1. Explain the risks and issues associated with cloud. How do you manage these risks?

Risks or issues with the cloud:

- Privileged user access
- Regulatory Compliance
- Data location
- Data segregation
- Recovery
- Investigative support
- Long term visibility

Risk Management involves:

- <u>Risk identification:</u> The process of identifying potential risks that could negatively impact the project or organization.
- <u>Risk analysis and evaluation:</u> The process of assessing the likelihood and impact of identified risks, and determining their significance to the organization.
- <u>Selection of countermeasures:</u> The process of identifying and selecting appropriate strategies to mitigate or eliminate identified risks.
- <u>Deployment of suitable countermeasures:</u> The process of implementing the selected countermeasures to manage the identified risks.
- <u>Continuous monitoring to assess the effectiveness of the solution:</u> The ongoing process of evaluating the effectiveness of the implemented countermeasures and making adjustments as necessary to ensure the continued management of risks.

2. How do you calculate the total cost of ownership for a cloud environment? OR Explain the three parameters on which cost of ownership for a cloud environment depends on.

When calculating the total cost of ownership (TCO) for cloud computing, there are three important parameters that organizations should consider:

1. **Identifying all cloud cost components:**
    a. **Consider both direct and indirect costs:** When calculating TCO, it is important to consider both direct costs (e.g., compute, storage, and network resources) and indirect costs (e.g., software licenses, employee training, and management overhead).
    b. **Identify all relevant cost components:** To ensure accuracy and completeness, it is important to identify all relevant cost components for

the cloud services being used. This can include costs associated with data transfer, API calls, and third-party services.

c. **Be aware of hidden costs:** In some cases, there may be hidden costs associated with cloud services, such as fees for exceeding usage limits or charges for data access. It is important to be aware of these potential costs and factor them into TCO calculations.

2. Identifying the combination of cloud services:

a. **Identify the specific services being used:** To accurately calculate TCO, it is important to identify the specific cloud services being used, including the type and amount of each service.

b. **Consider the potential for bundling:** In some cases, it may be more cost-effective to bundle multiple cloud services together rather than using them separately.

c. **Evaluate the impact of changes:** It is important to evaluate the potential impact of changes, such as adding or removing services, on TCO.

3. Identifying the variation in utilization:

a. **Consider utilization patterns:** It is important to consider the variation in utilization of cloud services over time. This can include factors such as seasonal spikes in usage or changes in demand due to business growth.

b. **Use data-driven analysis:** To accurately assess utilization patterns, it is important to use data-driven analysis techniques, such as machine learning or predictive analytics.

c. **Evaluate the potential impact of changes:** When considering the variation in utilization, it is important to evaluate the potential impact of changes, such as changes in usage patterns or the introduction of new services, on TCO.

3. Describe direct and indirect cloud costs.

Direct costs in cloud computing refer to the expenses that are directly related to the resources used in the cloud, such as compute, storage, and network resources. These costs are typically charged on a pay-as-you-go or usage-based model, where the customer is billed based on the actual usage of the cloud resources. Examples of direct costs include the hourly cost of a virtual machine, the cost of storing data in a cloud storage service, and the cost of transferring data between different regions or zones in the cloud.

Indirect costs in cloud computing refer to the expenses that are not directly related to the usage of cloud resources but are necessary for using and managing the cloud environment. These costs can include things like software licenses, employee training, and management overhead.

4. How does a pay-per-use model differ from elasticity of a model?

A pay-per-use model, also known as usage-based pricing, is a billing model in which customers are charged only for the resources they use. For example, a cloud provider may charge customers based on the number of virtual machine instances used, the amount of storage used, or the amount of data transferred. The pay-per-use model is beneficial because it allows customers to avoid upfront costs and only pay for the resources they need, making it easier to scale resources up or down based on demand.

Elasticity, on the other hand, refers to the ability of a cloud service to automatically and quickly adjust resources up or down to meet changing demand. Elasticity allows customers to scale resources up or down based on actual usage, without having to worry about provisioning and managing those resources manually. In other words, elasticity is the ability of a cloud service to be flexible and adapt to changing demand without affecting performance or availability.

5. Examine the benefits of chargeback models in terms of “Pay for what you use”.

Chargeback models in cloud computing refer to the practice of assigning costs to different departments or business units within an organization based on their actual usage of cloud resources. In a chargeback model, each department or business unit is billed based on the resources they use, just like in a pay-per-use model. The benefits of chargeback models in terms of "pay for what you use" include the following:

Cost Optimization: Chargeback models can help organizations optimize their cloud spending by encouraging departments to be more mindful of their resource usage and avoid overprovisioning resources.

Transparency: Chargeback models provide transparency into the actual usage of cloud resources, allowing organizations to make informed decisions about how to allocate their cloud spending.

Fairness: Chargeback models can help ensure that each department or business unit pays a fair share of the organization's cloud spending, preventing resentment and mistrust.

Accountability: Chargeback models can help promote accountability for cloud spending by making each department or business unit responsible for its own usage and assigning costs based on actual usage. This can help prevent shared expenses and promote a culture of accountability and responsibility.

6. Discuss the design guidelines for calculating total cost of Ownership for cloud computing?

SAME AS Q2

7. Explain the chargeback methodology for the allocation of direct and indirect costs.

- Chargeback methodology
    - Components
        - Cost
            - CapEx (Capital Expenses)
            - OpEx (Operating Expenses)
        - Billable items
            - VM, Server blade, network service, data service, WAN service, Security service, SLA
        - Atomic units
        - Pricing model
            - Fixed- recurring pricing – Regardless of Untilization
            - Variable- pricing by resource consumption
            - Variable- pricing by time - how long item used
            - Cost multipliers – additional services like encryption, compressing
            - Chargeback tools and solution
                - Data collection tools, arbitration tool, billing system,
                - Capacity analysis tool

8. What is an AAA model? Explain with the help of a diagram.

The AAA model, also known as the AAA framework, is an information security model that stands for Authentication, Authorization, and Accounting. The AAA model is commonly used in computer networks to control access to network resources and ensure that only authorized users are granted access.

Steps:

- The login/security server first checks if the LOGIN NAME and PASSWORD are legitimate. If so, the user is "**authenticated**" and permitted in.

- It then decides the modules of the application that he/she can view. This is called **authorization**.
- Then, the server keeps a **log/account** of all the resources utilized and the user activities.

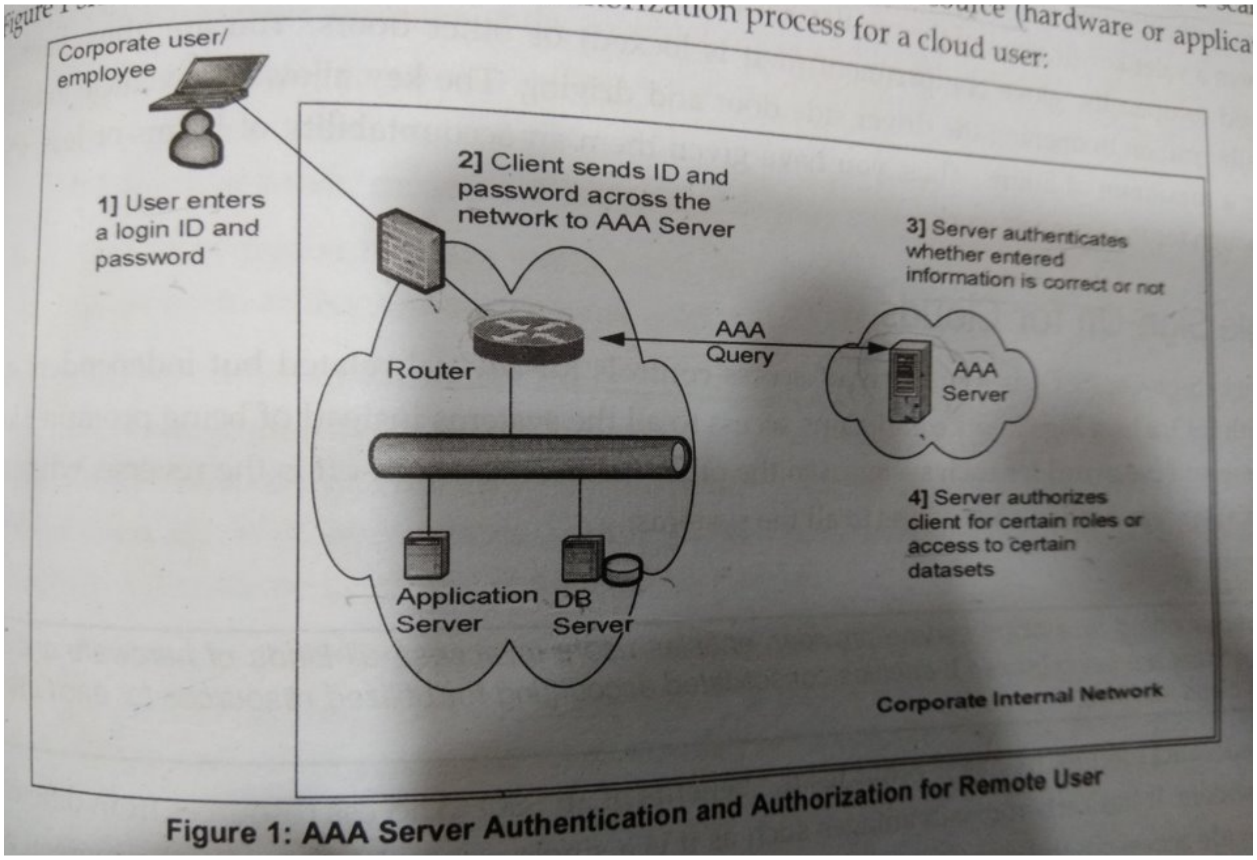

Corporate user/ employee
1] User enters a login ID and password
2] Client sends ID and password across the network to AAA Server
3] Server authenticates whether entered information is correct or not
Router
AAA Query
AAA Server
4] Server authorizes client for certain roles or access to certain datasets
Application Server
DB Server
Corporate Internal Network

Figure 1: AAA Server Authentication and Authorization for Remote User

9. Differentiate between SSO and SAML.

	SSO	SAML
Definition	Single Sign-On (SSO) is a user authentication process that enables a user to log in once and access multiple applications without having to re-enter login credentials.	Security Assertion Markup Language (SAML) is an XML-based open standard used for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider.

