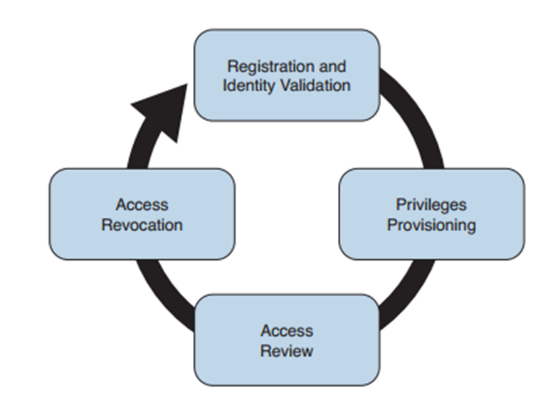
SOC Unit 1 question and answers.

1. Explain Identity and Access Lifecycle in detail?

Identity and access management (IAM) has a very broad definition and in general includes all policies, processes, and technologies used to manage the identity, authentication, and authorization of an organization’s resources.

four phases of the identity and access lifecycle

1. 

Registration and identity validation: A user provides information and registers for a digital identity. The issuer will verify the information and securely issue a unique and non descriptive identity.

■ Privileges provisioning: The resource owner authorizes the access rights to a specific account, and privileges are associated with it.

■ Access review: Access rights are constantly reviewed to avoid privilege creep.

■ Access revocation: Access to a given resource may be revoked due, for example, to account termination.

Registration and Identity Validation:-

The first step in a secure identity lifecycle is the user registration. During this phase, the user registers his data to request an identity. The second step of this process would be to verify the identity. This can be done in several ways, depending on the privileges associated with that identity.

•Privileges Provisioning:-

privileges will be assigned in accordance with the organization’s security policy.

Depending on the access control model applied, the process might need to ensure that an authorization request is sent to the resource owner and that privileges are not assigned until the access has been approved. A temporal limit should also be applied to the privileges assigned.

Access Review:-

Access rights and privileges associated with an account should be constantly reviewed to ensure that there is no violation to the organization’s security policy. The process should ensure a regular review of privileges as well as an event-driven review, such as when a user changes roles.

Access Revocation:-

When an employee changes jobs or leaves the organization, there may be a need to partially or completely revoke his associated access rights. A formal process should be established to make sure this is done properly. In some cases, privileges may need to be revoked before the actual event (for example, an involuntarily job termination) to ensure the employee does not cause damage to the organization before officially leaving.

2. Discuss Password Management and Password Creation in Detail?

Password authentication is usually considered one of the weakest authentication methods, yet it’s one of the most used due to its implementation simplicity.

The weakness of password authentication is mainly due to the human factor rather than technological issues. Here’s a list of some typical issues that lead to increased risk when using passwords as the sole authentication method:

■ Users tend to use the same password across all systems and accounts.

■ Users tend to write down passwords (for example, on a sticky note).

■ Users tend to use simple passwords (for example, their child’s name or 12345).

■ Users tend to use the default system password given at system installation.

Password management includes all processes, policies, and technologies that help an organization and its users to improve the security of their password-authentication systems.

Password management includes policies and technologies around password creation, pass- word storage, and password reset

Password Creation:-

Strength: Establishing a policy about the password strength is very important to reduce the risk of users setting up weak passwords, which are easier to compromise via bruteforce attacks, for example. Complexity and length requirements contribute to increasing the strength of a password. Complexity is usually enforced by asking the user to use a combination of characters, numbers, and symbols. Password length increases the difficulty of cracking a password. The shorter the password, the higher the risk.

Age: The age of a password (or better, the maximum age of a password) is an important attribute. Changing a password frequently is considered a best practice. The longer a password is used, the higher the risk of password compromise. The password requirement policy should dictate the maximum age of a password. Changing passwords frequently is better for security; however, it creates additional administrative overhead for users and systems.

Reusability: Reusing the same password or part of it also increases the risk of password compromise. It is common practice to change just the last digit of a password or to use only two passwords repeatedly and just swap them when the time comes. Policy around reusability should ensure that passwords are not reused within a given amount of time.

User-Generated Password: Using passwords created by the users is the easiest method but is the riskiest from a security point of view. Users tend to use easy passwords, reuse the same passwords, and, in some cases, disclose password to others. Enforcing password requirements helps reduce the risk.

System-Generated Password: Using system-generated passwords is a stronger method than using user-created passwords because the password requirements are directly enforced. In most cases, the system can create the passwords by using a random password generator, which ensures higher entropy and is usually more difficult to compromise. The drawback of this method is that these types of passwords are more difficult to remember. Users, therefore, tend to write them down, which defeats the purpose of having a secure password.

