CISSP

LAST MINUTE STUDY GUIDE

DOMAIN 5
IDENTITY AND ACCESS MANAGEMENT

Section 1: Introduction to IAM

Identity and Access Management (IAM) is the discipline that controls how users, devices, applications, and services gain access to resources in an organization. It is one of the most critical security domains because identity is the new security perimeter in a world where networks are distributed, cloud-driven, and mobile-first.

IAM is not just about usernames and passwords — it is a framework combining:

- **Policies** → Rules governing identity creation, management, and use.
- **Processes** → Procedures for provisioning, reviewing, and revoking identities.
- **Technologies** → Tools like directories, authentication servers, SSO, federation, and privileged access management.

1.1 Why IAM is Critical

1. **Prevents Unauthorized Access**
    - Only legitimate users should access organizational resources.
    - IAM enforces the Principle of Least Privilege (PoLP) to reduce attack surface.
2. **Supports Compliance and Auditing**
    - Regulations like GDPR, SOX, HIPAA, PCI DSS mandate strict identity control.
    - IAM provides evidence of access reviews and segregation of duties.
3. **Reduces Insider Threats and Privilege Misuse**
    - Insider attacks are often facilitated by excessive privileges.
    - IAM ensures access rights are proportional to responsibilities.
4. **Enables Business Agility**
    - With proper IAM, onboarding/offboarding of employees, contractors, and partners is quick and secure.
    - Federation allows cross-organization access without managing multiple credentials.
5. **Defends Against Credential-Based Attacks**
    - According to Verizon DBIR, over 80% of breaches involve compromised or weak credentials.

- Strong IAM strategies (MFA, adaptive authentication) mitigate these risks.

1.2 IAM and the AAA Security Framework

IAM relies on the AAA model (Authentication, Authorization, Accounting):

1. **Authentication** – Proving identity.
    - "Who are you?"
    - Implemented via passwords, biometrics, tokens, certificates, MFA.
2. **Authorization** – Determining access rights.
    - "What can you do?"
    - Managed via access control models (DAC, MAC, RBAC, ABAC).
3. **Accounting (Auditing)** – Recording activities.
    - "What did you do?"
    - Achieved through logging, monitoring, and audit trails.

Example:

- A user logs in with a smart card (Authentication).
- Based on role, they can access HR systems but not Finance systems (Authorization).
- Every action they take is logged in SIEM (Accounting).

1.3 IAM in Modern Security Context

- **Zero Trust Model** → "Never trust, always verify." IAM is central here: all requests must be authenticated and authorized regardless of location.
- **Cloud & Hybrid Environments** → IAM ensures consistent policies across on-prem, SaaS, and cloud workloads.
- **BYOD & Remote Work** → Employees access resources from personal devices → IAM enforces device posture checks, conditional access.
- **APIs and Service Accounts** → IAM governs non-human identities (applications, bots, IoT).

1.4 IAM Standards and Frameworks

- **NIST SP 800-63 (Digital Identity Guidelines)** → Defines assurance levels for identity proofing and authentication.
    - **IAL (Identity Assurance Level)** – Strength of identity proofing.
    - **AAL (Authenticator Assurance Level)** – Strength of authentication mechanism.
    - **FAL (Federation Assurance Level)** – Strength of federation assertions.
- **ISO/IEC 27001 & 27002** → IAM as part of information security controls (access control policies, least privilege, user lifecycle).
- **SOX (Sarbanes-Oxley)** → Requires separation of duties and user access reviews in financial reporting systems.
- **GDPR** → Protects personal data by limiting access to "need-to-know" identities only.

1.5 IAM Challenges

- **Password Fatigue** → Users managing too many credentials → leads to reuse and weak passwords.
- **Orphan Accounts** → Accounts left active after employees leave → risk of exploitation.
- **Privilege Creep** → Gradual accumulation of access rights due to job changes.
- **Shadow IT** → Users bypassing IAM policies by using unsanctioned apps.
- **Scalability** → IAM must handle thousands/millions of users in global organizations.

1.6 IAM Integration with Enterprise Security

IAM is not standalone — it integrates with:

- **Security Operations (SOC)** → Logs, SIEM, anomaly detection.
- **Incident Response** → Rapid deactivation of compromised accounts.
- **Network Security** → Network Access Control (802.1X, RADIUS).
- **Application Security** → Enforcing RBAC/ABAC inside apps.
- **Cloud Security** → Identity is the foundation of access to SaaS/IaaS/PaaS.

Section 2: Identity Management Concepts

Identity management (IdM) is the discipline that defines how digital identities are created, managed, maintained, and retired within an enterprise. The goal is to ensure that every identity is unique, correctly authenticated, properly authorized, and revoked when no longer needed.

2.1 What is a Digital Identity?

A **digital identity** is a set of attributes and credentials that uniquely represent an individual, device, application, or service in a system.

Components of a digital identity:

- **Identifiers** → Unique markers (e.g., username, employee ID, email).
- **Credentials** → Evidence used to prove identity (passwords, smart cards, biometrics, tokens, digital certificates).
- **Entitlements/Roles** → Permissions assigned to the identity (e.g., HR role, system admin role).
- **Attributes** → Additional details like department, job title, clearance level.