Security	SSO typically uses multiple security protocols to ensure secure access to applications.	SAML uses cryptography to ensure the secure exchange of user authentication and authorization data.
Implementation	SSO requires implementing an authentication server that is responsible for authenticating users and handling authorization requests.	SAML requires both the identity provider and service provider to implement SAML protocols for the secure exchange of authentication and authorization data.
Use Cases	SSO is commonly used in enterprise environments where users need to access multiple applications using a single set of login credentials.	SAML is commonly used for federated identity management scenarios where users need to access services across different organizations or domains.

10. What is the role of cookies in SSO?

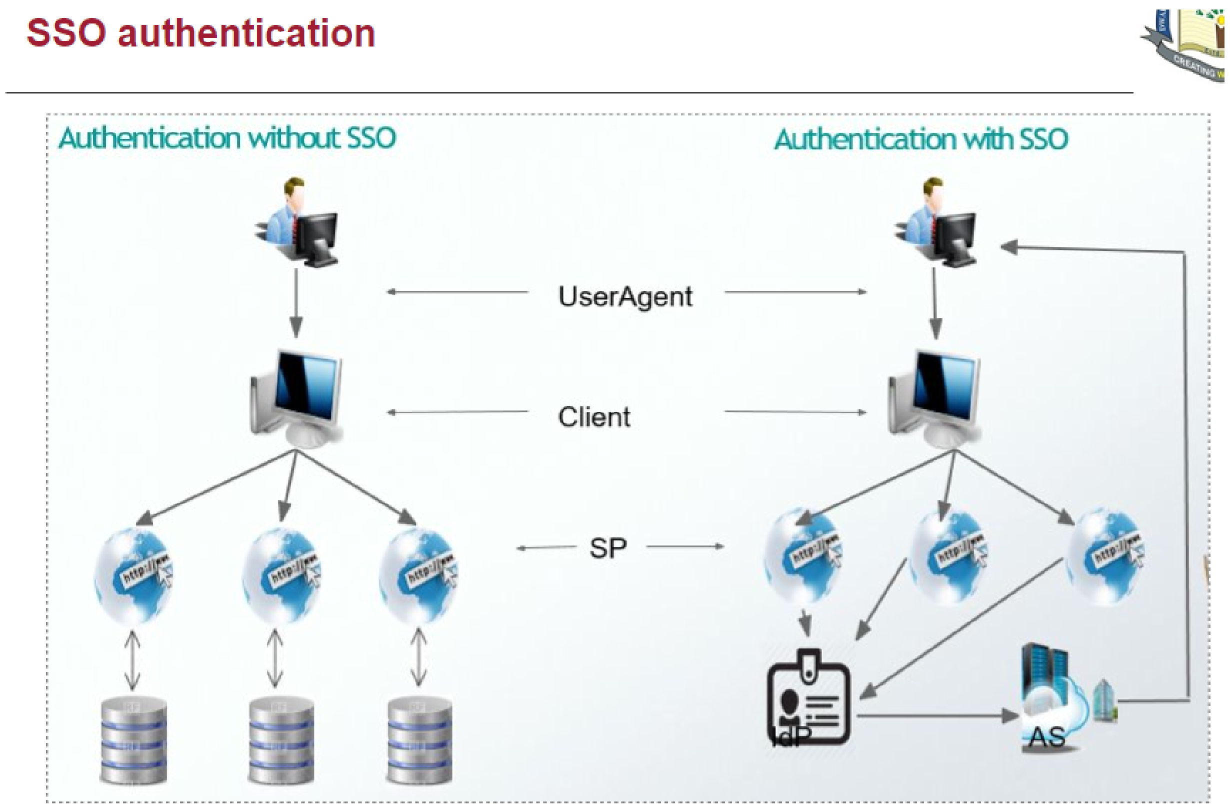
In Single Sign-On (SSO) systems, cookies play a crucial role in maintaining a persistent session between the user's browser and the SSO provider. When a user logs in to a website or application using SSO, the SSO provider creates an authentication token and stores it in a cookie, which is then sent to the user's browser. When the user visits another website or application that also supports SSO, the browser sends the cookie to the SSO provider, which uses the authentication token to authenticate the user without requiring them to log in again.

11. What is the industry implementation of AAA?

1. **RADIUS (Remote Authentication Dial-In User Service Protocol)**
   - UDP-based Communication
   - Used by ISP
2. **TACACS+ (Terminal Access Controller Access-control system)**
   - CISCO Proprietary
   - TCP-based so more reliable
   - Deals Authentication and Authorization as separate 2 tasks
3. **Kerberos:** DES-based

12. Write a short note on SSO. Explain its advantages and disadvantages.

Single Sign-On (SSO) is a user authentication process that enables a user to log in once and access multiple applications without having to re-enter login credentials. SSO typically uses multiple security protocols to ensure secure access to applications. SSO requires implementing an authentication server that is responsible for authenticating users and handling authorization requests. SSO is commonly used in enterprise environments where users need to access multiple applications using a single set of login credentials.



Advantages of SSO:

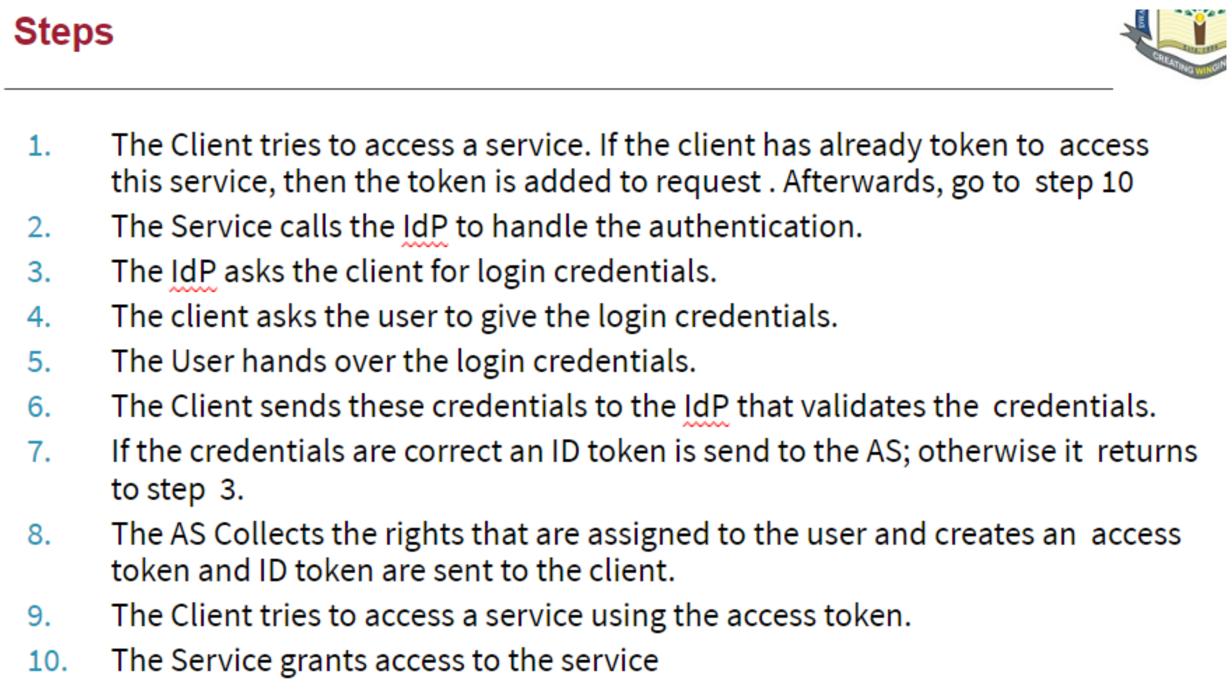
- Improves customer satisfaction
- Boosts productivity
- Improves compliance and security capabilities
- Facilitates B2B collaboration
- Stronger and/or automatic password changes
- Faster access to systems

Disadvantages of SSO:

