CU - Cloud Test Practitioner (CU CTP) Syllabus

Version 1.0 2022

Cloud United

Copyright Notice

Thank you to the main authors:

- Vipul Kocher, Smitha Menon and Gaurav Pandey

Thank you to the review committee

Alexander Allan, Alexis Jesús Herrera Colmenares, Alfonso Fernández, Ángel Rayo Acevedo, Arjan Brands, Armando Dörsek, Chantelle Jones, Christine Green, Daniel Van der Zwan, Fabiola Mero, Guino Henostroza, Héctor Ruvalcaba, Helge Sune Nymand, Ilhan Koral, Ismael Betancourt, Isaac Malamud, Julie Gardiner, Julio Córdoba Retana, Kyle Alexander Siemens, Leandro Melendez, Márcia Campos, Márcia Araújo Coelho, Miaomiao Tang, Miguel Angel De León Trejo, Orlando Torres, Richard Seidl, Roland Møller, Sammy Kolluru, Samuel Ouko, Thomas Cagley, Vanessa Islas Padilla, Wilson Gumba, Wim Decoutere

Revision History

Version	Date	Remarks
CU 0.92 2022	August 8th, 2022	For the CU Review Committee
CU 1.0 2022	September 20, 2022	First Public Release

Table of Contents

Purpose of this document

This syllabus forms the basis of the Cloud United - Cloud Test Practitioner (CTP) certification. This document defines what you need to know in order to pass the certification exam for CU-CTP and is copyright of Cloud United. The certification exam will only cover concepts and knowledge that are described in this document, although this document contains practical elements that will not be covered by the certification exam but is required to be covered in the training.

Resources of the CU

An overview of CU resources as well as all relevant information about the CU certification and other types of CU certifications are available on www.Cloud-united.com, the official website of Cloud United. The information to be found on www.cloud-united.org includes:

- A complete list of recognized CU training providers and available courses. Note that training is recommended but not required in order to take the CU-CTP certification exam.
- CU CTP Syllabus (this document) for download.
- A sample exam set of 10 CU-CTP questions with answers, for training purposes.
- We aim to have the documents available in further languages as soon as possible. For currently available language versions, please check www.cloud-united.org.

About CU Cloud Test Practitioner (CTP)

The cloud has changed the way applications are hosted and tested. The Cloud United - Cloud Test Practitioner (CTP) certification course focuses on the essential cloud characteristics and cloud-specific tests that a tester needs to know.

The certification itself is independent of any specific cloud provider; however, hands-on exercises will be performed on several of the most widely known providers as examples. Earners of the Cloud Test Practitioner certification will be able to perform various types of cloud specific tests and understand the differences between testing the cloud and testing applications in the cloud. The course does not teach basics of general testing as it is expected that the participants have this knowledge prior to joining this course. For the basics of testing, we recommend that participants hold the ISTQB Certified Tester

Foundation Level certificate (or similar) or have at least read through the syllabus and understand the content.

Business Outcomes (BO)

BO-1	Support the creation of a test strategy from the perspective of the applications based on the cloud characteristics and various test types such as compliance, security, scalability, performance, resilience, disaster recovery and monitoring
BO-2	Perform calculations for estimating cost of setting up cloud environments
BO-3	Support the creation of test environment by setting up and running docker images, clustering and using Kubernetes
BO-4	Create cloud instances of various operating systems (OS) and configurations as a prerequisite to setting up the required test environments.
BO-5	Setup load balancing and autoscaling for testing reliability, scalability, resilience etc.
BO-6	Perform data migration testing
BO-7	Perform tests related to multi-tenancy testing
BO-8	Perform basic security tests for cloud-hosted applications
BO-9	Perform basic load and elasticity testing for cloud-hosted applications

Learning Objectives/Cognitive Levels of Knowledge

Learning objectives (LOs) are brief statements that describe what you are expected to know after studying each chapter. The LOs are defined based on Bloom's modified taxonomy as follows:

- K1: Remember. Some of the action verbs are Remember, Recall, Choose, Define, Find, Match, Relate, Select
- K2: Understand. Some of the action verbs are Summarize, Generalize, Classify, Compare, Contrast, Demonstrate, Interpret, Rephrase
- K3: Apply. Some of the action verbs are Implement, Execute, Use, Apply

For more details of Bloom's taxonomy, please refer to [BT1] and [BT2] in References.

Hands-on Objectives

Hands-on Objectives (HOs) are brief statements that describe what you are expected to perform or execute to understand the practical aspect of Learning.

The HOs are defined as follows:

- H0: Live demo of an exercise or recorded video
- H1: Guided exercise. The trainees follow the sequence of steps performed by the trainer
- H2: Exercise with hints. Exercise to be solved by the trainee utilizing hints provided by the trainer
- H3: Unguided exercises without hints

General Prerequisites

Mandatory

- None

Recommended

- ISTQB® Certified Tester Foundation Level (CTFL) or equivalent
- Some software development or testing experience

Programming Language Prerequisites

Recommended

- Basic knowledge of any programming language.

Specific Tools Mentioned in the Syllabus [Disclaimer]

The tools mentioned in the syllabus are used solely as examples. These tools represent some of the most commonly used ones at the time of releasing this syllabus.
This syllabus' focus is on concepts. Therefore, specific tools are used only as the means to demonstrate those concepts, or to perform hands-on exercises.
The syllabus does not aim to promote any tool over any other or support any company that produces tools over any other. During training or otherwise, if other suitable alternatives are available, anyone is welcome to use them as well.