👉 **Exam Tip:** For CISSP, remember the formula:
Digital Identity = Identifier + Credentials + Attributes + Entitlements

2.2 Identity Lifecycle Management

Managing identities is a **continuous process**, not a one-time task. The **Identity Lifecycle** ensures that identities are handled securely from creation to deactivation.

Phases of the Identity Lifecycle

1. **Provisioning (Onboarding)**
    - The creation of a new account/identity when a user joins the organization.
    - Involves assigning identifiers and initial access rights based on job role.
    - Example: A new software engineer is automatically placed in the "Engineering" group in Active Directory with developer tool access.
2. **Administration / Maintenance (Ongoing Updates)**
    - Identities must be updated as users' roles change.
    - Prevents **privilege creep** (users collecting more access than they need).

- Example: When an employee moves from HR to Finance, their HR access should be revoked immediately and Finance access provisioned.

3. **Monitoring / Review**
    - Regular audits of access rights (monthly, quarterly).
    - Detects anomalies such as inactive accounts with high privileges.
    - Example: Quarterly certification review of admin accounts by managers.
4. **De-provisioning (Offboarding)**
    - Removing/locking accounts when no longer required.
    - Prevents "orphan accounts" (accounts left behind after termination).
    - Example: Contractor access revoked automatically at project completion.

👉 **Exam Tip:** A common CISSP pitfall is leaving terminated employee accounts active — always tie offboarding to immediate de-provisioning.

2.3 Identity Repositories

Organizations store and manage identities using specialized databases known as **directories**.

- **LDAP (Lightweight Directory Access Protocol)**
    - Industry-standard protocol for directory access.
    - Stores user attributes in hierarchical structures.
- **Active Directory (AD)**
    - Microsoft's directory service, widely used for enterprise IAM.
    - Provides Group Policy enforcement, Kerberos authentication, and domain management.
- **Meta-directories**
    - Aggregate data from multiple directories into a single consolidated view.
    - Useful in large enterprises with multiple domains (e.g., after mergers).
- **Virtual Directories**
    - Provide a unified "virtual" view of distributed identity sources without duplicating data.

- Example: A virtual directory allowing one login across on-prem and cloud directories.

2.4 Types of Identities

1. **Human Identities**
    - Employees, contractors, partners, customers.
    - Require authentication + authorization controls.
2. **Non-Human Identities**
    - Service accounts, APIs, bots, IoT devices.
    - Often overlooked but can be exploited if poorly managed.
    - Example: A default "admin/admin" account in a router.
3. **Privileged Identities**
    - Special accounts with elevated rights (e.g., root, domain admin).
    - Require strict controls like PAM (Privileged Access Management).

👉 **Exam Tip:** Non-human identities (service accounts, IoT) are often tested in CISSP because they are easier for attackers to exploit when not managed properly.

2.5 Identity Governance and Administration (IGA)

Identity Governance is the **policy framework** that ensures identities are managed in compliance with regulations and organizational standards.

Key IGA Processes:

- **Access Certification** → Managers review employees' access periodically.
- **Segregation of Duties (SoD)** → Prevents conflicts of interest (e.g., one user should not both create and approve financial transactions).
- **Role Engineering** → Defining standard roles (e.g., "Finance Analyst") that bundle permissions together for consistent provisioning.
- **Policy Enforcement** → Enforcing least privilege and "need-to-know" principles.

Real-world Example:

- SOX compliance requires periodic reviews of financial system access to prevent fraud.

- GDPR compliance requires minimizing access to personal data — IAM enforces this via IGA.

2.6 Challenges in Identity Management

- **Orphan Accounts** → Unused but active accounts.
- **Privilege Creep** → Users accumulating unnecessary privileges over time.
- **Shadow IT** → Users creating unsanctioned accounts outside IAM.
- **Scalability** → Managing millions of users across hybrid/multi-cloud environments.
- **Third-Party Identities** → Contractors and vendors often require temporary but secure access.

2.7 IAM and Business Alignment

Identity is not only a security control but also a business enabler.

- Simplifies employee onboarding/offboarding.
- Provides seamless customer experience via Customer IAM (CIAM).
- Supports federation to allow partner organizations to collaborate securely.

Section 3: Authentication Methods

Authentication is the process of verifying that a claimed identity is valid. It answers the critical security question:

👉 *"Can you prove you are who you say you are?"*

It is the first line of defense in IAM because without strong authentication, authorization and auditing are meaningless.

3.1 Authentication Factors

Authentication can be classified into five main factors. Remember: true multi-factor authentication (MFA) requires two or more factors from *different categories*, not just multiple passwords.

1. **Something You Know**

- Examples: Password, PIN, passphrase, security question.
- Weaknesses: Susceptible to phishing, brute-force, dictionary attacks, shoulder surfing.
- Strengths: Easy to implement, universally supported.