- Single point of failure

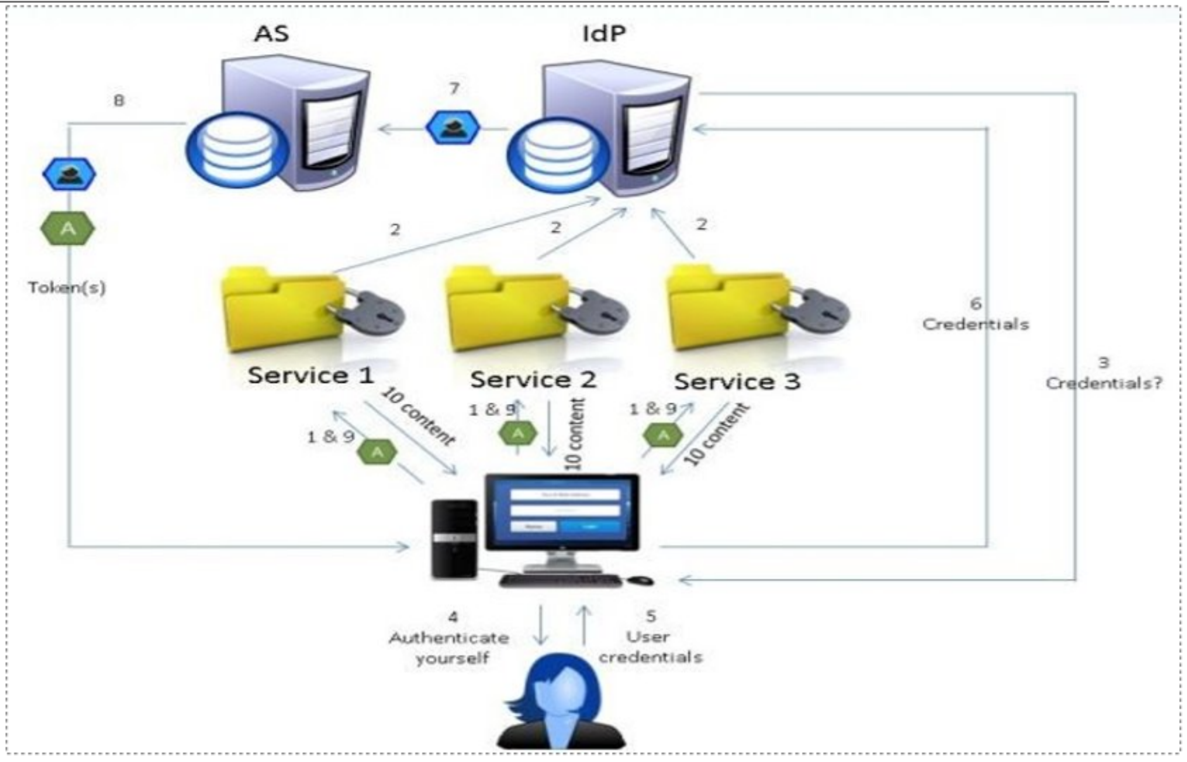
- Single high-value target (attracts more attackers)
- Necessary information disclosure between trusting site and SSO authority
- Lack of control over your user list

13. Explain the steps involved in SSO with the help of a diagram.



Steps

1. The Client tries to access a service. If the client has already token to access this service, then the token is added to request . Afterwards, go to step 10
2. The Service calls the IdP to handle the authentication.
3. The IdP asks the client for login credentials.
4. The client asks the user to give the login credentials.
5. The User hands over the login credentials.
6. The Client sends these credentials to the IdP that validates the credentials.
7. If the credentials are correct an ID token is send to the AS; otherwise it returns to step 3.
8. The AS Collects the rights that are assigned to the user and creates an access token and ID token are sent to the client.
9. The Client tries to access a service using the access token.
10. The Service grants access to the service



14. Analyze and Justify implementation difference between Single Sign On and SAML (Security Assertion Markup Language) models of cloud authorization.

SAME AS Q9

15. How can a Single Sign On be utilized to enable all service access using single authentication?

Single Sign-On (SSO) allows users to access multiple services using a single set of credentials. To enable this, a centralized authentication system is created, which authenticates users and generates an authentication token. The services are then configured to recognize this token using protocols such as SAML, OAuth, or OpenID Connect. This makes it easier for users to access different services without having to log in multiple times, thus reducing the burden of remembering multiple credentials. SSO also improves security by reducing the number of attack surfaces and enhances the user experience by making it more seamless to access different services.

16. What is SAML? What is it composed of? Why is it more advantageous than SSO?

Security Assertion Markup Language (SAML) is an XML-based open standard used for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider. The protocol uses the web infrastructure where XML data moves over HTTP protocols on TCP/IP networks.

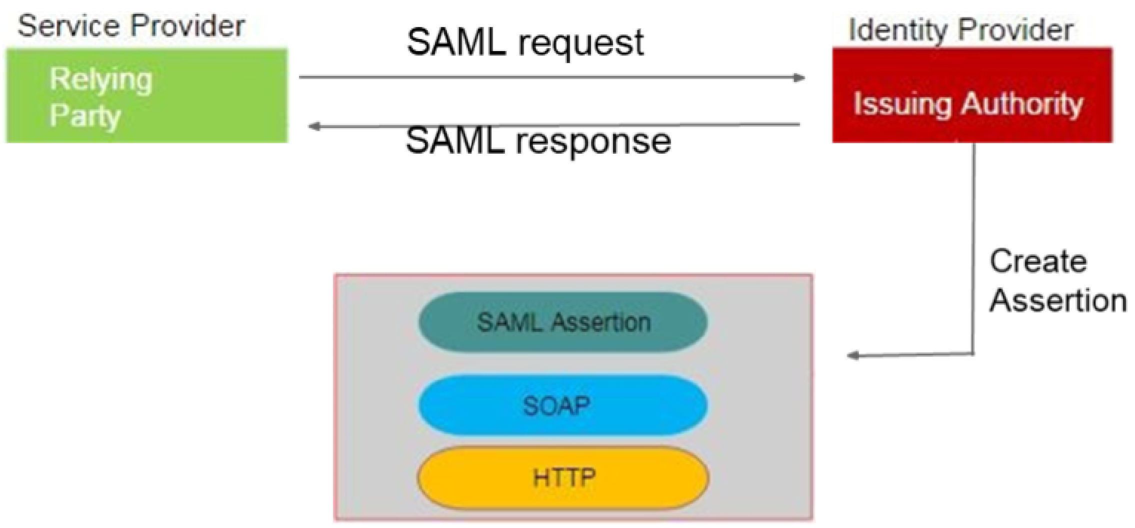
SAML is composed of:

- Assertions
- Request/response protocols
- Bindings (the SOAP-over-HTTP method of transporting SAML requests and responses)
- Profiles (for embedding and extracting SAML assertions in a framework or protocol)

Why SAML?

- Cookies don't do it –
  - Cookie (signed with server's private key) can be used for re-authentication at a particular server, but is of no use at a different server
- Cross domain authentication currently requires proprietary SSO software
- SAML intended as a Web standard that will supercede proprietary software

17. Explain how SAML works with the help of a diagram.

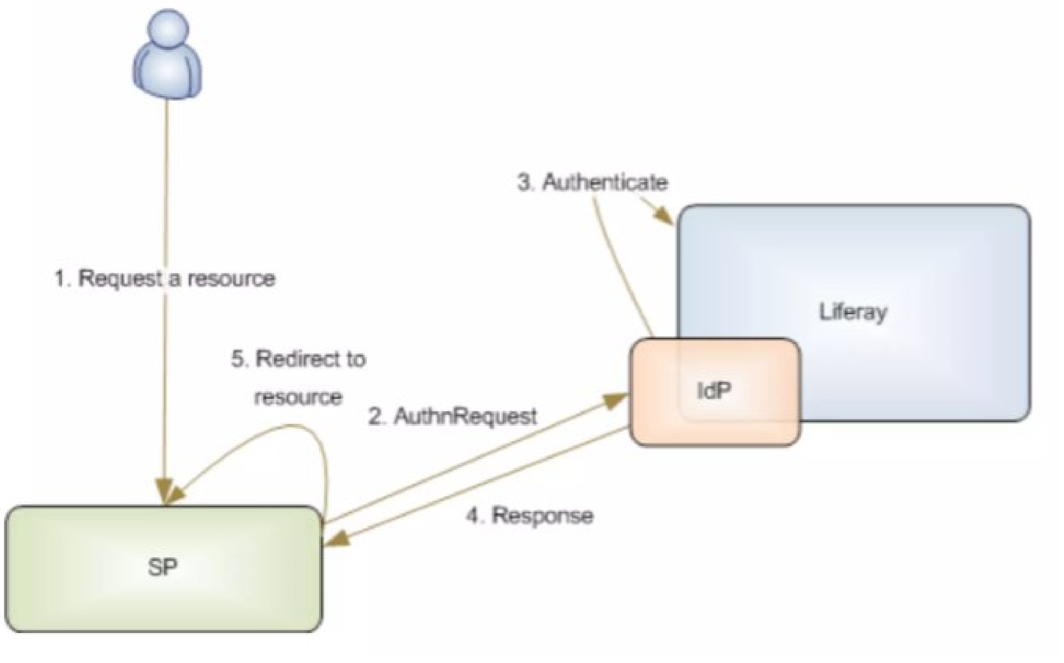


1. The user tries to access a service provided by the Service Provider (SP).
2. Since the user is not yet authenticated, the SP sends a SAML request to the IdP, which is responsible for authenticating the user.
3. The SAML request contains information about the SP's identity and the requested service, among other things.
4. The IdP receives the SAML request and initiates the user authentication process. This typically involves prompting the user to enter their credentials and verifying them against the IdP's user directory or database.
5. If the user is authenticated successfully, the IdP generates a SAML response that contains an assertion. The assertion confirms the user's identity and attributes, such as their name, email, and other relevant data.
6. The SAML response is digitally signed by the IdP to ensure its authenticity.
7. The IdP sends the SAML response back to the SP.
8. The SP receives the SAML response and verifies its signature to ensure its authenticity.