Chapter 1 - Introduction to Cloud (90 Minutes)

Keywords: Cloud, Cloud Computing, Resource Pooling, Private Cloud, Public Cloud, Community Cloud, Hybrid Cloud, Reduced up-front investment (CAPEX), Total Cost of Ownership (TCO), Total Operational Cost (TOC), Anything-as-a-Service (XaaS), Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), Infrastructure-as-a-Service (IaaS), Infrastructure as Code (IaC), Platform as Code (PaC), Compliance as Code (CaC)

LO #	Description
LO-1.1.1	Recall the key characteristics, advantages and disadvantages of a cloud computing system including the main deployment and service models. (K1)
LO-1.2.1	List the main advantages and disadvantages of cloud computing. (K1)
LO-1.3.1	Outline SaaS, PaaS, IaaS and other types of services such as database~, storage~, DR~, Function~, as-a-service offered by cloud platforms. (K2)
LO-1.4.1	Recall how managing various activities such as infrastructure, platform or compliance is done through code instead of doing manual tasks (K1)
LO-1.5.1	Comprehend the pay-per-use model of cloud service providers (K2)

HO #	Description
HO-1.5.1	Estimate the cost of running a system on a cloud platform for a certain period of time using the latest price charts. (H2)

1.1 Introduction to Cloud Computing

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models [NIST 01]

Configurable computing resources are:

- servers
- storage
- applications
- services

Characteristics of Cloud Computing

Some common characteristics of cloud systems are:

- Massive scalability: the cloud enables adding resources at a very large scale, which may otherwise be impossible with the same resources.
- Resilient computing: the cloud enables nearly zero downtime and continuous availability of software systems.
- Homogeneity: the cloud provides an ability to have homogenous virtual systems even when the underlying hardware in non-homogeneous
- Geographic distribution: the cloud data centers are distributed across the globe.
- Virtualization: gives providers the ability to virtualize servers, storage, or other physical hardware or data center resources.
- Service orientation: the cloud provides various features as software services, which can take advantage of the scale that the cloud provides.
- Low-cost software, the average unit and per-use cost of cloud tends to be low in relation to other options not utilizing the cloud.
- Advanced security, the cloud provides various network-level and application-level security provisions.

Five essential characteristics of cloud systems are:

- On-demand self-service: giving access to various services (such as computing capabilities, storage, software, etc.) without any intervention from anybody as and when required by the consumer.
- Broad network access: all services provided by the cloud are accessible over the network using standard protocols.
- Resource pooling: the resources such as processors, storage, virtual machines and network bandwidth are pooled and provided to the users.

- Measured Service: the cloud has built-in metering, which can charge the users as services are used.
- Rapid elasticity: the cloud makes available resources that can be reduced or increased based on the requirements of the client.

Cloud Computing deployment Models

The four cloud deployment models are:

- **Private Cloud**: exclusively for an organization, installed on premise/off premise. Provides high security and high degree of control
- **Public Cloud:** for general users, where the users are charged on the basis of usage. It is Less secure and more vulnerable to attacks
- **Community Cloud:** implemented jointly by many organizations with shared concerns
- **Hybrid Cloud:** combination of two or more types of clouds. Useful to provide the high security of private cloud and the availability of public cloud for spikes and load balancing

1.2 Cloud Computing Advantages and Disadvantages

As with anything, cloud computing comes with its advantages and disadvantages, which should be taken into consideration prior to any implementation. These advantages and disadvantages may vary, based upon the specific needs of the system/software at hand.

Typical advantages of cloud computing are:

Advantage	Explanation
• Location-Independent Access	Team members can have access anywhere in the world, providing flexibility for teams working from home or other parts of the world.
• Cost Reduction	Cloud enables reduction of up-front investment (CAPEX), Total Cost of Ownership (TCO), and Total Operational Cost (TOC).
• Dynamic Infrastructure	Provides reduced costs and improved services with less development and maintenance costs overall.
• Increased Flexibility	On-demand, flexible, scalable, improved, and adaptable services in a pay-as-you-go model, where services can be focused on what is actually needed.
• Reliable Performance	Consistent availability and performance with automatically provisioned peak loads to allow teams to focus more on the product/software itself.

• Faster Recovery and Resilience	Cloud enables higher resilience and faster recovery by virtue of launching new instances automatically
• Scale	Unlimited processing, storage, networking capabilities which can be quickly ramped up or down.
• Better support	Automatic software updates, improved document format compatibility, and improved compatibility between different operating systems.
• Improved Teamwork Capabilities	Easier facilitation of collaboration throughout the team.

Typical disadvantages or drawbacks of cloud computing are:

Disadvantage	Explanation
• Higher Connectivity Requirements	Cloud computing requires high speed network and connectivity.
• Potential Security Risks	There may be privacy and security issues. The data and applications on a public cloud might not be very secure.
• Increased External Dependency	Users have external dependency for mission-critical applications.
• Increased Monitoring Required	Cloud computing requires constantly monitoring and enforcement of service level agreements (SLAs).

1.3 Service Models in Cloud

XaaS (Anything as a Service) is a collective term that refers to the delivery of "anything" as a service. It recognizes the vast number of products, tools, and technologies that are now delivered to users as a service. Any IT function can be transformed into a service for enterprise consumption. The service is paid for in a flexible consumption model, rather than as an upfront purchase or license over the internet.