2. **Something You Have**
   - Examples: Smart cards, hardware tokens (RSA SecurID), OTP generators, mobile authenticator apps, SIM cards.
   - Weaknesses: Can be lost, stolen, or cloned.
   - Strengths: Provides possession-based assurance, resistant to remote attacks.
3. **Something You Are**
   - Examples: Biometric identifiers (fingerprint, iris, retina, facial recognition, voice, vein patterns).
   - Weaknesses: Privacy concerns, false acceptance/rejection, spoofing risks (e.g., fake fingerprints).
   - Strengths: Unique to the individual, difficult to share or replicate.
4. **Somewhere You Are (Location-based)**
   - Examples: GPS-based authentication, IP geolocation, cell tower triangulation.
   - Weaknesses: Can be spoofed with VPN/proxies.
   - Strengths: Useful for conditional access (e.g., blocking logins from foreign countries).
5. **Something You Do (Behavioral)**
   - Examples: Typing rhythm, mouse movements, touchscreen pressure, gait analysis.
   - Weaknesses: Still developing; not universally supported.
   - Strengths: Provides continuous authentication (not just one-time login).

👉 **Exam Tip:** Two passwords = **single-factor authentication**. A password + smart card = **multi-factor authentication**.

3.2 Password-Based Authentication

Passwords remain the most common method, despite weaknesses.

Password Best Practices:

- Minimum length (12–16 characters recommended).
- Complexity requirements (upper/lowercase, numbers, symbols).
- Avoid dictionary words.
- Avoid reuse across systems.
- Use password managers to reduce human error.

Attack Vectors:

- **Brute Force** → Trying every combination.
- **Dictionary Attack** → Using precompiled wordlists.
- **Credential Stuffing** → Using leaked username/password combos.
- **Password Spraying** → Trying a few common passwords across many accounts.

Countermeasures:

- Account lockouts or throttling login attempts.
- MFA enforcement.
- Salting + hashing passwords (bcrypt, Argon2).
- User education.

3.3 Token-Based Authentication

Tokens provide **something you have**.

- **Static Password Tokens** → Store a password for user reference (weak, outdated).
- **Synchronous Dynamic Tokens** → Generate OTPs based on time (TOTP) or event (HOTP).
- **Asynchronous Challenge-Response Tokens** → User enters response generated by token after server challenge.

Examples:

- RSA SecurID → Time-based OTP token.
- Google Authenticator / Authy → Mobile app OTP generator.
- YubiKey → Hardware-based OTP & FIDO2 authentication.

3.4 Biometric Authentication

Biometrics are **something you are**.

Common Types:

- **Fingerprint** → Widely used in phones.
- **Retina Scan** → Unique blood vessel patterns in the eye. Very accurate but intrusive.
- **Iris Scan** → Easier, less invasive than retina scans.
- **Facial Recognition** → Increasingly popular (FaceID). Risk: photos/deepfakes.
- **Voice Recognition** → Used in call centers, but can be spoofed.
- **Palm/Vein Recognition** → Newer, highly accurate, harder to forge.

Performance Metrics:

- **FAR (False Acceptance Rate)** → Probability unauthorized user is accepted.
- **FRR (False Rejection Rate)** → Probability authorized user is rejected.
- **CER (Crossover Error Rate)** → Point where FAR = FRR → lower CER = better system.

👉 **Exam Tip:** If you see a question about evaluating biometric systems, look for CER.

3.5 Multi-Factor Authentication (MFA)

MFA combines factors from at least **two different categories**.

Examples:

- Password + Smart Card.
- PIN + Fingerprint.
- OTP App + Location Verification.

Benefits:

- Stronger security against stolen credentials.
- Reduces risk of brute-force success.
- Meets compliance (PCI DSS, NIST).

Challenges:

- Usability issues.
- Cost of deployment.
- User resistance if authentication is too cumbersome.

3.6 Adaptive / Risk-Based Authentication

Adaptive authentication dynamically adjusts based on risk context.

Factors Considered:

- Device used (new vs trusted).
- Login location (normal office vs foreign country).
- Time of day (working hours vs odd hours).
- User behavior patterns.

Example:

- If a user logs in from the corporate office → only password required.
- If same user logs in from an unknown laptop in another country → require MFA (OTP, SMS, biometric).

👉 **This is essential in Zero Trust architectures and cloud security.**

3.7 Modern Authentication Protocols

- **FIDO2/WebAuthn**
  - Passwordless authentication standard.
  - Uses hardware tokens/biometrics tied to public key cryptography.
- **PKI-Based Authentication**
  - Certificates identify users/devices.
  - Used in VPNs, email encryption (S/MIME).
- **Smart Cards (CAC, PIV)**
  - Government/military standard for strong authentication.

Section 4: Multi-Factor and Adaptive Authentication

Authentication by itself (like just a password) is rarely enough in modern security. Attackers routinely steal, guess, or phish passwords. That's where multi-factor authentication (MFA) and adaptive authentication come in — adding multiple verification layers to make it exponentially harder for unauthorized users to gain access.

4.1 Multi-Factor Authentication (MFA)

Definition

MFA is the use of two or more authentication factors from different categories (knowledge, possession, inherence, location, behavior).

👉 **Key Rule:** Two passwords = NOT MFA (it's still just one factor: *something you know*).