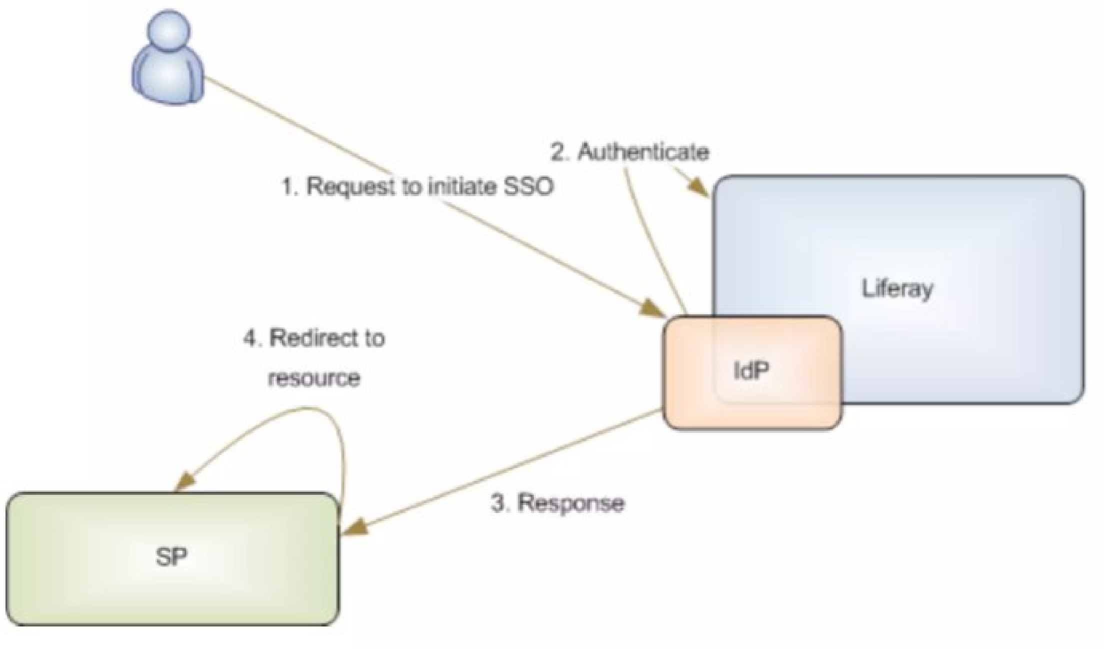
9. The SP grants the user access to the requested service.

18. Draw the diagrams for

a. SP initiated SSO



b. IdP initiated SSO



19. Explain in brief about

Assertion

A set of statements (claims) made by a SAML authority (asserting party)

- **Authentication statement**: subject was authenticated using a particular technique at a particular time
- **Attribute statement**: particular attribute values are associated with the subject
- **Authorization decision statement**: subject is authorized to perform certain actions

Subject

- Identifies the entity to which the assertion pertains
- Identifies confirmation method and (optionally) confirmation data
- If the relying party performs the specified authentication method (perhaps using the data) then it can treat the entity presenting the assertion as the entity that the SAML authority associates with the name identifier
- Example: method = public key, data = key information

Authentication Statement

Asserts that the enclosing assertions' subject was authenticated by a particular means at a particular time

- Authentication itself is *not* part of SAML
- Statement refers to an authentication act that took place at a prior time

```
<saml:AuthenticationStatement
          AuthenticationMethod="password"
          AuthenticationInstant="…." />
```

Attribute Statement

Asserts that the enclosing assertion’s subject is associated with attribute *attrib* with value *val*.

Example: the value of the attribute *Department* associated with the assertion’s subject is *Accounting*

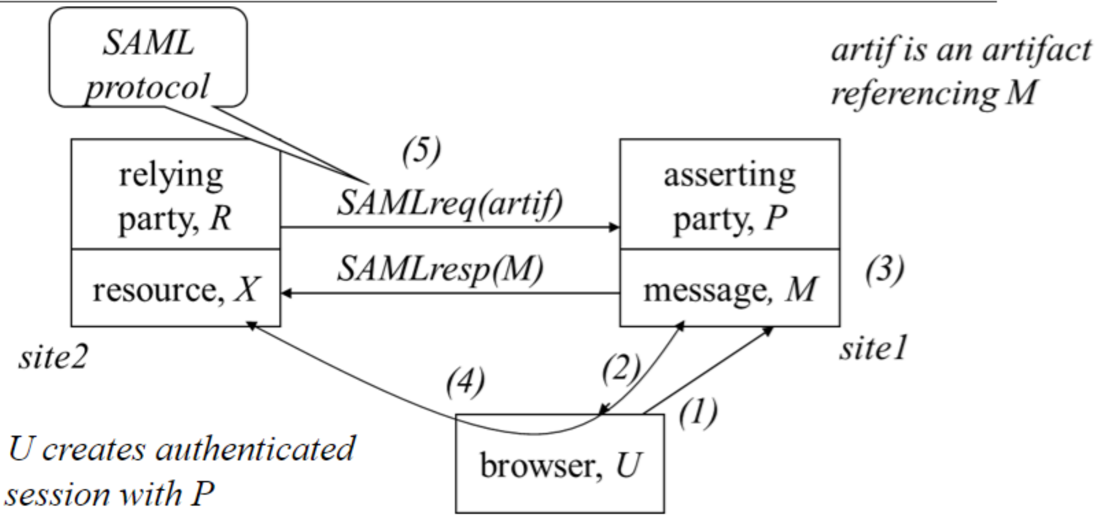
```
<saml:AttributeStatement>
   <saml:Attribute  Name =“attrib”>
      <saml:AttributeValue> val </saml:AttributeValue>
   </saml:Attribute>
</saml:AttributeStatement>
```

Authorization Decision Statement

Asserts that the enclosing assertion’s subject’s request for a particular action at the specified resource has resulted in the specified decision

```
<saml:AuthorizationDecisionStatement  Decision=“permit”
                                    Resource=“… some URI … >
   <saml:Action> Execute </saml:Action>
</saml:AuthorizationDecisionStatement>
```

Artifact – Pull Model



1. *U creates authenticated session with P*
2. *U requests access to X (through P).*
3. *P creates protocol msg, M, containing assertion about U, and an artifact referring to M*
4. *Access, containing artifact, is redirected from P to R through browser*
5. *R pulls M (identified by artifact) from P*

20. What are AWS S3, AWS EBS and AWS EFS? What is the difference between them?

AWS S3 (Simple Storage Service) is a scalable and highly durable object storage service that allows users to store and retrieve data from anywhere on the web. It is designed for storing large amounts of unstructured data, such as images, videos, and backups. S3 is accessed via HTTP/HTTPS and supports features such as versioning, encryption, and lifecycle policies. S3 is ideal for storing and retrieving large files and provides a cost-effective option for data archiving and backup.

AWS EBS (Elastic Block Store) is a block-level storage service designed for use with EC2 (Elastic Compute Cloud) instances. It provides persistent block storage volumes that can be attached to EC2 instances as virtual hard drives. EBS volumes are replicated within their Availability Zone to ensure high availability and durability. They support features such as encryption, snapshots, and point-in-time recovery. EBS is ideal for storing application data that requires low-latency access and frequent updates.

Amazon EFS (Amazon Elastic File System) provides scalable network file storage for Amazon EC2 cloud computing service users. Amazon EFS is automatically scalable -

which means your running applications won't have any problems if the workload suddenly increases - storage will scale accordingly. If the workload decreases - the amount of storage will be reduced, so you won't pay for the unused storage. EFS is ideal for applications that require shared access to data, such as content management systems, web serving, and big data analytics.

21. Explain the features of Amazon S3.

Lifecycle Management: In lifecycle management, Amazon S3 applies a set of rules that define actions to a group of objects. Life cycle management normally refers to moving the data from one storage tier to another storage tier. Then finally expiring it and completing the life cycle of an object

Bucket Policy: Bucket policy is an IAM policy where you can allow and deny permission to your Amazon S3 resources. With bucket policy, you also define security rules that apply to more than one file within a bucket
For example: If you do not want a user to access the "Simplilearn" bucket, then with the help of JSON script you can set permissions

Data Protection: Amazon S3 provides IT teams a highly durable, protected and scalable infrastructure designed for object storage.
Amazon S3 protects your data using 2 methods: Data Encryption and Versioning

Data Encryption: It refers to protection of data while it's being transmitted and at rest. Data Encryption can happen in two ways:

- Client-Side Encryption - Data encryption at rest
- Server-Side Encryption - Data encryption in motion

Versioning: It can be utilized to preserve, recover and restore early versions of every object you store in your Amazon S3 bucket. Unintentional erase or overwriting of objects can be easily regained with versioning

Cross-Region Replication: Cross-Region Replication provides automatic copying of every object uploaded to your buckets (source bucket and destination bucket) in different AWS regions

Transfer Acceleration: It enables fast, easy and secure transfers of files over long distances between your client and S3 bucket. The edge locations around the world provided by Amazon CloudFront are taken advantage by transfer acceleration. It works via carrying data over an optimized network bridge that keeps running between the AWS Edge Location (closest region to your clients) and your Amazon S3 bucket

22. What is Bigtable? Explain its basic data model.

Bigtable is a distributed storage system developed by Google for managing structured data. It is a NoSQL database that is designed to handle large amounts of data that can be organized into tables. Bigtable is used internally by Google for various applications such as Google Search, Google Maps, and Gmail.

The basic data model of Bigtable is based on a sparse, distributed, persistent multidimensional sorted map. The data is organized into tables, which are composed of rows and columns. Each table has a unique name, and the rows are identified by a row key. The columns are further organized into column families, which are identified by name.

Each row in a Big Table can have multiple column families, and each column family can have multiple columns. The data in each column is stored as a byte array, and the values can be of any data type. The columns are sorted lexicographically, so they can be efficiently accessed using range scans.

Bigtable provides strong consistency for read and writes operations within a single row, but eventual consistency for cross-row operations. It also supports atomic row-level operations such as read-modify-write.

Bigtable is designed to be highly scalable and fault-tolerant. It can scale up to petabytes of data and can handle millions of read and write operations per second. It achieves fault tolerance through data replication and automatic failover.

