Introduction

• Introduces key concepts related to computer security and cryptography.
• Security is not just a product, but a process that needs to be managed properly.
• Nothing is 100% secure, and the need for security should be balanced with its cost.
• The document mentions the three golden rules for ensuring computer security: not owning a computer, not powering it on, and not using it, as humorously stated by Robert (Bob) Morris.
• The security of a system is equivalent to the security of its weakest component.
• Security by obscurity is not effective, and cryptography alone is not enough to ensure security.
• Users cannot be relied upon for security, as they tend to prioritize convenience over security.
• The document discusses the meaning of security, which includes confidentiality, integrity, availability, authentication, non-repudiation, authorisation, auditing, attack-tolerance, disaster recovery, key-recovery, and digital forensics.
• It mentions different security mechanisms such as random numbers, encryption/decryption, hash functions, digital signatures, key exchange, and time stamping.
• Types of attackers are classified as outsiders and insiders, and types of attacks can be passive or active.
• The document briefly mentions TEMPEST, a signal intelligence attack, and the use of big data for user profiling.

1. Overview

- The document provides an overview of computer security and defines computer security as the protection afforded to an automated information system to preserve the integrity, availability, and confidentiality of its resources, including hardware, software, firmware, information/data, and telecommunications.
- Key security concepts include:
    - Computer Security challenges are not simple and require consideration of potential attacks. The procedures used can be counter-intuitive and involve algorithms and secret information. Deciding where to deploy mechanisms becomes a battle of wits between the attacker and the administrator. The importance of security is often not perceived until it fails, and it requires regular monitoring. Unfortunately, it is too often an afterthought and regarded as an impediment to using the system.
    - Security terminology includes vulnerabilities and Attacks: System resources with vulnerabilities can be corrupted, become leaky, or become unavailable. Attacks can be passive or active and can be carried out by insiders or outsiders.
    - Countermeasures: Countermeasures are means used to deal with security attacks and can prevent, detect, or recover from them.

However, countermeasures may introduce new vulnerabilities and will always have residual vulnerability. The goal is to minimize risk within given constraints.

- The document introduces security terminology, including vulnerabilities, attacks, system resources, and different types of attacks.
- Technical measures such as access control, identification and authentication, system and communication protection, and system and information integrity are essential for security. Management controls and procedures, including awareness and training, audit and accountability, contingency planning, and personnel security, are also important.
- It discusses vulnerabilities and attacks, highlighting that system resources with vulnerabilities may experience loss of integrity, loss of confidentiality, or loss of availability. Attacks can be passive or active, carried out by insiders or outsiders.
- Network security attacks can be classified as passive or active. Passive attacks involve eavesdropping and traffic analysis, while active attacks modify or fake data. Preventing passive attacks is challenging, so the focus is on detection. Preventing active attacks is also challenging, so the focus is on prevention.
- Countermeasures are the means used to deal with security attacks, including prevention, detection, and recovery. However, countermeasures may introduce new vulnerabilities and have residual vulnerability.
- Threat consequences include unauthorized disclosure, deception, disruption, and usurpation.
- The scope of computer security is broad, covering network security attacks, security functional requirements, and the use of technical and management controls and procedures.
- The document mentions the X.800 Security Architecture which is a security architecture for OSI, which provides a systematic way of defining security requirements and approaches to satisfying them. It defines security attacks, security mechanisms, and security services.
- The document briefly mentions security taxonomy, security trends, computer security losses, security technologies used, and computer security strategy, which includes specification/policy, implementation/mechanisms, and correctness/assurance.

2. User authentication

- User authentication is a fundamental security building block and the basis of access control and user accountability.
- User authentication involves verifying the identity claimed by a system entity through two steps: identification (specifying an identifier) and verification (binding the entity and identifier).
- There are four means of authenticating a user's identity: something the individual knows (e.g., password, PIN), something the individual possesses (e.g., key, token, smartcard), something the individual is (static biometrics, e.g., fingerprint, retina), and something the individual does (dynamic biometrics, e.g., voice, sign).