The three main forms of services are:

- **Software-as-a-Service (SaaS):** in this multitenant service model, the users use applications running on a cloud infrastructure.
- **Platform-as-a-Service (PaaS):** the provider delivers to the user a platform that includes all the systems and environments comprising the software development life cycle.

- **Infrastructure-as-a-Service (IaaS):** the infrastructure is made available to the user, and it can be used over the internet.

Other Popular Services include STaaS (Storage as a Service), DBaaS (Database as a Service), FaaS (Function as a Service), DaaS (Desktop as a Service) and DRaaS (Disaster Recovery as a Service). As there is now an endless list, simply consider this concept for services that may be required.

1.4 X as Code

XaC (X as Code) is a term that refers to the ability to manage various activities as code. Some of the examples of the same are

- **Infrastructure as Code (IaC)**: is the management and provisioning of infrastructure through code instead of relying on manual processes.
- **Platform as Code (PaC)**: is the management and provisioning of the execution environments necessary for the application through code.
- **Compliance as Code (CaC) (Policy as Code)**: codification of the compliance controls so their adherence, application, and remediation can be automated.

1.5 Cost of Cloud Computing

Cloud enables cost-effectiveness by charging the users only for the services and resources they use by enabling pay-as-you-go model. Users get charged for the type of CPU, amount of RAM, and storage and for the time they use these resources. Similarly, features/services used, for example, in SaaS/PaaS, also get charged on the usage basis.

However, there is a risk of unintended costs when servers that are left running use resources that are not being actively or intentionally used by the customer.

Public versus Private versus On-prem

Migration to the cloud almost eliminates the higher upfront cost which are usually non-cloud projects. However, it can increase the cost of data communication on the network. Cloud computing provides lesser cost for CPU-intensive jobs than data-intensive jobs. Data intensive applications can perform better when employed on a private cloud rather than a public or hybrid cloud.

Chapter 2 - Cloud Technology and Terminology (210 Minutes)

Keywords: Virtualization, Hypervisors, Hyperscalers, Load Balancing, Cloud Servers, Containerization, Docker, Docker CLI, Kubernetes, Kubernetes Cluster, The Master, Nodes, Pods

LO-2.1.1	Differentiate between type I and type II hypervisors. (K2)
LO-2.1.2	Compare some popular hyperscalers with respect to the services offered by them. (K2)
LO-2.2.1	Explain cloud load balancing and how it helps improve system performance and fault tolerance by distributing traffic over multiple virtual servers. (K2)
LO-2.2.2	Demonstrate autoscaling for sudden increases in the workload and subsequent reduction after a period of time. (K2)
LO-2.3.1	Explain key concepts of containers, container provisioning, and container orchestration (K2)
LO-2.3.2	Summarize the potential advantages and drawbacks of containers. (K2)
LO-2.3.3	Recall the role of docker (as a tool) that helps in easily creating, deploying and running applications by using containers. (K1)
LO-2.3.4	Compare containers with virtual machines. (K2)
LO-2.3.5	Outline the Kubernetes architecture. (K2)

HO #	Description
HO-2.1.1	Set up a cloud instance on a hyperscaler. (H1)
HO-2.2.1	Create a load balancer and verify that it balances load appropriately. Use a tool to monitor the load balancer. (H1)
HO-2.3.3	Utilize the Docker CLI to executive basic commands like running and stopping a container. (H2)

HO-2.3.4	Build a customized Docker image and deploy it on the cloud. (H1)

2.1 Virtualization, Hypervisors, and Hyperscalers

Virtualization is the key technology enabling the cloud revolution. Hypervisors enable virtualization and cloud providers use them to enable scaling at a hyper level hence the term hyperscaler.

Virtualization is the creation of a virtual version of a storage device, an operating system, a server, or network resources. It is one of the key technologies behind cloud computing as a whole.
Hypervisors are software tools used to create and manage **Cloud Servers**.

- Type 1 hypervisors / Bare-Metal Virtualization: hypervisors that run directly on the system hardware
- Type 2 hypervisors / OS virtualization: hypervisors that run on a host operating system that provides virtualization services

Hyperscalers refers to the systems that provide the ability to scale as the demand increases. There are many different hyperscaler providers.
Various hyperscalers differ in terms of resources, types of services offered, tools, billing etc., so it is important to review several before making a commitment to a provider.

2.2 Load Balancing

Load Balancing is used for distribution of computing workloads across multiple resources (CPUs, computer clusters, etc.) to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Load balancing can be performed using dedicated software or hardware, as well as using the cloud.

Using a load balancer can be seen as a service continuity guarantee, as it can handle high traffic and/or request spikes. For cloud load balancing, at least two cloud servers are required.

Applications having multiple tiers **require multi-tier load balancing that utilizes both** internal and internet-facing load balancers. Internet facing load balancers receive user requests and typically pass them on to the webservers whereas internal-facing load balancers are typically used for servers on the internal network with private IP addresses.

2.3 Containerization

Containerization is the packaging together of software code with all of its necessary components like libraries, frameworks, and other dependencies so that they are isolated in their own "container." Containers do not need hypervisors and run directly within the host machine's OS kernel.

Containers provide the following advantages:

- **Flexible:** you can containerize even the most complex applications.
- **Lightweight:** you can use containers to leverage and share the host kernel.
- **Interchangeable:** you can deploy updates and upgrades on the fly.
- **Portable:** you can build locally, deploy to the cloud, and run anywhere.
- **Scalable:** you can increase and automatically distribute container replicas.
- **Stackable:** you can stack services on the fly.