23. Explain compression in Bigtable.

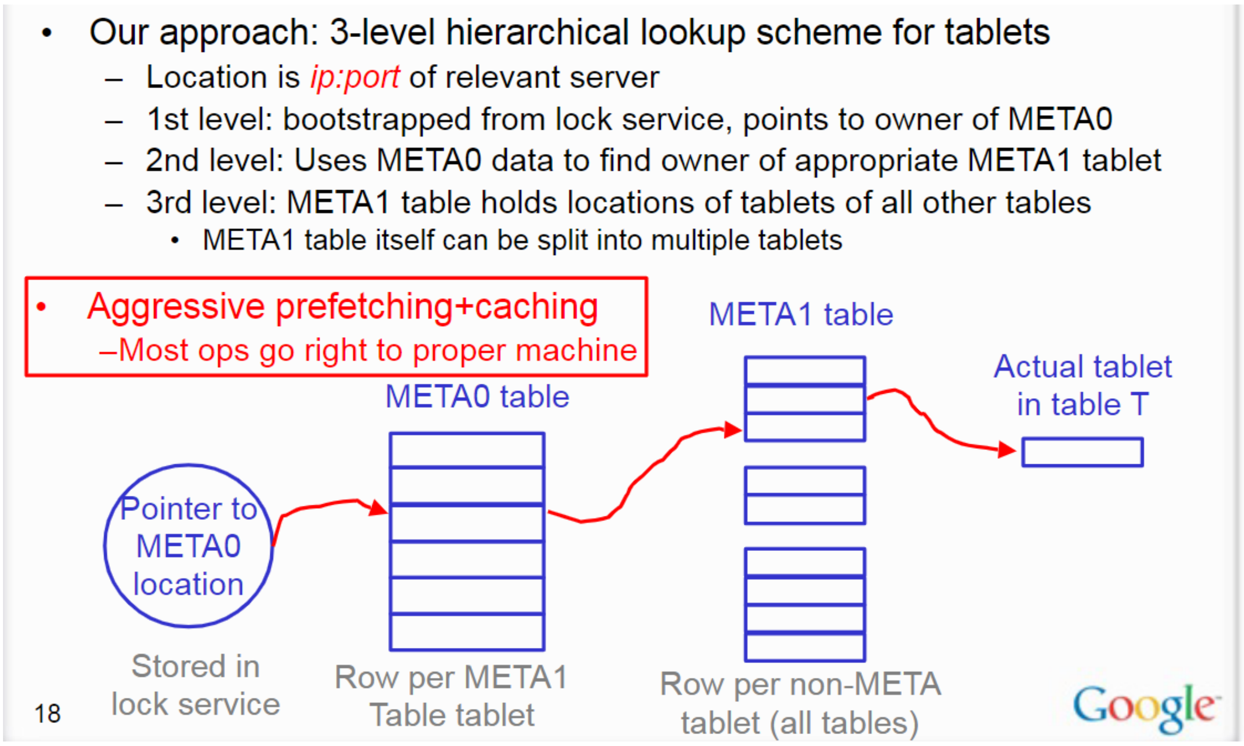
Compression is an important feature in Bigtable, as it helps to reduce the storage space required for storing data, and also improves the performance of read and write operations. In Bigtable, compression is applied at the column-family level, which means that different column families can have different compression settings.

Bigtable supports several compression algorithms, such as Snappy, Gzip, and LZO. Snappy is the default compression algorithm used in Bigtable, as it provides a good balance between compression ratio and performance. Gzip provides higher compression ratios but at the cost of higher CPU usage. LZO provides lower compression ratios but with very low CPU usage.

When data is written to a Big Table, it is first serialized into a binary format and then compressed using the specified compression algorithm. During read operations, the compressed data is first decompressed, and then deserialized back into its original format. This process is transparent to the user, and does not require any additional configuration.

Compression can be configured on a per-column-family basis, using the compression algorithm of choice. Column families can be created with different compression settings, based on the type of data they store, or the performance requirements of the application.

24. Analyse functioning of a 3-level hierarchical lookup scheme for tablets.



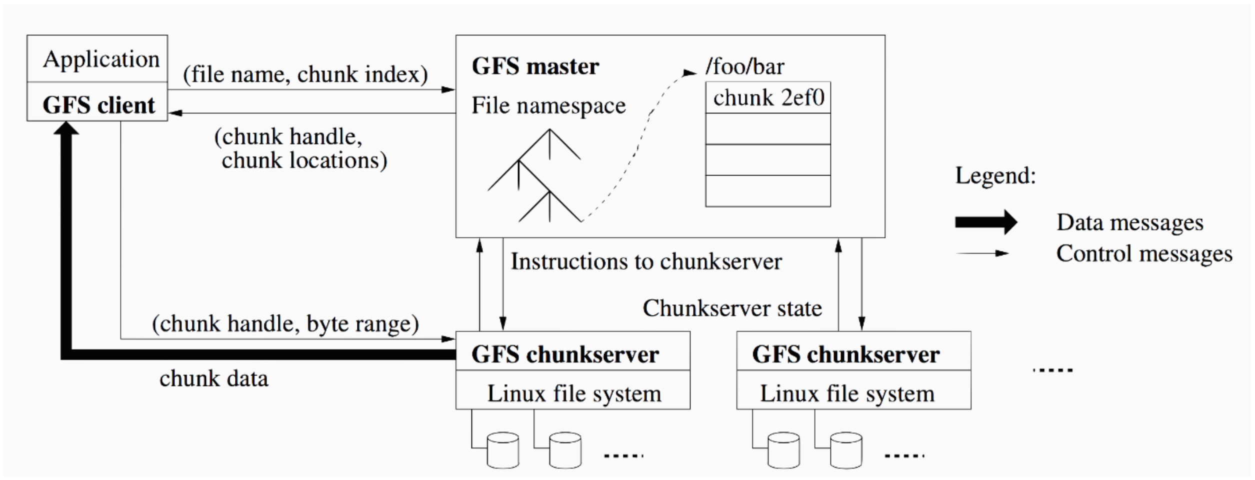
- Our approach: 3-level hierarchical lookup scheme for tablets
  - Location is *ip:port* of relevant server
  - 1st level: bootstrapped from lock service, points to owner of META0
  - 2nd level: Uses META0 data to find owner of appropriate META1 tablet
  - 3rd level: META1 table holds locations of tablets of all other tables
    - META1 table itself can be split into multiple tablets
- Aggressive prefetching+caching
  - Most ops go right to proper machine

Pointer to META0 location
Stored in lock service
META0 table
Row per META1 Table tablet
META1 table
Row per non-META tablet (all tables)
Actual tablet in table T
18

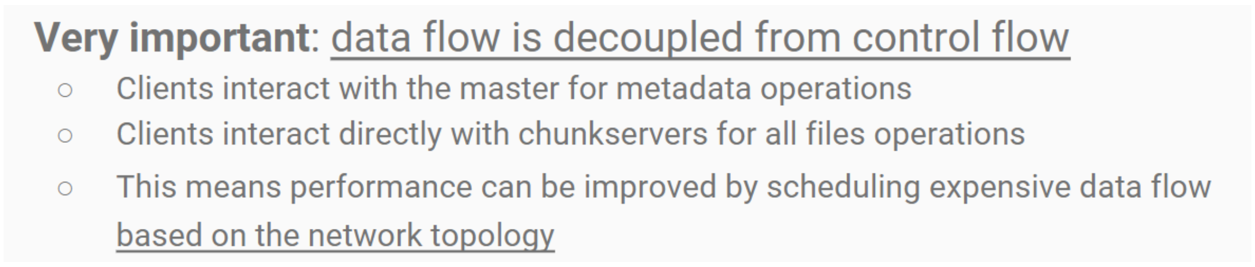
25. Write a note on GFS.

GFS is a distributed file system developed by Google to store and manage large amounts of data across commodity hardware. Its design is based on three concepts: fault tolerance, scalability, and performance. To achieve fault tolerance, GFS stores data in multiple replicas, distributed across different nodes in the cluster. It uses a master-slave architecture to control metadata operations, while the data is stored in data nodes. The system allows for dynamic expansion or reduction of the cluster without disruption to ongoing operations, and it provides a single global namespace for accessing files, simplifying the management of large-scale data. GFS optimizes performance for large, sequential reads and writes, typical of data-intensive applications, using techniques such as data partitioning, distributed caching, and pipelining of data transfers. It has been used extensively by Google for applications such as Google Search, Google Maps, and YouTube. Its design has influenced the development of other distributed file systems, such as Hadoop Distributed File System (HDFS).

26. Justify using Google File System architecture. “Decoupling of data flow from control flow helps to improve performance”.



1. There is a single master in the whole cluster which stores metadata.
2. Other nodes act as chunk servers for storing data.
3. The file system namespace and locking facilities are managed by the master.
4. The master periodically communicates with the chunk servers to collect management information and give instructions to chunk servers to do work such as load balancing or fail recovery.
5. With a single master, many complicated distributed algorithms can be avoided and the design of the system can be simplified.
6. The single GFS master could be the performance bottleneck and single point of failure.
7. To mitigate this, Google uses a shadow master to replicate all the data on the master and the design guarantees that all data operations are transferred between the master and the clients and they can be cached for future use.
8. With the current quality of commodity servers, the single master can handle a cluster of more than 1000 nodes.


