulnerabilities Since vulnerabilities are found earlier in the SDLC, it's easier and faster to remediate them. Findings can often be fixed before the code enters the QA cycle.	**More expensive to fix vulnerabilities**Since vulnerabilities are found toward the end of the SDLC, remediation often gets pushed into the next cycle. Critical vulnerabilities may be fixed as an emergency release.
Can't discover run-time and environment-related issues Since the tool scans static code, it can't discover run-time vulnerabilities.	**Can discover run-time and environment-related issues**Since the tool uses dynamic analysis on an application, it is able to find run-time vulnerabilities.
Typically supports all kinds of software Examples include web applications, web services, and thick clients.	**Typically scans only apps like web applications and web services**DAST is not useful for other types of software.


Developer / Tester
Testserver
with IAST Agents
IAST
Management Server
Developer /
Acceptance Testing
Findings
Review Findings

SAST	DAST	IAST
White box testing	Black box testing	Grey box testing
Does Source Code Analysis	Finds runtime vulnerability	Does both
Can be integrated into SDLC	Not integrated into SDLC	Can be integrated into SDLC
Used in the early stage of SDLC	Used at the later stage of SDLC	Used at both the stages of SDLC
Less expensive to fix the found vulnerability	Expensive to fix the found vulnerability	Depends on the vulnerability
Has false positives	Low false positives	Low false positives

Monitoring Fundamentals

- Monitoring refers to the process of observing and tracking various parameters, activities, or systems to ensure they are performing as expected. In various contexts, monitoring plays a crucial role in maintaining efficiency, security, and reliability.

1. **Early Detection of Issues**
2. **Optimization and Performance Improvement**
3. **Resource Utilization**
4. **Security and Compliance**
5. **Capacity Planning**
6. **Enhanced Reliability and Availability**