Container versus Virtual Machines

A container runs natively on an operating system (OS) and shares the kernel of the host machine with other containers; a virtual machine (VM) runs a full-blown "guest" OS. VMs consume more resources than containers do but provide more isolation, thus more security over containers. Containers, in general, can be run on VMs whereas the reverse may not be true. Containers are therefor used to create highly scalable solutions in the cloud environment.

Docker

Docker is an example of one of the global leading platforms for building applications using containers. Docker containers are created from a read-only template called a "Docker image" and is run using a "Docker Engine", which is a client-server application. A Docker registry stores Docker images. A Docker Hub and a Docker Cloud are public registries that anyone can use.

Kubernetes

Kubernetes is an open-source platform that orchestrates the placement (scheduling) and execution of application containers within and across computer clusters. It helps make sure that containerized applications run where and when they are required and have the resources and tools they need to function properly.

Kubernetes Clusters are a cluster of computers that are connected to work as a single unit and consists of two types of resources:

- The **Master** that coordinates the cluster; and
- **Nodes**, which are workers that run applications.

Instead of managing containers individually, Kubernetes containers are housed into **Pods** for scheduling and execution. These are the unit of replication.

Applications built for cloud from the ground-up make use of cloud specific architecture that utilizes containers and container orchestration tools to provide robust, scalable systems.

Chapter 3 - Cloud Testing (660 Minutes)

Keywords: Cloud Testing, Conventional Testing, Cloud-Based Software Testing, Private Cloud, Public Cloud, On-Prem Cloud, Scalability, Service-Level-Agreements (SLAs), Hybrid-Cloud, Monitoring, Alerting, Load Testing, Elasticity Testing, Resiliency Testing, Security Testing, Reliability Testing, Availability Testing, Disaster-Recovery (DR) Testing, Multi-Tenancy Testing, Compliance Testing, (Data-) Migration Testing, End-to-End (E2E) Testing, Syntax, Semantics, Interoperability, Portability, Access Management (IAM), Data Loss Prevention (DLP), Encryption, Security Information and Event Management (SIEM), Data Privacy, ISO, Recovery Time Objective (RTO), Recovery Point Objective (RPO)

LO-3.1.1	Outline the differences between cloud testing and conventional testing. (K2)
LO-3.1.2	Illustrate the benefits of cloud testing over conventional testing. (K2)
LO-3.1.3	Explain the various types of cloud testing, namely testing applications on the cloud, testing the cloud, testing across clouds, and testing using the cloud. (K2)
LO-3.2.1	List some of the key test types involved in testing the cloud. (K2)
LO-3.3.1	Explain End-to-End (E2E) testing in the context of cloud. (K2)
LO-3.3.2	Explain data testing in the context of cloud. (K2)
LO-3.3.3	Explain compliance testing in the context of cloud. (K2)
LO-3.3.4	Explain multi-tenancy testing in the context of cloud. (K2)
LO-3.3.5	Explain security testing in the context of cloud and containers. (K2)
LO-3.3.6	Explain monitoring and alerting testing in the context of cloud. (K2)
LO-3.3.7	Explain performance testing in the context of cloud. (K2)
LO-3.3.8	Explain reliability and availability testing in the context of cloud. (K2)

LO-3.3.9	Explain elasticity and scalability testing in the context of cloud. (K2)
LO-3.3.10	Explain interoperability and portability testing in the context of cloud. (K2)
LO-3.3.11	Explain disaster-recovery (DR) testing in the context of cloud. (K2)

HO #	Description
HO-3.3.1	Perform functional testing of an app, accessing the cloud database and run queries. (H2)
HO-3.3.2	Perform multitenancy testing for a cloud-hosted app with defects. (H2)
HO-3.3.3	Perform security testing for a cloud-hosted app using a tool. (H1)
HO-3.3.4	Use a monitoring tool to set up and test monitoring and logging. (H1)
HO-3.3.5	Use a load testing tool to generate load for a cloud-hosted app. (H1)
HO-3.3.6	Perform scalability test for both — scale up and scale down scenario. (H1)
HO-3.3.7	Perform testing for a disaster recovery scenario. (H1)

3.1 Cloud Testing versus Conventional Software Testing

Cloud testing is a form of software testing in which web applications use cloud computing environments (a "cloud") to simulate real-world user traffic. [Wiki1]. It includes testing for both - functional and non-functional requirements.

Cloud-Based Software Testing ensures the quality of functions and performance of SaaS/PaaS/IaaS applications. It involves, among other things, the testing for elasticity and scalability-based SLAs, testing for cloud offered

services and characteristics and also testing for multitenancy, disaster recovery, compliance… etc.
Conventional Software Testing is testing the software for cloud testing minus cloud specific testing; however, it may include testing that leverages the cloud infrastructure

General advantages of cloud testing are:

- **Reduction of execution time:** Cloud environments are easy to setup and teardown reducing overall time for test execution
- **Easier to access environments:** Cloud being available via internet, it is possible to access the environments from anywhere
- **Ease of deployment:** Infrastructure as code and other cloud services make it easy to deploy the applications on the cloud
- **Ease of management:** Cloud has various management interfaces for managing the test environment, tools etc.
- **Cost reduction, including tool costs:** Pay as you go helps reduce the costs
- **Increased scalability:** Scalability of cloud helps scalability for tools esp. load generation and parallel execution of automation

Types of Cloud Testing

- Testing the applications on the cloud including testing for migration to Cloud and testing across clouds — private, public cloud, on-prem
- Testing using the cloud
- Testing the cloud itself

Testing Applications on the Cloud
This is when applications that are hosted on the cloud are tested. These could be developed natively for the cloud or applications that have been migrated to the cloud. Some applications may be hybrid cloud application - that may be using more than one cloud type (public, private) or hosted on at least two different cloud providers.