Examples of MFA Combinations

- **Password (know) + Smart Card (have).**
- **Fingerprint (are) + OTP from mobile app (have).**
- **PIN (know) + Biometric (are) + GPS-based location (where you are).**

Benefits of MFA

1. **Defense against credential compromise**
    - Even if a password is stolen, attacker still needs token/biometric.
2. **Regulatory Compliance**
    - PCI DSS requires MFA for administrative and remote access.
    - HIPAA and SOX recommend MFA for sensitive data access.
3. **Reduced Impact of Phishing**
    - MFA stops attackers from logging in even with phished credentials.
4. **Zero Trust Integration**
    - MFA is a cornerstone of "never trust, always verify."

Challenges of MFA

- **User Resistance** → Users may complain about friction.

- **Cost** → Hardware tokens and biometric devices can be expensive.
- **Deployment Complexity** → Legacy systems may not support MFA.
- **Backup Mechanisms** → Risk if users lose their token (need recovery flows).

👉 **Exam Trap:** If a question asks for the "strongest authentication," always pick MFA with different factors, not just complex passwords.

4.2 Adaptive / Risk-Based Authentication

Definition

Adaptive authentication adjusts requirements dynamically based on **risk signals**. Instead of always demanding MFA, the system evaluates context.

Risk Signals Considered

- **Device Recognition** → Is the login from a known/trusted device?
- **Location** → Expected vs unexpected geography (office vs foreign country).
- **Time of Day** → Login attempts outside working hours may trigger extra checks.
- **Behavioral Analysis** → Deviations from normal typing speed, mouse movement, or transaction patterns.
- **Network Context** → Trusted corporate VPN vs unknown public Wi-Fi.

Example of Adaptive Authentication

- A user logs in from their usual laptop, during office hours, in their home country → Only password required.
- Same user logs in from an unknown laptop, late at night, in a different country → MFA is triggered (OTP + biometric).

This balances security with usability by applying stricter checks only when risk is higher.

4.3 MFA in Practice (Real-World Use Cases)

- **Banking** → OTP SMS + PIN + biometrics on mobile banking apps.
- **Corporate VPNs** → Smart card + PIN + certificate.
- **Cloud Access** → Password + push notification to authenticator app.

- **Government/Military** → CAC (Common Access Card) or PIV card + PIN + fingerprint.

4.4 NIST Guidelines on MFA (SP 800-63B)

NIST categorizes authentication into **Authenticator Assurance Levels (AALs):**

- **AAL1** → Single-factor (password only).
- **AAL2** → Two-factor authentication (OTP, smart card).
- **AAL3** → Multi-factor with hardware cryptographic tokens, highest assurance.

👉 For CISSP exam, remember: **AAL3 = strongest (e.g., smart card + biometric with PKI).**

4.5 Federation and MFA

- MFA often integrates with **federated identity systems** (like SAML, OAuth, OpenID Connect).
- Example: Logging into Office 365 using corporate credentials + OTP via Microsoft Authenticator.
- Ensures secure SSO while still applying MFA requirements.

4.6 Future Trends in Authentication

- **Passwordless Authentication** → Using FIDO2/WebAuthn with hardware keys or biometrics.
- **Continuous Authentication** → Monitoring user behavior throughout a session (typing patterns, location).
- **AI-Powered Adaptive Authentication** → Machine learning detects anomalies in login attempts

Section 5: Authentication Protocols

Authentication protocols define the rules and mechanisms by which systems prove the identity of users, devices, and services. They provide a standardized way for systems to exchange credentials securely across networks.

5.1 Kerberos

Kerberos is one of the most important authentication protocols for CISSP. It was developed at MIT and is used heavily in Microsoft Active Directory environments.

Core Features

- Provides mutual authentication (client ↔ server).
- Uses symmetric cryptography and a system of tickets.
- Prevents replay attacks.
- Requires time synchronization between systems (within ~5 minutes typically).

Components

1. **Key Distribution Center (KDC)** → Central authority consisting of:
    - **Authentication Server (AS)** → Verifies user credentials initially.
    - **Ticket Granting Server (TGS)** → Issues service tickets for accessing resources.
2. **Ticket Granting Ticket (TGT)** → Temporary proof of authentication.
3. **Service Ticket** → Granted by the TGS to access specific services (e.g., file server).

Kerberos Authentication Flow (Simplified)

1. User logs in → sends request to AS.
2. AS verifies credentials → issues TGT.
3. When accessing a service, user presents TGT to TGS.
4. TGS issues a Service Ticket.
5. Service Ticket is presented to the resource server → access granted.

👉 **Exam Tip:** If you see "ticket-based authentication, mutual authentication, replay protection" → it's Kerberos.

Weaknesses

- Requires synchronized clocks.
- KDC is a **single point of failure**.
- If KDC is compromised → entire system compromised.

5.2 RADIUS (Remote Authentication Dial-In User Service)

Overview

- Developed for network access (dial-up originally, now Wi-Fi/VPN).
- Provides **centralized AAA (Authentication, Authorization, Accounting)**.
- Uses **UDP** (ports 1812 = auth, 1813 = accounting).