One-Time Password and Token: one-time password is a randomly generated password that can be used only once. One of the most used methods for implementing one-time password authentication is through a token device. The token device can be either a hardware device or implemented in software (soft-token), and it is basically a one-time password generator. For example, most of the authentication systems for online banking use token technologies

3. What is Kerberos. Explain in Detail?

Kerberos is one well-known authentication protocol that provides single sign-on capabilities. It was proposed by MIT and in its last version (v5) is described in RFC 4120. Here are the

main entities or objects involved in the Kerberos protocol:

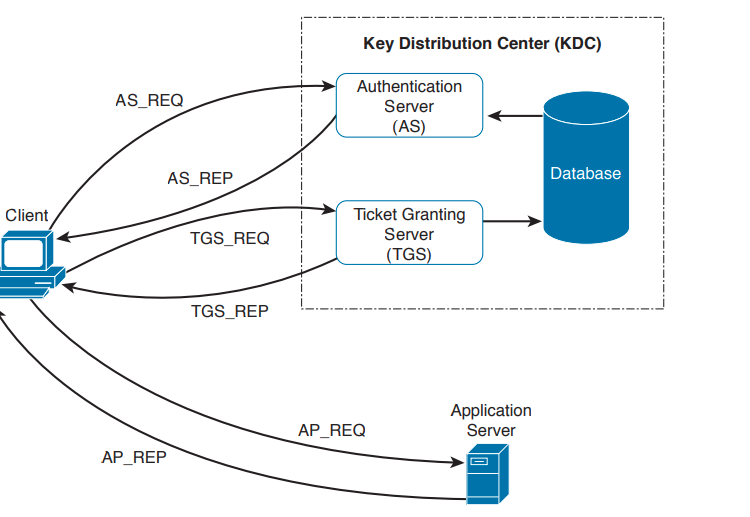
■ Key Distribution Server (KDC): The main component of a Kerberos system. It includes three components, the authentication server (AS), which provides the initial authentication ticket; the ticket-granting service (TGS), which provides ticket-granting ticket (TGT), also called the service ticket; and the Kerberos database, which includes all the information about users, hosts, servers (principals), and so on.

■ Principal: A client or server entity that participates in the Kerberos realm.

■ Ticket: A record that proves the identity of the client when authenticating to a server. This needs to be used together with an authenticator.

■ Authenticator: Further proof of identity that is used to reduce the likelihood of a replay based attack. The authenticator message includes information about the principal and a session key.

■ Realm: Identifies an authentication and authorization domain where the authentication service has authority to provide its service. Authentication of a principal can also happen outside a realm, if there is a trusted relation between realms. This is called cross-realm authentication



In its basic implementation, when a principal (for example, a user) requests access to another principal (for example, a server),

it sends a request (AS\_REQ) to the authentication server (AS) that includes its identity and the principal identifier of the server it wants to access.

The AS checks that the client and server exist in the Kerberos database, generates a session key, and creates a response (AS\_RES) that includes a ticket-granting ticket (TGT).

The client principal is ready to send a request (TGS\_REQ) to the TGS to obtain a session ticket. This request includes the TGT and the authenticator.

The TGS verifies that the principal server exists in the Kerberos database and then issues a service ticket that is then sent with its reply (TGS\_REP) to the client principal that also includes a session key.

The client principal can now request access to the server principal (AP\_REQ), which includes the service ticket and the new authenticator built based on the new session key. The server may reply with an AP\_REP that has information proving the server’s identity, if mutual authentication is required.

4. Discuss Privileges provisioning and Revocation in detail?

5. What is Single Sign-On? Explain with an Example?

The single sign-on (SSO) is that a user needs to authenticate with only one system and only once to get access to the organization’s resources. The user needs to authenticate against each of the systems but provides the same password. In an SSO system, typically the authentication is done by providing proof that the user has been authenticated. This avoids the need to input the credentials multiple times.

Although the concept is very simple, its implementation is very difficult due to the high diversity of systems usually present in a large enterprise. Effectively, organizations implementing SSO are usually implementing it only in part of the network on a subset of their systems. Additionally, SSO suffers from the same limitations as other centralized authentication systems: namely, that the authentication server can become a single point of failure and that once an account is compromised, an attacker is able to access all the systems for which that user has access rights.

Directory systems (for example, LDAP-based systems) are usually considered a type of SSO implementation. Other known implementations of SSO are Kerberos, SESAME, OpenID, and OAuth, to name a few.

The key concepts related to SSO, all of which are described in more detail

■ Single sign-on is an authentication method in which a user authenticates to an authentication server, also called an SSO server. The SSO server provides proof of authentication, which can be used to access other systems within the organization without the need to authenticate again.

■ Kerberos is a protocol used to implement SSO. It uses the notion of ticket to contain the proof of authentication.

■ Federated SSO extends the concept of SSO to multiple organizations. A user can authenticate with an SSO server within one organization, and the proof of authentication will be valid to authenticate on a system within a different organization.

■ SAML, OAuth, and OpenID Connect are known frameworks used to implement federated SSO.