Testing the Cloud
Cloud providers test their cloud infrastructure for the Service-Level-Agreements (SLAs) that they offer to their customers. They main focus is to

correct functioning of the cloud platform itself, including billing, security of the cloud infrastructure, and other non-functional characteristics.

Testing Using the Cloud
Various tools are hosted on the cloud. Utilizing these tools and the cloud infrastructure to aid in testing is "testing using the cloud." This involves using tools for cloud-based load generation, defect and test management, and security testing. There are several advantages of testing using the cloud: for example, the elimination of upfront investments on tools and infrastructure, the creation of real-world situations through simulation of geographically distributed load patterns, and the facilitation of on-demand performance testing for organizations.

3.2 Types of Tests for Cloud-Hosted Applications

Various tests for cloud-hosted applications involve both, functional and non-functional tests. Some of the test types for testing the applications hosted on the cloud are

- Scalability
- Service-Level-Agreements (SLAs)
- Testing of containers including hybrid-cloud scenarios
- Testing for monitoring and alerting
- Load testing
- Scalability, elasticity, and resiliency testing
- Security testing
- Reliability and availability testing
- Disaster-Recovery (DR) testing
- Multi-tenancy testing
- Compliance testing
- Migration testing including data-migration testing
- End-to-End (E2E) testing

3.3 Cloud Specific Tests

End-to-End (E2E) Tests in Cloud Computing
E2E tests are system integration tests where multiple systems, that work together, are tested with focus is on the business processes. Both functional and non-functional tests are performed.

Typical issues found during E2E tests are:

- **Syntax:** the data inconsistencies across systems.
- **Semantics:** the wrong interpretation of data, for example, currencies, date interpretation etc.
- **Schema Inconsistencies:** inconsistent use of limits, max and min, cardinality, mandatory versus optional.
- **Interface Inconsistencies:** if utilizing connected systems, the interfaces may become incompatible.
- **Performance and Timeouts:** the system may perform poorly, which make result in the occurrence of timeouts.
- **Robustness:** migrated systems may perform poorly against bad inputs.
- **Security:** considering both application security and network security.
- **Configuration Issues:** if default passwords are left or wrong configuration settings are maintained.

Some of the challenges of E2E tests are:

- Test environment:
  - Not realistic/representative enough to achieve the goal.
  - No control over the third-party systems
  - Non-availability of simulators/stubs
- Unannounced changes in the third-party systems
- Lack of knowledge of the complete business processes that are associated

Data Testing in Cloud Computing

Data poses some unique challenges on cloud such as:

- the location of the storage
- the latency in accessing certain information
- potential compliance issues
- backups and restore (etc.)

Testing for data when migrating an application is also required. This is covered later and in more detail in the <<CU - Migration Test Specialist course: https://www.cloud-united.org/cu-courses.

Tests related to data may include:

- Data location tests (where is the data is located for your software/application)

    - **On-prem:** test for data accessibility, security, data transfer time and costs.
    - **Cloud**: data storage costs, storage scalability, SLA verification including, data access performance, data availability and application recovery time needs, local data storage law compliance.
    - **Hybrid cloud deployments**: data may be on premise or in the cloud depending upon data retention policy of the organization, local data storage law compliance.
- Dependency on cloud vendor provided services for data storage, access, and retrieval
- SLA of cloud service for the data access (CRUD) as well as data restore
- Type of data storage
- Data ownership, interoperability, and portability
    - Data created by the user
    - Data that is created as part of using cloud-based computing resources
- The customer:
    - Switches between service providers
    - Utilizes user services of multiple providers
    - Links in-house services with service provider
    - Migrates from in-house services to cloud, etc.

Multitenancy Testing in Cloud Computing

Multitenancy testing involves the testing of apps where there is a single instance of an application serving multiple non-related customers. The sharing can be in relation to infrastructure, platform, database, or application instance. This above-mentioned sharing may lead to functional, billing, security, reliability, and performance issues, which is why thorough testing is required.

Some of the challenges of multitenancy testing are:

- Varying workloads: as multiple tenants may result in a different overall user experience in production as compared to testing.
- Cloud vendor specific SLAs, where information is needed to ensure the correct measures are taken.
- Increased difficulty in isolation testing for public clouds, which may result from inadequate failure containment between tenants.

Compliance Testing in Cloud Computing
Compliance testing is highly regulated in industries such as Healthcare and Banking, as they have to adhere to laws and standards of compliance, for example, BASEL III and HIPPA. There are also compliance practices and norms related to data security and access rights. Many of these may be impacted by use of third-party cloud-based computing resources. In addition, applications may store data on the cloud that has to adhere to regulatory compliancy laws based on the national laws.

Typical compliance concerns in the cloud are generally related to location of the data, access rights, administrative rights, security of the platform, and segregation of data, especially in case of multitenancy.

Key items customers need to verify in cloud service contracts are related to data ownership, cloud provider SLAs for performance, security, high availability, and disaster recovery, cloud provider compliance accreditations, and the provider's plan for responding to incidents.

Some cloud platforms may provide various tools and services related to compliance that can be very useful.

Security Testing in Cloud Computing
Security is a major concern with the cloud. Some of the aspects of cloud security are related to Identity and Access Management (IAM), Data Loss Prevention (DLP), encryption, Security Information and Event Management (SIEM).