- Password authentication is a widely used method where the user provides a name/login and password, which is compared to the saved password for the specified login.
- Password authentication is vulnerable to various attacks, such as offline dictionary attacks, specific account attacks, password guessing against a single user, workstation hijacking, exploiting user mistakes, exploiting multiple password use, and electronic monitoring.
- Countermeasures for password vulnerabilities include stopping unauthorized access to password files, using intrusion detection measures, implementing account lockout mechanisms, enforcing policies against using common passwords, providing training and enforcement of policies, automatic workstation logout, and using encrypted network links.
- Hashed passwords are commonly used in implementations, with stronger hash/salt variants such as MD5 or Blowfish-based hash algorithms. These implementations provide better security and can handle longer passwords.
- Password cracking techniques include dictionary attacks (trying each word and obvious variants in a large dictionary against the hash in the password file) and rainbow table attacks (precomputing tables of hash values for all salts).
- Users often choose weak passwords, such as short passwords or easily guessable passwords, which make cracking passwords easier for attackers.
- Access control for password files can mitigate offline guessing attacks by denying access to encrypted passwords, making them available only to privileged users, and using separate shadow password files.
- Improving password choices can be achieved through user education (informing users about password best practices), computer-generated passwords (system-generated passwords that are difficult to guess), and proactive password checking (implementing rules and guidance to reject weak or easily guessable passwords).
- Token authentication involves using objects like embossed cards, magnetic stripe cards, memory cards, or smartcards to authenticate users. These tokens can be used alone or in combination with a password/PIN for computer use.
- Biometric authentication verifies the user's identity based on one of their physical characteristics, such as fingerprints or retinas.
- Remote user authentication is more complex and often involves challenge-response protocols to protect against attacks like eavesdropping and replay.
- The document provides a comprehensive summary of user authentication methods, including passwords, tokens, and biometrics, as well as the security issues associated with authentication.

Important Points:

- User authentication is a fundamental security building block and the basis of access control and user accountability.
- Password authentication is widely used but vulnerable to various attacks, such as dictionary attacks and password guessing.
- Countermeasures for password vulnerabilities include stopping unauthorized access to password files and enforcing policies against using common passwords.

- Hashed passwords with stronger algorithms, such as MD5 or Blowfish, are commonly used in implementations.
- Password cracking techniques include dictionary attacks and rainbow table attacks.
- Access control for password files can mitigate offline guessing attacks.
- Improving password choices can be achieved through user education, computer-generated passwords, and proactive password checking techniques.
- Token authentication involves the use of objects possessed by the user, such as embossed cards, magnetic stripe cards, memory cards, or smartcards.
- Biometric authentication authenticates users based on their physical characteristics, such as fingerprints or retinas.
- Remote user authentication is more complex and often involves challenge-response protocols to protect against attacks like eavesdropping and replay.

3. Denial of Service

- Denial of Service (DoS) is an action that prevents or impairs the authorized use of networks, systems, or applications by exhausting resources such as CPU, memory, bandwidth, and disk space.
- Classic Denial of Service Attacks can use simple flooding ping from a higher capacity link to a lower one, causing a loss of traffic. The source of flood traffic is easily identified.
- Source Address Spoofing is another common attack where forged source addresses are used to generate large volumes of packets directed at a target. The congestion caused by the flood responses is scattered across the Internet, making the real source harder to identify.
- SYN Spoofing is an attack on the ability of a server to respond to future connection requests by overflowing the tables used to manage them. This results in an attack on system resources.
- Different types of flooding attacks include ICMP Flood, which uses ICMP packets; UDP Flood, which uses UDP packets to some port; and TCP SYN Flood, which uses TCP SYN (connection request) packets for a volume attack.
- Distributed Denial of Service (DDoS) Attacks involve multiple systems that allow for much higher traffic volumes. These attacks often involve compromised PCs or workstations forming a botnet.

4. Intrusion Detection

1. **Intruders**

- Intruders can range from benign to serious and can be categorized as user trespass, software trespass, masquerader, misfeasor, or clandestine user.

2. **Examples of Intrusion**

- Examples of intrusion include remote root compromise, web server defacement, password guessing or cracking, copying sensitive data, running a packet sniffer, distributing pirated software, accessing the internet through unsecured connections, impersonating a user, and using unattended workstations.

3. **Hackers**

- Hackers are motivated by the thrill of access and status. The hacking community operates on a meritocratic system, where status is determined by competence.
- Intrusions by benign hackers can still consume resources and slow down performance.
- Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), and Virtual Private Networks (VPNs) can help counter hacking attempts.
- The establishment of Cyber Emergency Response Teams (CERTs) aims to collect and disseminate vulnerability information and responses.