Key Features

- Authentication: Username + password verified against central server.
- Authorization: Determines what services user can access.
- Accounting: Tracks usage (start/stop time, resources used).

Strengths

- Lightweight, widely supported.
- Scales well for ISPs, enterprise Wi-Fi, VPNs.

Weaknesses

- Uses **UDP**, less reliable than TCP.
- Encrypts only the password in packet → other attributes in cleartext.
- Less granular control vs TACACS+.

👉 **Exam Trap:** If question mentions "UDP, central AAA, VPN/Wi-Fi login" → answer is **RADIUS**.

5.3 TACACS+ (Terminal Access Controller Access Control System Plus)

Overview

- Originally by DoD, now Cisco proprietary.
- Provides **separation of AAA functions** (Authentication, Authorization, Accounting handled independently).
- Uses **TCP (port 49)** → more reliable than RADIUS.

Strengths

- Encrypts the entire packet (not just the password).
- More granular control → command-level authorization.

- Preferred in **network administration (routers, switches, firewalls)**.

Weaknesses

- Cisco proprietary → less universal adoption.

👉 **Exam Trap:** If question mentions **TCP port 49, full packet encryption, granular control** → answer is **TACACS+**.

5.4 Diameter

Overview

- Successor to RADIUS.
- Provides more robust, secure, and scalable AAA.
- Uses **TCP or SCTP**.
- Commonly used in **4G/5G mobile networks**.

Strengths

- More secure than RADIUS.
- Peer-to-peer communication model.
- Supports failover and error reporting.

Weaknesses

- More complex, heavier protocol.
- Less widely adopted compared to RADIUS/TACACS+.

5.5 Other Authentication Protocols (Exam-Relevant)

- **LDAP (Lightweight Directory Access Protocol)**
  - Used for directory access (Active Directory).
  - Not inherently secure → use **LDAPS** (with TLS).
- **CHAP (Challenge Handshake Authentication Protocol)**
  - Uses a 3-way handshake with challenge-response.
  - Protects against replay attacks, but considered weak today.
- **MS-CHAPv2**

- Microsoft's extension of CHAP.
- Used in older VPNs but now deprecated due to vulnerabilities.

- **PAP (Password Authentication Protocol)**
  - Transmits passwords in **cleartext**.
  - Weakest protocol → never use in secure environments.

5.6 Comparison Table

Protocol	Transport	Encryption	AAA Separation	Use Case	Weakness
Kerberos	TCP/UDP	Symmetric keys + tickets	N/A	Active Directory	Needs time sync, KDC SPOF
RADIUS	UDP 1812/1813	Password only	No (combined AAA)	VPNs, Wi-Fi	Partial encryption
TACACS+	TCP 49	Full packet	Yes	Network admin (routers/switches)	Cisco proprietary
Diameter	TCP/SCTP	Strong	Yes	Mobile networks (4G/5G)	Complex, heavy
PAP	TCP/IP	None	No	Legacy systems	Cleartext passwords
CHAP	PPP	Hash challenge	No	Legacy PPP	Weak, deprecated

Section 6: Single Sign-On (SSO) and Federation

6.1 Single Sign-On (SSO)

Definition:
Single Sign-On (SSO) allows a user to authenticate once and gain access to multiple systems/applications without re-entering credentials.

Example: A corporate employee logs into their workstation once and automatically gets access to email, file servers, and internal apps without entering a password again.

How SSO Works

1. User authenticates against a central identity provider (IdP).
2. IdP issues a **token/ticket** proving authentication.
3. Token is presented to other applications → granting access automatically.

Benefits of SSO

- **User Convenience** → Reduces "password fatigue."
- **Improved Security** → Fewer credentials to manage lowers reuse risk.
- **Centralized Access Control** → Easier enforcement of policies and monitoring.
- **Compliance Support** → Easier auditing since authentication is centralized.

Risks/Challenges of SSO

- **Single Point of Failure** → If SSO system is compromised, attacker gets access to all resources.
- **High-Value Target** → Attackers focus heavily on breaking into IdPs.
- **Complexity** → Requires careful trust relationships and integration.

👉 **Exam Tip:** If a question says "one login provides access to multiple systems," answer = **SSO**.

6.2 Federation

Definition:
Federation extends SSO across multiple organizations/domains. It allows a user authenticated in one domain to access resources in another without re-entering credentials.

Example: A university student logs into the campus portal and automatically gains access to online library databases hosted by external vendors (federated identity).

How Federation Works

- A trust relationship is established between organizations.
- One domain acts as the Identity Provider (IdP).
- Other domains (service providers, SPs) trust the IdP's authentication assertions.

Benefits of Federation

- Enables business-to-business (B2B) collaboration.
- Reduces need for multiple identity stores across organizations.
- Enhances user experience → seamless login across trusted domains.

Federation vs SSO

- **SSO**: Works within a single organization or domain.
- **Federation**: Extends SSO across multiple organizations/domains.

👉 **Exam Trap:** If question says "across multiple organizations" → answer is **Federation**, not just SSO.