Very important: data flow is decoupled from control flow

- Clients interact with the master for metadata operations
- Clients interact directly with chunkservers for all files operations
- This means performance can be improved by scheduling expensive data flow based on the network topology

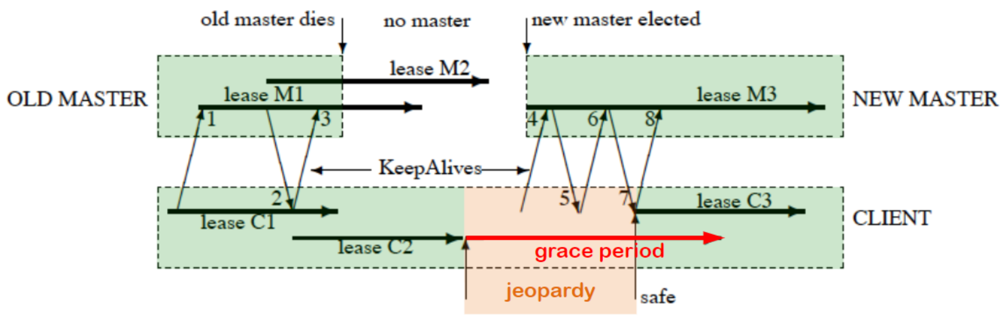
27. Explain the Chubby lock service.

- **Chubby lock service is intended for use within a loosely-coupled distributed system consisting large number of machines (10.000) connected by a high-speed network**
  - Provides coarse-grained locking
  - And reliable (low-volume) storage
- **Chubby provides an interface much like a distributed file system with advisory locks**
  - Whole file read and writes operation (no seek)
  - Advisory locks
  - Notification of various events such as file modification
- **Design emphasis**
  - **Availability**
  - **Reliability**
  - **But not for high performance / high throughput**
- **Chubby uses asynchronous consensus: PAXOS with lease timers to ensure liveness**

28. With appropriate example elaborate master fail-over scenario handled by chubby.

Fail-overs

- **When a master fails or otherwise loses mastership**
  - Discards its in-memory state about sessions, handles, and locks
  - Session lease timer is stopped
  - **If a master election occurs quickly, clients can contact the new master before their local lease timer expire**
  - **If the election takes a long time, client flush their caches (= JEOPARDY) and wait for the GRACE PERIOD (45 seconds) while trying to find the new master**



29. What is Amazon EC2? What are the steps involved in launching an Amazon EC2 instance?

Amazon EC2 overview

Amazon EC2

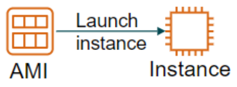
- **Amazon Elastic Compute Cloud (Amazon EC2)**
  - Provides virtual machines—referred to as EC2 instances—in the cloud.
  - Gives you *full control* over the guest operating system (Windows or Linux) on each instance.
- You can launch instances of any size into an Availability Zone anywhere in the world.
  - Launch instances from **Amazon Machine Images (AMIs)**.
  - Launch instances with a few clicks or a line of code, and they are ready in minutes.
- You can control traffic to and from instances.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1. Select an AMI

Choices made using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**



- Amazon Machine Image (AMI)
  - Is a template that is used to create an EC2 instance (which is a **virtual machine, or VM**, that runs in the AWS Cloud)
  - Contains a **Windows** or **Linux** operating system
  - Often also has some **software** pre-installed
- AMI choices:
  - Quick Start – *Linux and Windows AMIs that are provided by AWS*
  - My AMIs – *Any AMIs that you created*
  - AWS Marketplace – *Pre-configured templates from third parties*
  - Community AMIs – *AMIs shared by others; use at your own risk*

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2. Select an instance type

Choices made using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- Consider your use case
  - How will the EC2 instance you create be used?
- The **instance type** that you choose determines –
  - Memory (RAM)
  - Processing power (CPU)
  - Disk space and disk type (Storage)
  - Network performance
- Instance type categories –
  - General purpose
  - Compute optimized
  - Memory optimized
  - Storage optimized
  - Accelerated computing
- Instance types offer *family*, *generation*, and *size*

aws

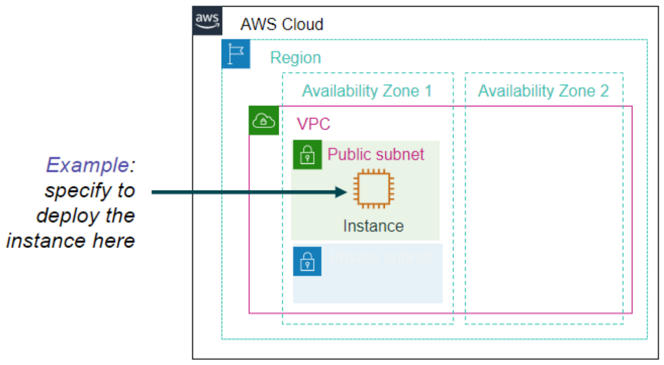
3. Specify network settings

Choices made by using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- Where should the instance be deployed?
  - Identify the **VPC** and optionally the **subnet**
- Should a **public IP address** be automatically assigned?
  - To make it internet-accessible

Example: specify to deploy the instance here



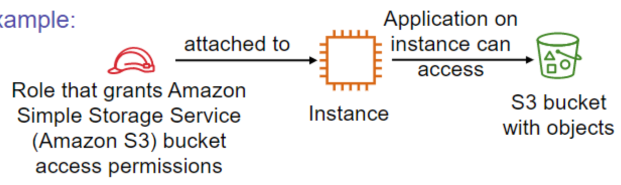
4. Attach IAM role (optional)

Choices made by using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- Will software on the EC2 instance need to interact with other AWS services?
  - If yes, attach an appropriate **IAM Role**.
- An AWS Identity and Access Management (IAM) role that is attached to an EC2 instance is kept in an **instance profile**.
- You are *not* restricted to attaching a role only at instance launch.
  - You can also attach a role to an instance that already exists.

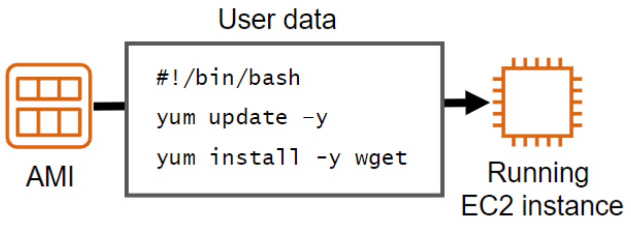
Example:


attached to
Role that grants Amazon Simple Storage Service (Amazon S3) bucket access permissions
Instance
Application on instance can access
S3 bucket with objects

5. User data script (optional)

Choices made by using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**



```
#!/bin/bash
yum update -y
yum install -y wget
```

- Optionally specify a user data script at instance launch
- Use **user data** scripts to customize the runtime environment of your instance
  - Script runs the first time the instance starts
- Can be used strategically
  - For example, reduce the number of custom AMIs that you build and maintain

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6. Specify storage

Choices made by using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- Configure the root volume
  - Where the guest operating system is installed
- Attach additional storage volumes (optional)
  - AMI might already include more than one volume
- For each volume, specify:
  - The **size** of the disk (in GB)
  - The **volume type**
    - Different types of solid state drives (SSDs) and hard disk drives (HDDs) are available
  - If the volume will be deleted when the instance is terminated
  - If **encryption** should be used

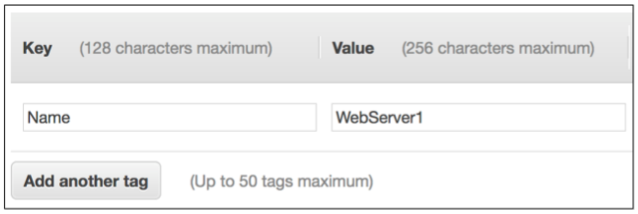
7. Add tags

Choices made by using the Launch Instance Wizard:

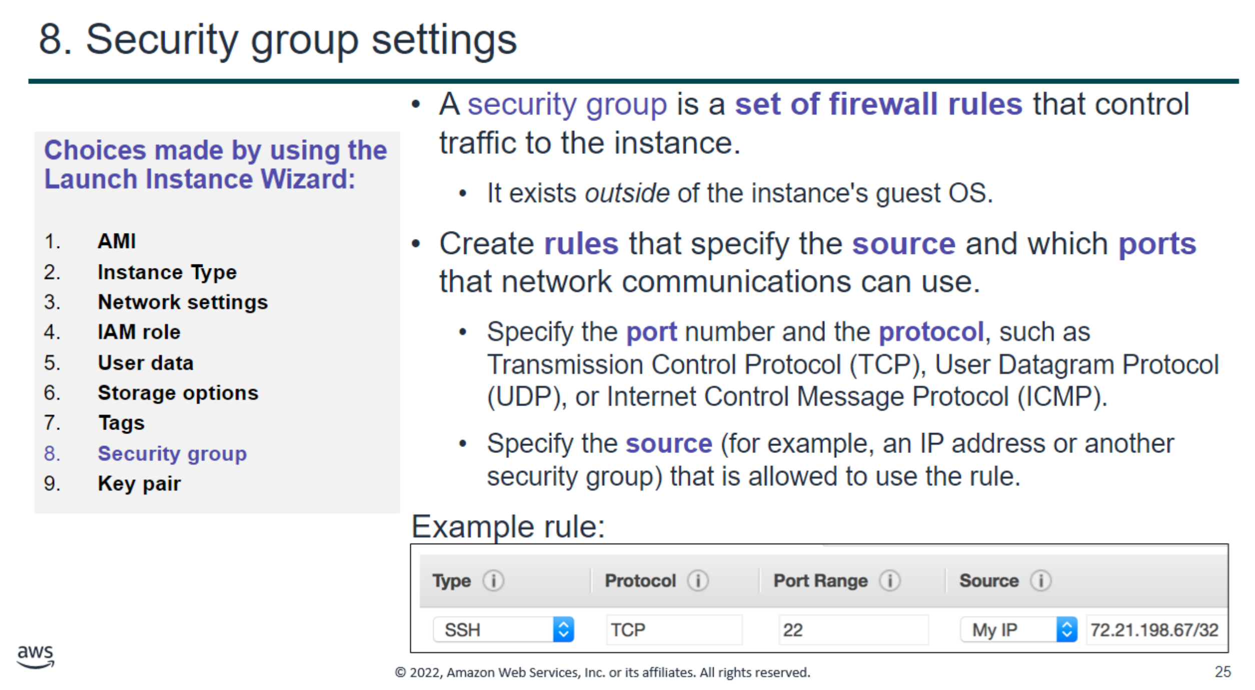
1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- A tag is a label that you can assign to an AWS resource.
  - Consists of a *key* and an optional *value.*
- Tagging is how you can attach **metadata** to an EC2 instance.
- Potential benefits of tagging—Filtering, automation, cost allocation, and access control.