4. **Hacker Behavior Example**

- Hackers follow a series of steps, including selecting a target using IP lookup tools, mapping the network for accessible services, identifying potentially vulnerable services, brute-forcing passwords, installing remote administration tools, waiting for admin logins, and using obtained passwords to access the network.

5. **Criminal Enterprise**

- Organized groups of hackers pose a significant threat, particularly to corporations, governments, and e-commerce servers.
- Criminal hackers usually have specific targets and act quickly and discreetly.
- IDS, IPS, and other security measures are less effective against criminal enterprise attacks.
- Sensitive data requires strong protection measures.

6. **Criminal Enterprise Behavior**

- Criminal enterprise behavior includes acting quickly and precisely to avoid detection, exploiting vulnerable ports, using trojan horses for re-entry, capturing passwords with sniffers, and making few or no mistakes.

7. **Insider Attacks**

- Insider attacks are among the most difficult to detect and prevent, as employees have access and systems knowledge.
- Motivations for insider attacks can include revenge or a sense of entitlement.
- Preventive measures for insider attacks include implementing the principle of least privilege, monitoring logs, using strong authentication, implementing termination processes to block access, and mirroring data.

8. **Insider Behavior Example**

- Insider behavior examples include creating network accounts for themselves and their friends, accessing accounts beyond their daily job requirements, contacting former and prospective employers, engaging in discreet instant messaging chats, visiting websites catering to disgruntled employees, performing large downloads and file copying, and accessing the network during off-hours.

9. **Intrusion Detection Systems**

- Intrusion Detection Systems (IDSs) can be classified as host-based IDS (HBIDS) and network-based IDS (NBIDS).
- HBIDS monitors system activity on a single host, while NBIDS monitors network traffic.
- The logical components of IDS include sensors to collect data, analyzers to determine intrusion, and a user interface for managing and viewing IDS.

10. **IDS Principles**

- IDS assumes that intruder behavior differs from legitimate users and observes deviations from past history.

- IDS faces challenges of false positives, false negatives, and the need to compromise accuracy.

11. **Host-Based IDS**
- Host-Based IDS is specialized software that monitors system activity to detect suspicious behavior.
- HBIDS's primary purpose is to detect intrusions, log suspicious events, and send alerts.
- HBIDS can detect both external and internal intrusions and uses anomaly-based and signature-based detection approaches.

12. **Audit Records**
- Audit records are fundamental tools for intrusion detection.
- Audit records can be native audit records provided by the operating system or detection-specific audit records specific to IDS tasks.
- Detection-specific audit records often contain fields for subject, action, object, exception-condition, resource-usage, and time-stamp.

13. **Anomaly Detection**
- Anomaly detection includes threshold detection and profile-based detection.
- Threshold detection checks excessive event occurrences over time, while profile-based detection characterizes past behavior to detect significant deviations.

14. **Signature Detection**
- Signature detection observes events on the system and applies a set of rules to decide if an intruder is present.
- Signature detection can be rule-based anomaly detection, which matches current behavior with expected behavior, or rule-based penetration identification, which identifies known penetrations or weaknesses.

15. **Distributed Host-Based IDS**
- Distributed Host-Based IDS architecture involves multiple HBIDS sensors distributed across a network.

16. **Network-Based IDS**
- Network-Based IDS (NIDS) monitors network traffic at specific points to detect intrusion patterns.
- NIDS examines network, transport, and/or application level protocol activity directed towards systems.
- NIDS comprises a number of sensors, including inline sensors and passive sensors.

17. **Intrusion Detection Techniques**
- Intrusion detection techniques include signature detection and anomaly detection.
- Signature detection focuses on unexpected application services, policy violations, denial of service attacks, scanning, and worms.
- Anomaly detection aims to detect deviations from normal behavior, such as abnormal resource utilization or unusual user behavior.

5. Firewalls and Intrusion Prevention Systems

Firewalls and Intrusion Prevention Systems (IPS) are effective means of protecting LANs and are essential for organizations and individuals who have internet connectivity.