When it comes to security there is a clear division of responsibility. The customer is responsible for security of their own applications, OS instances, data, network and firewall configurations, encryption, and key security. The platform provider is responsible for resources provided by them, such as infrastructure, compute and storage, network (etc.). However, it is important to keep in mind that regardless of who is declared responsible, security issues have a negative impact on all parties involved and need to be given utmost respect.

Some of the causes of security and privacy issues may result from the movement of data and the application on networks, various governmental security policies, internal and external security threats, loss of access and privacy concerns of customers. It is important to consider that the physical

security of the cloud hardware, typically, is not a main concern of the customer and may be overlooked.

Some common security threats are:

- Data privacy
- Data retention and improper destruction
- Data security
- Data compliance
- Network security
- App security
- Storage security
- Identity, authentication, and access management
- Vulnerable public APIs
- Account takeover
- Denial-of-service attacks
- User access roles misconfiguration
- Multitenancy penetration

Some of the applicable security standards are:

- ISO/IEC 27004:2009 — Information Security Management
- ISO/IEC 27017:2015 Information technology — Security techniques — Code of practice for information security controls based on ISO/IEC 27002 for cloud services
- ISO/IEC 19086-4:2019 — Cloud computing
- NIST Special Publication (SP) 800-55 Rev.1, Performance Measurement Guide for Information Security
- CIS Security Metrics v1.1.0
- The Cloud Control Matrix (CCM) Working Group (WG) Implementation guidelines, a cybersecurity control framework for cloud computing

There are additional security tests required if the software/application is container based.

Kernel is used by all of the containers that has a large attack surface. OS vulnerabilities of the container may allow for malicious code to have unauthorized access to the entire system. Different parts of the code may

process data with different security requirements. It is generally considered a best practice to have different containers for the scenarios described above needs. Similarly, container orchestration security also needs to be investigated. For example, in a Kubernetes environment, vulnerabilities in the Kubernetes software could compromise all of the pods that are running on it.

Monitoring and Alerting in Cloud Computing

Monitoring and alerting are two terms that are often used interchangeably, even though they are quite different when it comes to cloud computing. **Monitoring** is the process of maintaining surveillance over the given system for state changes and flow of the data. **Alerting** is the capability of a monitoring system to detect and notify the operators about meaningful events.

Monitoring and alerting are useful for detecting and informing of issues such as errors, service unavailability, performance, and security issues, which can have a detrimental impact on software/applications. It can also be used to detect usability issues that are based on understanding of user behavior, which may change over time.

To test monitoring and alerting one needs to test the configuration steps and options and producing situations that are to be monitored and alerted. Testability of monitors is a key requirement. Tests should be conducted for the modification of alerts and removal of monitors. Logs should also be tested for size, archival, confidential data encryption/masking, and appropriate classification of events (info, warning, error, etc.).

Performance Testing in Cloud Computing

Cloud-specific performance issues may be result from a lack of proper resources, for example, disk space, limited bandwidth, lower CPU speed, memory, and/or network connections. Hybrid cloud may suffer from poor performance if they are not planned properly because of bottlenecks, latencies, and a lack of resources.

Verification of the response time while accounting for the network latency and its impact on the performance numbers is a key test. When there is a reduction in load workload should be balanced again and this too needs to be tested.

There are many ways to conduct performance testing:

- An application on the cloud, load generator on machines in the lab: this is useful when load testing labs already exists on-prem and network bandwidth is not an issue.
- An application on the machines in the lab, load generator on cloud: this is when a load generator tool is to be used in a SaaS model because of licenses or lab infrastructure that can be dynamically grown on the cloud.
- Both, when the application and the load generator are on the cloud — this makes it easy to setup and dismantle the test environment, which may also result in lower latency.

Reliability and Availability in Cloud Computing
Reliability denotes how often resources are available without disruption (loss of data, code reset during execution) and how often they fail. **Availability** is the possibility of obtaining the resources whenever they are needed with the consideration of the time it takes for these resources to be provisioned.

Elasticity and Scalability in Cloud Computing
Elasticity is the ability to scale resources both up and down as and when required. **Scalability** can be defined as the ability of the system to perform well even when the resources have been scaled up. Essentially one could consider that it is elasticity that enables scalability.

Scalability can be provided in two ways:

- **Horizontally (Scale Out):** more nodes/machines are added
- **Vertically (Scale Up):** existing node/machine is upgraded with more resources, for example, higher CPU/more RAM etc.

Elasticity testing involves checking the maximum limits and auto provisioning/freeing resources on the go. It also involves checking the impact of autoscaling on the performance. Various types of load patterns can be tried while testing elasticity.

Interoperability and Portability in Cloud Computing
Cloud **Interoperability** is the ability to use the same tools (or application) across various cloud service providers platforms. The interoperability can be defined at various levels, viz. application, service, management, and data interoperability.

Cloud **Portability** ensures that one cloud solution will be able to work with other platforms and applications, as well as with other clouds. Vendor lock-in is one of the biggest risks leading to a of lack of portability. A vendor lock-in is when there is a tight coupling to a specific cloud provider.

Disaster Recovery in Cloud Computing

Disaster recovery (DR) refers to the preparation for and recovering from a disaster. Disaster scenarios include hardware or software failure, network outage, a power outage, physical damage to a building like fire or flooding (etc.), human error, or some other significant event (often considered an Act of God).