Example:



Key (128 characters maximum)	**Value** (256 characters maximum)
Name	WebServer1



8. Security group settings

Choices made by using the Launch Instance Wizard:

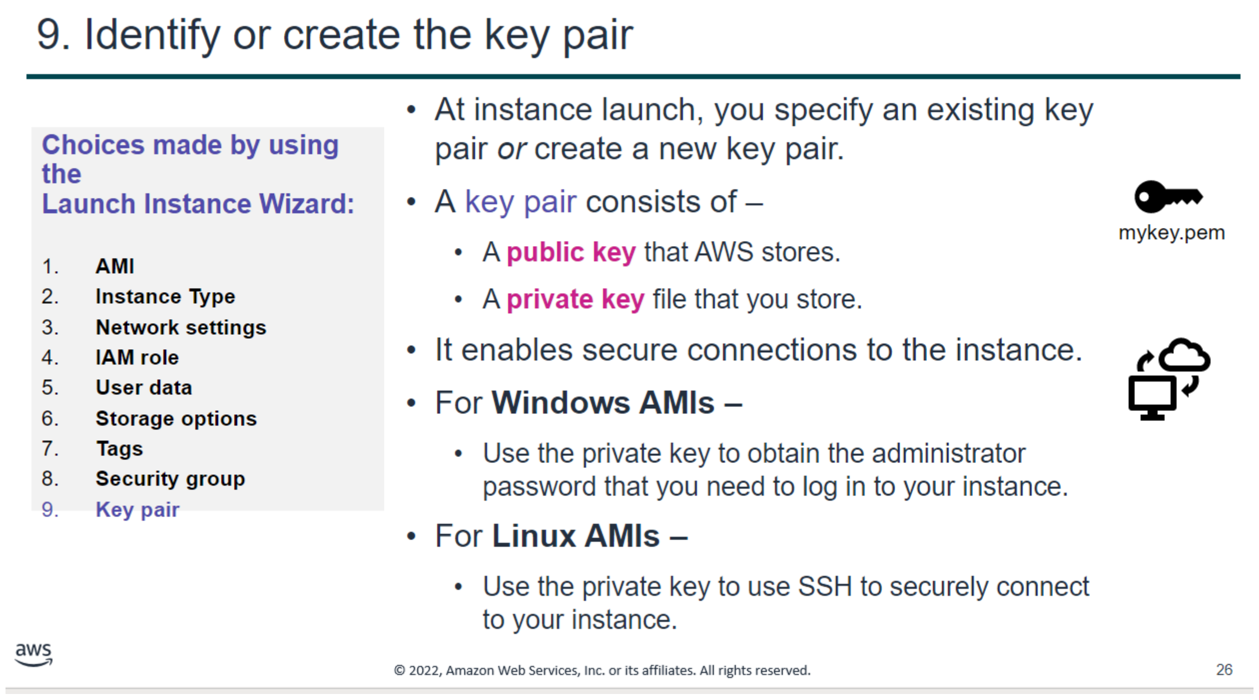
1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- A security group is a **set of firewall rules** that control traffic to the instance.
  - It exists *outside* of the instance's guest OS.
- Create **rules** that specify the **source** and which **ports** that network communications can use.
  - Specify the **port** number and the **protocol**, such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), or Internet Control Message Protocol (ICMP).
  - Specify the **source** (for example, an IP address or another security group) that is allowed to use the rule.

Example rule:

Type	Protocol	Port Range	Source	
SSH	TCP	22	My IP	72.21.198.67/32

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. 25



9. Identify or create the key pair

Choices made by using the Launch Instance Wizard:

1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**

- At instance launch, you specify an existing key pair *or* create a new key pair.
- A key pair consists of –
  - A **public key** that AWS stores.
  - A **private key** file that you store.
- It enables secure connections to the instance.
- For **Windows AMIs** –
  - Use the private key to obtain the administrator password that you need to log in to your instance.
- For **Linux AMIs** –
  - Use the private key to use SSH to securely connect to your instance.

mykey.pem

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. 26

30. What are the four pillars of cost optimization? Explain in brief about them.

Pillar 1: Right Size:

➔ Provision instances to match the need

        - CPU, memory, storage, and network throughput
        - Select appropriate instance types for your use
- Use Amazon CloudWatch metrics
    - How idle are instances? When?
    - Downsize instances
- Best practice: Right size, then reserve

Pillar 2: Increase elasticity:

- Stop or hibernate Amazon EBS-backed instances that are not actively in use
    - Example: non-production development or test instances
- Use automatic scaling to match needs based on usage
    - Automated and time-based elasticity .

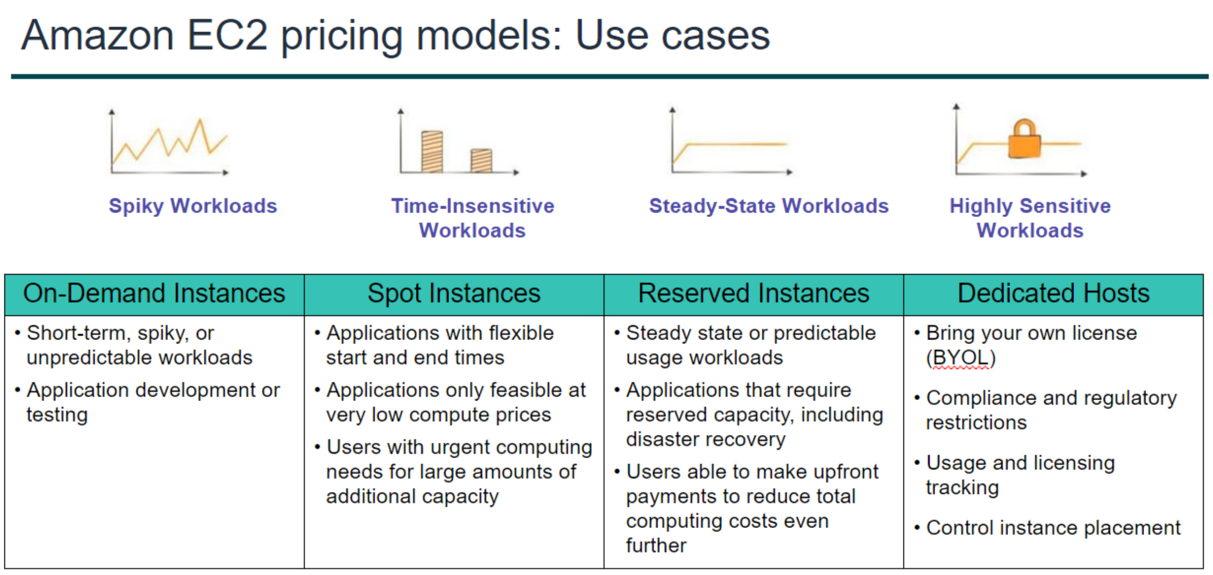
Pillar 3: Optimal Pricing Model

- Leverage the right pricing model for your use case
    - Consider your usage patterns
- Optimize and combine purchase types
- Examples:
    - Use On-Demand Instance and Spot Instances for variable workloads
    - Use Reserved Instances for predictable workloads
- Consider serverless solutions (AWS Lambda)

Pillar 4: Optimize storage choices

- Reduce costs while maintaining storage performance and availability
- Resize EBS volumes
- Change EBS volume types
    - Can you meet performance requirements with less expensive storage?
    - Example: Amazon EBS Throughput Optimized HDD (st1) storage typically costs half as much as the default General Purpose SSD (gp2) storage option.
- Delete EBS snapshots that are no longer needed
- Identify the most appropriate destination for specific types of Data
    - Does the application need the instance to reside on Amazon EBS?
    - Amazon S3 storage options with lifecycle policies can reduce costs

31. Explain the use cases of Amazon EC2 pricing models.



Amazon EC2 pricing models: Use cases

Spiky Workloads — Time-Insensitive Workloads — Steady-State Workloads — Highly Sensitive Workloads

On-Demand Instances	Spot Instances	Reserved Instances	Dedicated Hosts
• Short-term, spiky, or unpredictable workloads • Application development or testing	• Applications with flexible start and end times • Applications only feasible at very low compute prices • Users with urgent computing needs for large amounts of additional capacity	• Steady state or predictable usage workloads • Applications that require reserved capacity, including disaster recovery • Users able to make upfront payments to reduce total computing costs even further	• Bring your own license (BYOL) • Compliance and regulatory restrictions • Usage and licensing tracking • Control instance placement

32. Give a comparative analysis between block and object storage with a suitable example of its application area.

Block Storage:

Block storage is a type of storage system that stores data in fixed-size blocks. Each block is assigned a unique identifier, and data can be read from or written to individual blocks. Block storage is commonly used in applications that require high performance and low latency, such as databases, file systems, and virtual machines.