Firewalls can be used to secure workstations and servers, and they can also be used as a perimeter defense by creating a single choke point to impose security.
There are different types of firewalls, including:

- Packet Filtering Firewall: This type of firewall applies rules to packets based on information in the packet header, such as source/destination IP address and port. It uses a list of rules to determine whether to forward or discard a packet.
- Stateful Inspection Firewall: This type of firewall reviews packet header information and also keeps track of TCP connections. It tightens rules for TCP traffic and only allows incoming traffic to high-numbered ports for packets matching an entry in its directory.
- Application-Level Gateway: This type of firewall acts as a relay for application-level traffic. It authenticates users, contacts the application on the remote host, and relays TCP segments between the server and the user.
- Circuit-Level Gateway: This type of firewall sets up two TCP connections, one to an inside user and one to an outside host, and relays TCP segments between the connections without examining the contents.

Firewalls can be located in different ways, such as using bastion hosts, individual host-based firewalls, or personal firewalls.
Intrusion Prevention Systems (IPS) are a recent addition to security products. They can block traffic like a firewall and also have intrusion detection system (IDS) capabilities.
IPS can be host-based or network-based and use both signature techniques (identifying malicious application packets) and anomaly detection techniques (identifying behavior patterns that indicate malware).
There are different firewall topologies, such as host-resident firewalls, screening routers, single bastion firewalls, double bastion firewalls, and distributed firewall configurations.
The document also introduces the concept of Unified Threat Management (UTM) products, which combine multiple security features into a single appliance.
The summary concludes by summarizing the need for firewalls, the different types of firewalls, firewall hosting, intrusion prevention systems, and firewall topologies.

6. Buffer Overflow

Buffer overflow is a common attack mechanism, with examples including the Morris Worm, Code Red, Slammer, Sasser, and many others.

Prevention techniques exist, but buffer overflow is still a major concern due to legacy buggy code and continued careless programming techniques.

Buffer overflow is caused by a programming error that allows more data to be stored than the capacity available in a fixed-sized buffer.

Buffer overflow can lead to overwriting adjacent memory locations, corruption of program data, unexpected transfer of control, memory access violation, and execution of code chosen by the attacker.

Memory is a flat sequence of bytes, each identified by an address. Memory protection allows for marking areas of memory as readable, writable, or executable.

Interpreting memory involves understanding how certain ranges of bytes are interpreted, such as little-endian byte order for integers and pointers on x86.

Linux process memory includes the main executable, heap for dynamic allocation, libraries, stack for temporary variables, and code, data, and read-only sections.

An example of a buffer overflow is when a program copies user input to a fixed-size buffer and allows more data to be stored than the buffer's capacity.

Buffer overflow can result in the overwriting of adjacent memory locations with user input, potentially leading to unexpected behavior or security vulnerabilities.

7. Operating System Security

- Operating System (OS) security is a hardening process that includes planning installation, configuration, update, and maintenance for the OS and key applications.
- Implementing basic hardening measures can prevent a large proportion of attacks seen in recent years.
- Patching the OS and third-party apps, restricting admin privileges, and white-listing approved applications are effective ways to improve OS security.
- It is important to plan the build and deployment process to counter threats and assess risks.
- Underlying OS and applications should be secured, critical content should be protected, and network protection mechanisms should be in place.

Important Points:

1. OS Security:
    - Basic hardening measures can prevent a large proportion of attacks.
    - Patch the OS and third-party apps.
    - Restrict admin privileges to users who need them.
    - White-list approved applications.
2. User Management:
    - Set file and directory permissions carefully.
    - Use groups to differentiate between roles.
    - Use extreme care in granting and using root privileges.
    - Manage user's group memberships and set appropriate password ages.
3. Running As Unprivileged User/Group:
    - Every process "runs as" some user.

   - It is important to ensure this user is not root to prevent compromising the entire system.
   - Use a dedicated user/group to make it easier to identify the source of log messages.
4. Logging:

   - Effective logging is crucial for system security.
   - Linux logs using syslogd or Syslog-NG, which receive log data from various sources and write log messages to local or remote log files.
   - Syslog-NG provides more flexibility in configuring log-data sources and destinations and can log via TCP, which can be encrypted.
   - Customizing log defaults is recommended.
5. Log Management:

   - Balance the number of log files used to manage the size of log files.
   - Rotate log files and delete old copies using logrotate utility.
   - Configure application logging in addition to managing system logs.
6. Application Security:

   - Running applications as unprivileged user/group, running in a chroot jail, modularity, encryption, and logging are important considerations for application security.
7. Windows Security:

   - Windows security architecture includes the Security Reference Monitor (SRM), Local Security Authority (LSA), Security Account Manager (SAM), Active Directory (AD), WinLogon (local), and NetLogon (net).
8. Windows Privileges:

   - Windows privileges are systemwide permissions assigned to user accounts.
   - Some privileges are deemed "dangerous," while others are deemed "benign."
9. Windows System Hardening:

   - Process of shoring up defenses, reducing exposed functionality, and disabling features.
   - Servers are easier to harden due to specific and controlled purposes and better computer configuration skills of administrators.
10. Stripping Privileges:

   - It is important to strip privileges from an account soon after an application starts.
   - For example, the Index server process runs as system but sheds unnecessary privileges as soon as possible.

8. Access Control

- Access control is the prevention of unauthorized use or access to resources in a computer system.
- It is a central element of computer security, and it involves users and groups authenticating to the system and being assigned access rights to specific resources.
- Access control principles include reliable input, fine and coarse specifications, least privilege, separation of duty, open and closed policies, policy combinations, conflict resolution, and administrative policies.
- The elements of access control include **subjects** (entities that can access objects), **objects** (access controlled resources such as files, directories, records, programs), and access rights (ways in which subjects access objects, such as read, write, execute, delete).
- **Discretionary access control** is often provided using an access matrix, which lists subjects in one dimension and objects in the other dimension, specifying the access rights of a subject to an object.
- Access control models and structures help in organizing and implementing access control policies and requirements.
- Protection domains are sets of objects with associated access rights, and they define the access permissions in the access matrix.
- UNIX file concepts involve inodes (control structure with information on file attributes and permissions), directories (hierarchical trees containing files and directories), and file access control using inodes and directories.
- UNIX provides additional access control features such as set user ID (SetUID) and set group ID (SetGID) to temporarily use the rights of the file owner/group, a sticky bit on directories to limit certain actions to the owner, and the superuser who is exempt from usual access control restrictions.
- Modern UNIX systems also support Access Control Lists (ACLs), which allow specifying additional users/groups and their associated permissions beyond the standard permissions.
- Role-Based Access Control (RBAC), as defined by the NIST RBAC Model, restricts system access based on users' assigned roles and responsibilities

Important Points:

- Access control is crucial to prevent unauthorized use of resources in a computer system.
- Access control principles include reliable input, least privilege, separation of duty, open and closed policies, policy combinations, and administrative policies.
- Access control elements involve subjects, objects, and access rights.
- Discretionary access control uses an access matrix to define access rights for subjects to objects.
- Protection domains define access permissions in the access matrix.
- UNIX file concepts involve inodes, directories, and file access control using inodes and directories.
- UNIX provides additional access control features such as SetUID, SetGID, sticky bit, and superuser.

- ACLs in modern UNIX systems allow specifying additional users/groups and permissions beyond standard permissions
- Role-Based Access Control (RBAC), as defined by the NIST RBAC Model, restricts system access based on users' assigned roles and responsibilities

9. Malicious Software

- Malicious software, also known as malware, refers to programs that exploit system vulnerabilities.
- Malware can be in the form of viruses, logic bombs, backdoors, worms, bots, or other types of programs.
- Viruses are software that infect other programs by modifying them to include a copy of the virus. They go through phases of dormancy, propagation, triggering, and execution.
- Worms are replicating programs that spread over a network by connecting to other systems and copying themselves. They can disguise themselves as system processes and have been used in various attacks.
- Other types of malware include logic bombs, trojan horses, mobile code, rootkits, keyloggers, and zombies.
- Malware can have different objectives, such as system corruption, attack agent, information theft, or stealth access.
- Countermeasures against malware include prevention through policies, awareness, and vulnerability mitigation, as well as detection, identification, and removal of malware.
- Anti-virus technology has evolved over time, with different generations of scanners and methods for detecting and removing viruses.
- Worm countermeasures overlap with anti-virus techniques, and once a worm is on a system, it can be detected by anti-virus software.