6.3 Federation & SSO Technologies

SAML (Security Assertion Markup Language)

- XML-based standard for exchanging authentication and authorization information.
- Common in **enterprise SSO and federation**.
- Involves:
    - **IdP (Identity Provider)** → Authenticates user.
    - **SP (Service Provider)** → Provides services/resources.
    - **Assertions** → XML statements about authentication/authorization.

Use Case: Corporate SSO for SaaS apps like Salesforce, Workday.

OAuth 2.0

- An **authorization framework** (NOT authentication by itself).

- Allows a user to grant limited access to a third-party app without sharing credentials.
- Example: You allow a travel booking site to access your Google calendar.

Roles in OAuth 2.0:

- **Resource Owner** → User.
- **Client** → Application requesting access.
- **Resource Server** → Holds protected resources (e.g., Google Calendar).
- **Authorization Server** → Issues tokens.

👉 **Exam Trap:** If the question asks about "delegated authorization" → answer = **OAuth 2.0**.

OpenID Connect (OIDC)

- Built **on top of OAuth 2.0**.
- Provides **authentication + authorization** (OAuth alone only does authorization).
- Uses JSON Web Tokens (JWT) for identity assertions.

Example: Logging into a third-party website using your Google or Facebook account.

Kerberos for SSO

- Within Active Directory environments, Kerberos itself provides SSO.
- Once the user has a Ticket-Granting Ticket (TGT), they can access multiple services without re-entering credentials.

6.4 Exam-Relevant Comparisons

Technology	Purpose	Format	Common Use
SSO	Authenticate once, access multiple apps	Varies	Internal enterprise apps
Federation	Extend SSO across orgs/domains	Varies	Partner portals, edu systems
SAML	Federated authentication	XML	Enterprise SaaS apps

OAuth 2.0	Delegated authorization	Tokens	Allowing apps access to user data
OpenID Connect	Authentication + authorization	JWT (JSON)	Social logins (Google, Facebook)
Kerberos	Ticket-based SSO	Binary tickets	Active Directory environments

Section 7: Authorization and Access Control Models

7.1 What is Authorization?

Authorization answers the question:
👉 *"Now that I know who you are (authentication), what are you allowed to do?"*

It involves policies, rules, and mechanisms that control who can access which resources, under what conditions, and to what extent.

Authorization mechanisms are usually enforced by:

- **Access Control Lists (ACLs)** → Specify which subjects can access which objects.
- **Policies** → High-level rules (least privilege, separation of duties).
- **Models** → Formal approaches to structuring access (DAC, MAC, RBAC, ABAC, etc.).

7.2 Discretionary Access Control (DAC)

Definition:

- Access is at the discretion of the resource owner.
- Owner decides who can access their objects and what permissions they have.

Example:

- A file owner in Windows sets read/write permissions for other users.

Strengths:

- Flexible, user-friendly.

- Common in commercial OS (Windows, UNIX).

Weaknesses:

- Prone to Trojan horse attacks (malicious code inherits user rights).
- Users may grant excessive access, violating least privilege.

👉 **Exam Tip:** If you see "owner decides access rights" → it's DAC.

7.3 Mandatory Access Control (MAC)

Definition:

- Access decisions are enforced by the system, based on security labels (clearance + classification).
- Users cannot change permissions.

Example:

- Military environments:
  - Subject = Secret clearance.
  - Object = Confidential data.
  - Access allowed only if subject's clearance ≥ object classification.

Strengths:

- Strong security.
- Enforces centralized control.

Weaknesses:

- Inflexible, difficult to manage in dynamic business environments.

MAC Implementations:

- Bell-LaPadula Model → Focus on confidentiality ("no read up, no write down").
- Biba Model → Focus on integrity ("no read down, no write up").

👉 **Exam Tip:** If you see "clearance and classification" → it's MAC.

7.4 Role-Based Access Control (RBAC)

Definition:

- Access is based on a **user's role** in the organization.
- Roles are assigned predefined permissions.

Example:

- HR role → access payroll systems.
- Finance role → access accounting systems.
- IT Admin role → manage servers.

Strengths:

- Scales well in enterprises.
- Easier to manage than individual permissions.
- Supports **least privilege** (users only get rights for their role).

Weaknesses:

- Role explosion if too many roles are created.
- May require regular reviews to ensure roles match job functions.

👉 **Exam Tip:** If you see "job function determines access" → it's RBAC.

7.5 Rule-Based Access Control (RuBAC)

Definition:

- Access decisions are made based on system-enforced rules (not individual or role).

Example:

- Firewall rules → Block all traffic except port 443.
- Time-based access → Employees can log in only between 9am–6pm.

Strengths:

- Useful for dynamic enforcement (firewalls, routers).
- Fine-grained control.

Weaknesses:

- Complexity can increase with many rules.

7.6 Attribute-Based Access Control (ABAC)

Definition:

- Access decisions are based on attributes of subject, object, action, and environment.

Example:

- Policy: "Managers (subject attribute) can approve expense reports (object) under $10,000 (action), during business hours (environment)."

Strengths:

- Highly flexible and granular.
- Well-suited for cloud environments and large enterprises.