Example of application area: One example of block storage is Amazon Elastic Block Store (EBS), which is a block storage service offered by Amazon Web Services (AWS). EBS is commonly used for storing data for virtual machines running on AWS EC2 instances. EBS provides consistent low-latency performance and can be easily attached and detached from EC2 instances.

Object Storage:

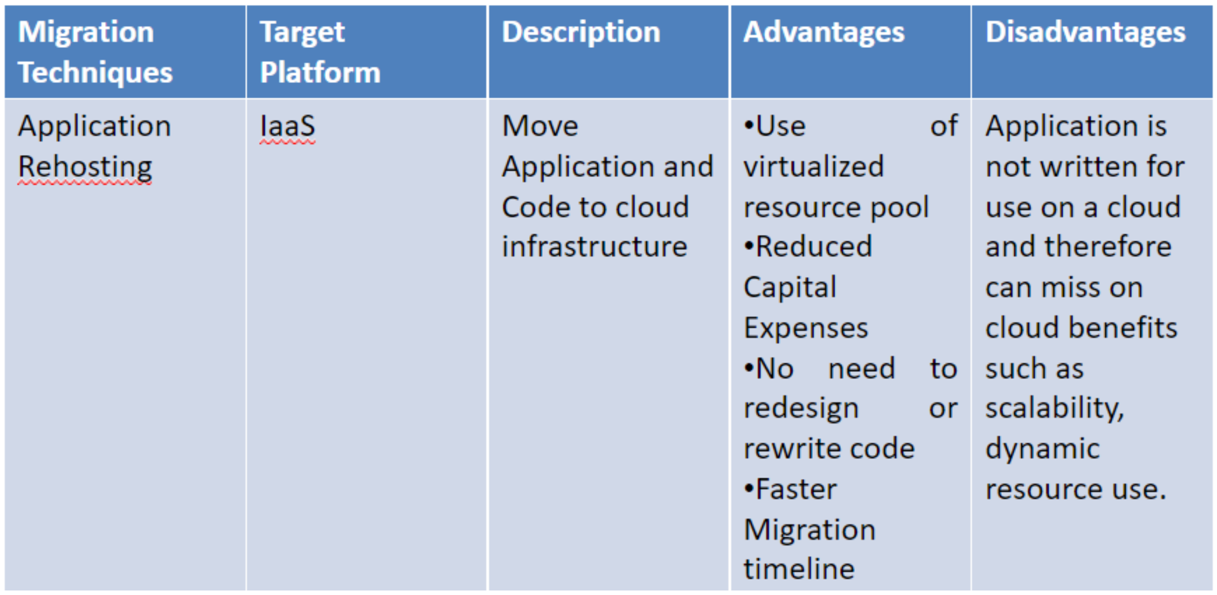
Object storage is a type of storage system that stores data as objects. Each object consists of data, metadata, and a unique identifier. Object storage is designed to store and retrieve large volumes of unstructured data, such as images, videos, and log files. Object storage is commonly used in applications that require scalable, durable, and cost-effective storage.

Example of application area: One example of object storage is Amazon Simple Storage Service (S3), which is an object storage service offered by AWS. S3 is commonly used for storing and retrieving large volumes of data, such as images, videos, and log files. S3 provides high durability and availability and can be easily integrated with other AWS services.

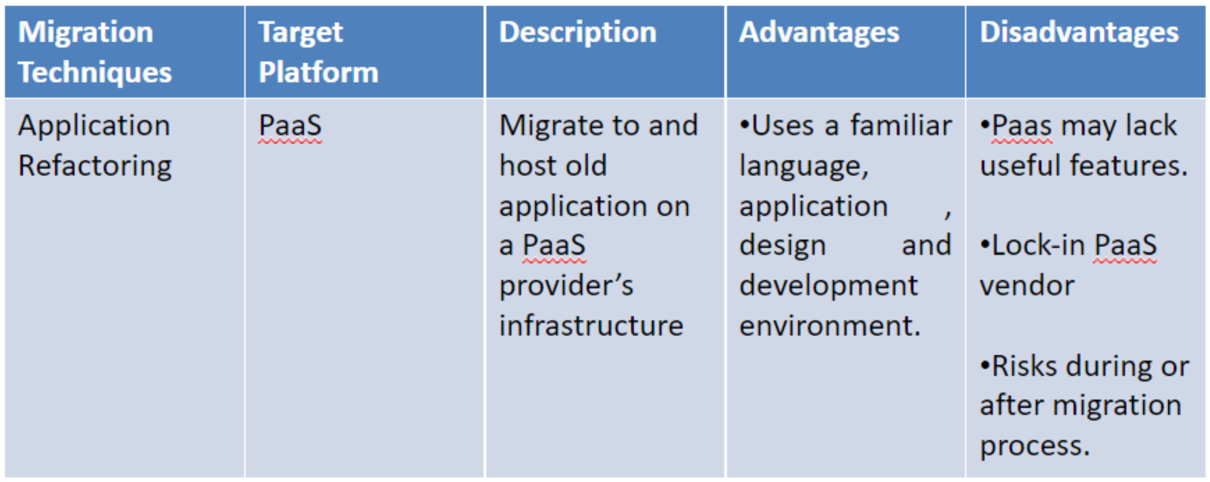
33. What are the key aspects that will migrate the users to cloud applications?

- Technical and business advantages in cloud
- Criticality of application that is being moved to cloud
- Elasticity in terms of resource scheduling support offered by cloud provider
- Technical factors such as cloud infrastructure
- Governance in terms of security, legal and compliance requirements

34. Write in brief about some cloud migration techniques.



Migration Techniques	Target Platform	Description	Advantages	Disadvantages
Application Rehosting	IaaS	Move Application and Code to cloud infrastructure	•Use of virtualized resource pool •Reduced Capital Expenses •No need to redesign or rewrite code •Faster Migration timeline	Application is not written for use on a cloud and therefore can miss on cloud benefits such as scalability, dynamic resource use.



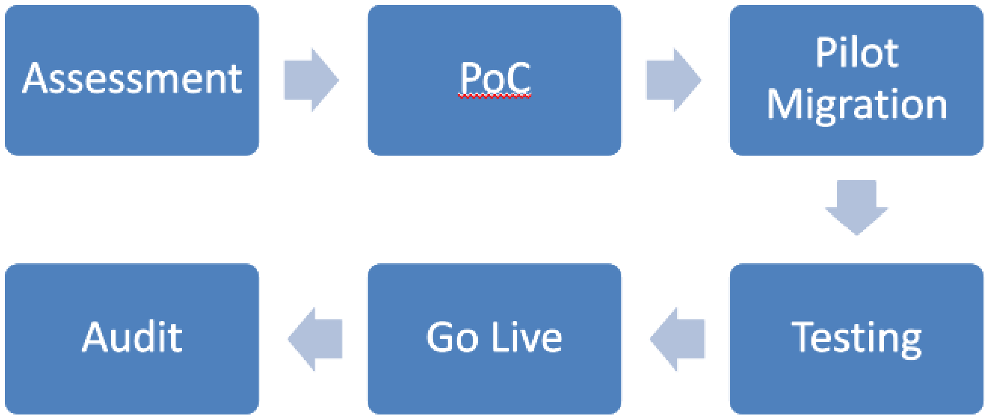
Migration Techniques	Target Platform	Description	Advantages	Disadvantages
Application Refactoring	PaaS	Migrate to and host old application on a PaaS provider's infrastructure	•Uses a familiar language, application , design and development environment.	•Paas may lack useful features. •Lock-in PaaS vendor •Risks during or after migration process.

Migration Techniques	Target Platform	Description	Advantages	Disadvantages
Application Revision and Optimization	IaaS or PaaS	Modify existing application code to make it suitable for public or private cloud, then re-host it for new deployment	•Better Performance.	•Lots of development time and Manpower

Migration Techniques	Target Platform	Description	Advantages	Disadvantages
Application Re-Architecture and re-building	PaaS	•Discard old code and application •Deign a new architecture •Develop a new application •Test and Go Live	•Improved Scalability by using SOA •Improved Modularity for easier module-based upgrades.	•Requires lots of investment in time and manpower. •Lock-In

Migration Techniques	Target Platform	Description	Advantages	Disadvantages
Replacement of the application	SaaS	•Discard old application •Select a SaaS service that meets the requirement . •Copy old data for use from within the new SaaS Service	•Reduced IT investment	•New use for old IT infrastructure needs to be found. •Vendor Lock-In

35. What are the phases of migrating an application to the cloud?



The phases of migrating application to an IaaS cloud is given below:

- **Assessment:**
    - ✓ Requires review of internal and cloud provider
    - ✓ Move to cloud have technical merits and help to reduce CapEx and OpEx for enterprises.
- Some of the assessment factors for cloud provider are:
    - Data migration
    - Security
    - Compliance with regulations
    - Technical support for issues
    - Resource usage for each month and the total fee
    - Features and functionality
    - SLA for uptime and performance and penalty for SLA violations

The phases of migrating application to an IaaS cloud is given below:

- **PoC: Proof of Concept:**
  - ✓ Requires Vendor evaluation to make sure that the vendor has required functionality.
  - ✓ In this phase administrator gets a login access to run through features of cloud.
- **Pilot Migration:** a small group of users access the cloud while keeping the setup with the previous setup as well.
- **Testing:** In this phase , enterprise migrates user data to cloud and tests the application. Tests Includes: Security, data migration, service uptime, performance etc
- **Go Live:** The process of going live can be done completely at once or it can be phased.
- **Audit:** review the use of cloud environment