10. Software Security

One of the key points noted is that many vulnerabilities in software result from poor programming practices, including insufficient checking and validation of program input.
The document emphasizes the importance of awareness of software security issues and highlights the Open Web Application Security Top Ten, which includes five software-related flaws.
It mentions that **SSL** (Secure Sockets Layer) is widely used but not fully understood by developers, leading to vulnerabilities in mobile apps. Some libraries, like cURL, are also mentioned as being error-prone.
The document addresses the motivation for software security, highlighting the risks of faulty SSL implementations that can expose apps to man-in-the-middle attacks and put credentials and private information at risk.
It introduces MITHYS (**Mind The Hand You Shake**), a solution to protect mobile devices from SSL usage vulnerabilities. MITHYS informs users about vulnerable applications, detects ongoing attacks, and prevents the use of SSL-vulnerable applications.

The document discusses the importance of both software quality and security, noting that while software quality focuses on accidental failures from unanticipated input, software security deals with intentional attacks.
It introduces the concept of defensive programming, which aims to ensure the continued function of software despite unforeseen usage, by validating assumptions and checking for potential errors.
The document emphasizes the need to handle program input correctly, including identifying all data sources, explicitly validating assumptions on size and type, and preventing buffer overflow vulnerabilities.
It mentions the interpretation of program input, highlighting the need to validate interpretations before use, especially for inputs like filenames, URLs, email addresses, and identifiers.
The document discusses injection attacks, which occur when invalid input influences program execution, often in scripting languages or web CGI scripts.
It provides examples of safe coding practices, such as validating input syntax, handling alternate encodings, validating numeric input, and input fuzzing.
The document also touches on other topics related to software security, such as writing safe program code, dealing with race conditions in shared memory, interacting with the operating system, system calls and standard library functions, and handling program output.
Overall, the document provides insights into software security issues, best practices for handling program input, and safe coding techniques

Important Points:

- Many vulnerabilities result from poor programming practices.
- Insufficient checking and validation of program input can lead to vulnerabilities, such as buffer overflow.
- SSL is widely used but not fully understood by developers, leading to vulnerabilities in mobile apps.
- Some libraries, like cURL, are error-prone.
- Faulty SSL implementations expose apps to man-in-the-middle attacks and put credentials at risk.
- MITHYS is a solution to protect mobile devices from SSL usage vulnerabilities.
- Software quality and security are related but address different types of failures.
- Defensive programming and secure programming aim to ensure software resilience.
- Correct handling of program input is crucial to prevent vulnerabilities.
- Validation of input size, interpretation, and assumptions is necessary.
- Injection attacks occur when invalid input influences program execution.
- Safe coding practices include validating input syntax, handling alternate encodings, and input fuzzing

11. Trusted Computing & Multilevel Security

1. Introduction to Trusted Computing:
    - Trusted computing refers to the extent to which someone can have confidence in a system meeting its specifications and performing desired functions.

- A trusted system is believed to enforce a given set of attributes with a certain level of assurance.
- Building a trusted system requires a root of trust (RoT) and a chain of trust (CoT).
- The RoT is the element that is implicitly trusted, and the CoT is a cascade relationship of trust between entities starting from the RoT.

2. Trust in Computing:
    - Trust is the confidence that a system performs its claimed functions and does not perform unwanted actions.
    - Trust can be inherited from the RoT to build the CoT.
3. Creating a Root of Trust:

    - There are two approaches to creating a RoT: a priori and a posteriori.
    - A priori approach involves creating a protected region of memory or monitoring the system state to detect unexpected changes.
    - Hardware RoT uses hardware isolation, while hybrid RoT combines minimal hardware and strict estimation of computational complexity and communication speed.
    - External verifiers can challenge the system with function execution to assess the system state and ensure functions are executed within a precise time.
4. Multi-Level Security:
    - Multi-level security (MLS) involves classifying sensitive information into security classes and assigning access rights based on security classification.
    - Security classes control modes of access, such as read, append, write, and execute.
    - Properties of MLS ensure no read up (subjects can only read objects of less or equal security level) and no write down (subjects can only write into objects of greater or equal security level).
    - MLS mechanisms prevent users or processes with lower clearances from accessing information at higher security levels.
5. Access Control List (ACL) in Linux:
    - ACL in Linux consists of a set of rules that specify how a user or group can access files and directories.
    - Default ACL entries are set on directories and specify the default access information for files within the directory.
    - ACLs enable fine-grained control over access permissions.