Weaknesses:

- Complexity in defining and managing policies.
- Requires strong governance to prevent misconfigurations.

👉 **Exam Tip:** If you see "policies using attributes like user, resource, environment" → it's ABAC.

7.7 Other Access Control Principles

- **Least Privilege** → Users get only the access necessary to perform duties.
- **Need-to-Know** → Even with clearance, access only to data needed for tasks.
- **Separation of Duties (SoD)** → Splits tasks among multiple people to prevent fraud.
    - Example: One person initiates a payment, another approves it.
- **Job Rotation** → Rotating duties reduces fraud and ensures cross-training.

7.8 Comparison Table (High-Yield for CISSP)

Model	Who Controls?	Basis	Example	Strength	Weakness
DAC	Owner	User discretion	File permissions in Windows	Flexible	Weak security,

					prone to abuse
MAC	System (labels)	Clearance & classification	Military systems	Strong security	Inflexible
RBAC	System (roles)	Job functions	HR role, IT Admin	Scalable	Role explosion
RuBAC	System (rules)	Time, location, firewall rules	Time-based login	Granular	Complex rules
ABAC	System (policies)	Attributes (user, resource, env)	Cloud IAM	Very flexible	Complex, harder to audit

Section 8: Identity as a Service (IDaaS) and Cloud IAM

8.1 What is IDaaS?

Identity as a Service (IDaaS) refers to cloud-based identity and access management services provided on a subscription model.
Instead of building and maintaining IAM infrastructure on-premises, organizations outsource it to specialized vendors.

Examples of IDaaS Providers:

- **Okta** (pioneer in IDaaS, widely adopted).
- **Microsoft Azure AD / Entra ID** (integrated with Office 365, Azure services).
- **Google Identity Platform**.
- **AWS IAM** (manages access to AWS resources).
- **Ping Identity**, **Auth0**.

8.2 Why Organizations Use IDaaS

1. **Scalability** → Easily handles thousands/millions of identities without local infrastructure.
2. **Cost Reduction** → No need to maintain on-prem IAM servers.

3. **Cloud-First Strategy** → Supports SaaS/IaaS applications natively.
4. **User Experience** → Provides centralized SSO and MFA for cloud and on-prem apps.
5. **Security Enhancements** → Continuous monitoring, risk-based authentication, built-in MFA.

8.3 Key Features of IDaaS

- **Single Sign-On (SSO)** across cloud, mobile, and on-prem applications.
- **Federation Support** (SAML, OAuth 2.0, OpenID Connect).
- **Multi-Factor Authentication (MFA)** integrated into login flows.
- **Provisioning and De-provisioning** automated through HR or IT systems.
- **Directory Integration** → Sync with Active Directory or LDAP.
- **Access Certification and Governance** for compliance.
- **Privileged Identity Management (PIM)** to control and monitor elevated accounts.
- **API Security** → Protect access between applications/services.

8.4 Cloud IAM (Provider-Specific)

AWS IAM (Identity & Access Management)

- Manages access to AWS services/resources.
- Provides users, groups, roles, and policies.
- Supports least privilege via JSON-based policies.
- Supports federation with corporate AD via SAML.
- Security concern: Misconfigured S3 bucket permissions often cause data leaks.

Microsoft Azure AD (Entra ID)

- Provides cloud-based directory services.
- Enables conditional access policies (e.g., block login from risky IPs).
- Integrates with Microsoft 365 and thousands of SaaS apps.
- Supports B2B (partners) and B2C (customers) identities.

Google Identity Platform

- Provides SSO and federation for G Suite/Google Workspace.
- Strong integration with Google Cloud Platform (GCP).

Okta / Ping Identity / Auth0

- Independent IDaaS providers with advanced federation, adaptive authentication, and CIAM (Customer IAM).
- Popular for hybrid/multi-cloud enterprises.

8.5 IDaaS vs Traditional IAM

Feature	Traditional IAM (On-Prem)	IDaaS (Cloud)
Deployment	In-house servers	Cloud-hosted
Cost	High upfront (CAPEX)	Subscription (OPEX)
Scalability	Limited	Elastic
Updates	Manual patches	Automatic by provider
Access	Usually internal	Anywhere (global)
Federation	Limited	Built-in (SAML, OAuth, OIDC)

8.6 Risks and Challenges of IDaaS

- **Vendor Lock-In** → Switching providers may be costly/difficult.
- **Reliance on Internet Connectivity** → Outage can disrupt access to resources.
- **Shared Responsibility Model** → Customer still responsible for configuring policies correctly.
- **Compliance Issues** → Must ensure provider meets GDPR, HIPAA, SOX requirements.
- **Insider Threats at Provider** → Blind trust in provider security controls.

8.7 Best Practices for Cloud IAM

1. **Enforce MFA** for all accounts, especially admins.
2. **Implement Conditional Access Policies** (geo-location, device posture).

3. **Apply Least Privilege** → Granular policies, avoid "*" wildcards.
4. **Monitor and Audit** → Use logging (AWS CloudTrail, Azure AD logs).
5. **Secure API Keys and Tokens** → Rotate regularly, never hardcode.
6. **Use Privileged Identity Management (PIM)** → Just-in-time admin privileges.