There are several DR options in the cloud such as managed application and managed disaster recovery, backup to and restore from the cloud, backup to and restore to the cloud, or replication to virtual machines in the cloud etc.

There are two main DR objectives in cloud computing:

- Recovery Time Objective (RTO): this is the time taken to recover from the disaster.
- Recovery Point Objective (RPO): this is the amount of acceptable data loss, in term of number of hours.

Some of the requirements from DR infrastructure are related to the physical protection of assets, capacity to scale, contractual agreements. network infrastructure and enough server capacity to run all mission-critical services. Each of these aspects needs to be tested accordingly.

Failing back is the process of restoring the normal service of the primary site without the loss of data after a disaster has occurred on a given system.

Some of the tests that are required for failing back are related to:

- Loss of transactions
- Loss of data
- Status of the ongoing transactions when the DR occurred

In cloud computing, when it comes to testing, there are many aspects that can be tested the same way as in any conventional software/application. As mentioned throughout this chapter, there are also many areas that require special consideration and care to ensure that all the cloud relevant aspects are

considered throughout the entire quality assurance process to maintain the quality of the software/application.

Chapter 4 — Cloud Migration (210 Minutes)

Keywords: (Cloud) Migration, Legacy to Cloud, Cloud to Cloud, Cloud to Hybrid Cloud, Migration Tests, Functional Testing, Non-Functional Testing, Data Migration Testing,

LO-4.1.1	List various types of migration with respect to the cloud. (K1)
LO-4.1.2	Explain the challenges associated with cloud migration. (K2)
LO-4.1.3	List various cloud migration strategies. (K1)
LO-4.2.1	Explain the types of tests for migration to the cloud. (K2)
LO-4.3.1	Outline various types of data migration tests. (K2)

HO #	Description
HO-4.1.1	Test a re-platforming scenario, where data is migrated from a database on one platform to another. (H1)

4.1 Introduction to Cloud Migration

There are various types of cloud migrations that are based on what is being migrated (application, database server, or operating system) or on the basis of the source and the target (Legacy to Cloud, Cloud to Cloud, Cloud to Hybrid Cloud) of the migration.

Challenges of Migration to Cloud

Migrating to the cloud has its own set of challenges and risks, such as data integrity, security, privacy, and business acceptability, which can be mitigated through adoption of additional procedures.

Various Migration Strategies

There are six generally accepted strategies for migration:

- **Re-Host**: moving the application from on-prem to cloud without changes.
- **Re-Platform**: changing one or more components to the cloud-based components.
- **Re-Factor/Re-Architect**: rewriting the application to make better use of the cloud specific architecture such as micro-services etc.
- **Re-Purchase**: purchasing a SaaS solution and migrating all the data to it.

- **Retire**: stopping the usage given software.
- **Retain**: continuing the usage of given software

4.2 Types of Migration Tests

Types of **Migration Tests** can be divided into functional and non-functional testing:

- Functional Testing
  - Pre-migration testing: testing prior to migration and testing the cloud infrastructure to which the migration will occur.
  - Functional validation: validating the functionality.
  - Data validation: performing data validation tests.
  - Integration testing: E2E testing with third-party applications.
- Non-Functional Testing
  - Scalability
  - Application performance
  - Resilience
  - Disaster recovery and business continuity plan
  - Application security
  - Cut-over & go-live certification

4.3 Data Migration Testing

Data Migration Testing is a verification process of the migration of the legacy system to a new system to ensure minimal disruption/downtime. The overall goal is to migrate while maintaining the integrity of the date (e.g., no loss of data) and ensuring that all the specified functional and non-functional aspects of the application are met post-migration.

Data migration testing includes testing with old data, new data, or combination of both. It also includes the testing of old features (unchanged features) and the new features which are being implemented. It involves check the migrated data for syntax, semantics, numbers, and various other parameters. It forms an important activity because most migration activities involve data migration.

Chapter 5 — Advanced Topics (90 Minutes)

Keywords: Cloud Native Applications, Microservices Architecture, Serverless Architecture, Communication Testing, GUI-Based Testing, API-Based Testing, Orchestration Testing, Circuit Breaker Testing, Contract First Testing, Performance Testing, Security Testing, Scalability Testing, Resilience Testing, Observability Testing, Container Testing, Infrastructure as Code (IaC), Static Testing, Unit Testing, System Testing, System Integration Testing, Blue/Green Deployment, Testing in Production (TiP), A/B Testing, Canary Testing, Rolling Update Testing, Chaos Testing

LO-5.1.1	Explain what a cloud-native application is, as well as its components. (K2)
LO-5.1.2	List various types of tests for testing microservices. (K1)
LO-5.1.3	Enumerate various aspects of container testing. (K1)
LO-5.2.1	Define the concept of Infrastructure as Code (IaC). (K1)
LO-5.2.2	Differentiate between imperative and declarative ways to approach IaC. (K1)
LO-5.2.3	Enumerate various test levels for IaC. (K1)
LO-5.3.1	Summarize the various types of tests carried out in Testing in Production (TiP). (K2)
LO-5.3.2	Explain the implementation of and requirements for chaos testing. (K2)

HO #	Description
HO-5.1.1	(Optional) Do a demo of a container/microservices testing tool. (HO-0)
HO-5.2.1	(Optional) Do a demo of a IaC testing tool. (HO-0)
HO-5.3.1	(Optional) Do a demo of a chaos testing tool. (HO-0)

5.1 Cloud Native Applications

A cloud native application is designed to take advantage of the characteristics of a cloud computing software:

- Microservices-based
- Serverless architecture (e.g., AWS Lambda)
- Container-based (e.g., Docker)
- Dynamically orchestrated (for example, using Kubernetes)

Microservices Architecture

Microservices are service-oriented architecture patterns where applications are built as a collection of various small independent service units. Microservices architecture is a software engineering approach that develops an application with single-function modules and has well-defined interfaces. These modules are independently deployed and operated by small teams who own the entire lifecycle of the service.

Serverless Architecture

A serverless architecture is a way to build and run applications and services without having to manage different types of infrastructure. The application still runs on servers, but all the server management is done by service providers. It is not required to provision, scale, and maintain servers to run your applications, databases, and storage systems.

Types of Tests for Cloud Native Applications:

There are two types of tests for cloud-native applications.
Functional tests:

- API Testing: the process of testing while utilizing APIs.
- End-to-End Testing: as described in section 3.3 Cloud Specific Tests
- Orchestration Testing: when container orchestrators automate container management and requires testing.
- Circuit Breaker Testing: the process of checking (testing) the circuit breaker. A circuit breaker is an architectural pattern where a failing service results in the immediate failure of a remote invocation for a pre-determined period of time to prevent excess consumption of resources.
- Contract Testing: the performance of tests to check that the services can communicate with each other without using exact test data.

Non-functional tests:

- Scalability and Performance Testing
- Resilience
- Observability
- Security

Container Testing
Containers do not change the functionality of the software/application, hence there is no difference in the general functional testing approach. Functional automation tools can be used to automate GUI-based or API-based testing. However, it is important to note that container engines on multiple platforms may result in changed behavior, which may require tests based on operating system and multiple versions of the container engines. Performance, security, and certification tests should also be performed on containers. There are many tools available for performing static and dynamic tests on containers.

5.2 Infrastructure as Code (IaC)

Introduction to IaC
Infrastructure as Code (IaC) refers to the management of infrastructure (networks, virtual machines, load balancers, and connection topology) in a descriptive model using a versioning tool, for exams in the source code. An IaC automation tool could read the file and build a system based on the specifications of a given user. In general, an IaC model should generate the same environment every time it is applied.

Some of the most commonly used tools for IaC are Terraform, AWS CloudFormation, Azure Resource Manager, and Google Cloud Deployment Manager. Some of the tools used for testing are Terrascan, Terratest, Ansible Lint, Cfn-Lint. Please note that this is not an exhaustive list.

Approaches to IaC
There are two approaches to IaC:

- **Declarative approach**: One defines the desired state of the system, including the required resources and their properties. The tool then creates the infrastructure.
- **Imperative approach**: Specific commands to achieve the desired configuration are listed and executed in the correct order to create the infrastructure.

Different Test Levels for IaC
There are several different test levels used for testing IaC:

- Static Testing
- Unit Testing
- System Testing
- System Integration Testing

5.3 Testing in Production (TiP)

Types of TiP Tests

- A/B Testing: is when two versions of a website, app, or feature are released to gauge if users prefer one over the other.
- Canary Testing: is when the rolling out the product is done incrementally to a larger and larger set of users to prevent large scale failures and get early feedback.
- Blue/Green Testing: is when deployment is done on a system that is not live and fully tested then the router is switched to direct all incoming requests to this fully tested system.
- Rolling Update Testing: is when a subset of the running application is updated instead of simultaneously updating every application instance, to contain the number and type of failures.

Chaos Testing
Chaos testing is testing a system's integrity by proactively simulating and identifying failures in a given environment.

Chaos testing:

- Requires a set of tools to inject faults and monitor the system.
- Needs to be run in a production environment to get the correct results. Test environments can almost never be similar to the production environment.
- Some of the tools used are — Chaos monkey, Gremlin etc.

Complexity of the cloud is increasing to cater to the increasing demands placed on the software applications. The demand in general for availability and resilience are also increasing. This means that the field of cloud testing is also ever-evolving along with the technologies required to accomplish it. A cloud tester needs to be aware of the latest developments related to the cloud

technologies, the tools that are readily available, and how these impact the testing of applications.

References

This document has been designed and created utilizing first-hand experiences gathered from the industry by the SMEs involved in creating Cloud United.

- **[A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives]** — L. Anderson, P. W. Airasian, and D. R. Krathwohl (Allyn & Bacon 2001)
- **[Revised Bloom's Taxonomy Action Verbs]** Available at https://www.apu.edu/live_data/files/333/blooms_taxonomy_action_verbs.pdf
- **[Cloud for the Modern Enterprise]** Winning Practices to Transform Legacy IT Organizations — Mirco Hering (April 2018)
- **[The Cloud Handbook]** How to Create World-Class Agility, Reliability, & Security in Technology Organizations — Gene Kim, Jez Humble, John Willis & Patrick Debois (October 2016 edition)
- **[Embedding Cloud in the Enterprise]** Cutter IT Journal (November 2011 edition)
- **[Cloud Guide]** The IT Revolution (2015 edition)
- **[Cloud for Dummies]** IBM — Sanjeev Sharma & Bernie Coyne (John Wiley & Sons, 2nd edition 2015)
- **[ISTQB-FL 2018]** ISTQB Foundation Level Syllabus version 2018. Available at https://www.istqb.org/downloads/category/51-ctfl2018.html
- **[Agile Alliance organization]** https://www.agilealliance.org
- **[The NIST Definition of Cloud Computing]** Available at https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf
- **[Wiki1]** - https://en.wikipedia.org/wiki/Cloud_testing].