8.8 Exam-Focused Points

- **SSO + Federation** → IDaaS makes this seamless across SaaS apps.
- **Okta, Azure AD, Ping** = well-known IDaaS providers.
- **AWS IAM** = identity management inside AWS ecosystem.
- **Biggest risk** = **misconfiguration and vendor lock-in**.
- **Cloud IAM** = foundation of **Zero Trust** in cloud.

Section 9: Access Control Attacks and Mitigations

Access control is only as strong as the mechanisms protecting it. Attackers constantly attempt to bypass, exploit, or weaken authentication and authorization systems. Understanding these attacks and their mitigations is key for both the exam and real-world security.

9.1 Password Attacks

Passwords are the most common authentication factor, and therefore, the most targeted.

Types of Password Attacks

1. **Brute-Force Attack**
    - Attacker tries every possible combination until the correct one is found.
    - Example: Trying all 8-character combinations → billions of possibilities.
    - Mitigation: Account lockouts, login attempt throttling, long passwords.
2. **Dictionary Attack**
    - Uses a precompiled list of common passwords and words.
    - Example: "Password123" is one of the first tested.

- Mitigation: Enforce strong password policies, disallow common passwords.

3. **Credential Stuffing**
    - Uses stolen username/passwords from breaches to try across multiple sites.
    - Example: Using a leaked Netflix password on banking sites.
    - Mitigation: MFA, unique passwords per system, breach monitoring.
4. **Password Spraying**
    - Tries a few common passwords against many accounts (avoids lockouts).
    - Example: Testing "Welcome@123" against all employee accounts.
    - Mitigation: MFA, account monitoring for failed login patterns.
5. **Rainbow Table Attack**
    - Uses precomputed hash values to crack passwords.
    - Mitigation: Salt + strong hash algorithms (bcrypt, Argon2).

9.2 Pass-the-Hash Attack

- Attackers steal **hashed passwords** from memory or disk.
- Instead of cracking the hash, they "replay" it to authenticate.
- Common in **Windows environments** with NTLM authentication.

Mitigations:

- Disable NTLM where possible, use Kerberos.
- Limit admin accounts and use unique local admin passwords.
- Implement Credential Guard or similar protections.
- Regularly patch systems.

9.3 Replay Attacks

- Attacker captures authentication traffic and reuses it later.
- Example: Capturing Kerberos ticket or session token.

Mitigations:

- Use timestamps and nonces (Kerberos includes this).
- Enforce short session lifetimes.
- Encrypt traffic (TLS).

9.4 Privilege Escalation Attacks

- Attacker gains more privileges than intended.

Types:

1. **Vertical Escalation** → Normal user → admin.
2. **Horizontal Escalation** → User A accesses User B's data.

Mitigations:

- Principle of least privilege.
- Regular access reviews.
- Patch privilege escalation vulnerabilities.
- Monitor for abnormal account activity.

9.5 Social Engineering Attacks on Access Control

- Attackers trick humans into bypassing security.

Examples:

- Phishing for credentials.
- Pretexting helpdesk to reset passwords.
- Piggybacking/tailgating into secure areas.

Mitigations:

- Security awareness training.
- Anti-phishing tools and MFA.
- Physical controls (badges, turnstiles, mantraps).

9.6 Insider Threats

- Employees misusing legitimate access (fraud, data theft).

- Often the hardest to detect since access is valid.

Mitigations:

- Job rotation and mandatory vacations (fraud detection).
- Monitoring user behavior analytics (UBA).
- Strict separation of duties (SoD).
- Termination checklists (revoke all access immediately).

9.7 Session Hijacking

- Attacker takes over an active session after authentication.
- Example: Stealing a session cookie from a web app.

Mitigations:

- Use secure cookies (HttpOnly, Secure, SameSite).
- Regenerate session IDs after login.
- Use TLS to prevent sniffing.

9.8 Access Control Misconfigurations

- Over-permissive access → "Everyone: Full Control."
- Default accounts left enabled (admin/admin).
- Failure to revoke accounts after employee exits.

Mitigations:

- Regular access reviews.
- Automated provisioning/de-provisioning.
- Disable default accounts immediately.
- Implement governance policies.

9.9 Exam-Focused Summary

- **Password Attacks** → Brute-force (all combos), Dictionary (wordlist), Rainbow Table (precomputed hashes), Credential Stuffing (reuse leaked creds), Spraying (common password across many accounts).

- **Pass-the-Hash** → Windows NTLM weakness → use Kerberos.
- **Replay Attacks** → Captured credentials reused → mitigated by timestamps, nonces, encryption.
- **Privilege Escalation** → Least privilege, access reviews, patching.
- **Social Engineering** → Training, MFA, physical controls.
- **Insider Threats** → Job rotation, UBA, SoD.
- **Session Hijacking** → Secure cookies, TLS, regenerate IDs.
- **Misconfigurations** → Regular audits, disable defaults.

THANK YOU

Enroll with MoS – **CISSP Training @ ₹4,999**!

WWW.MINISTRYOFSECURITY.CO