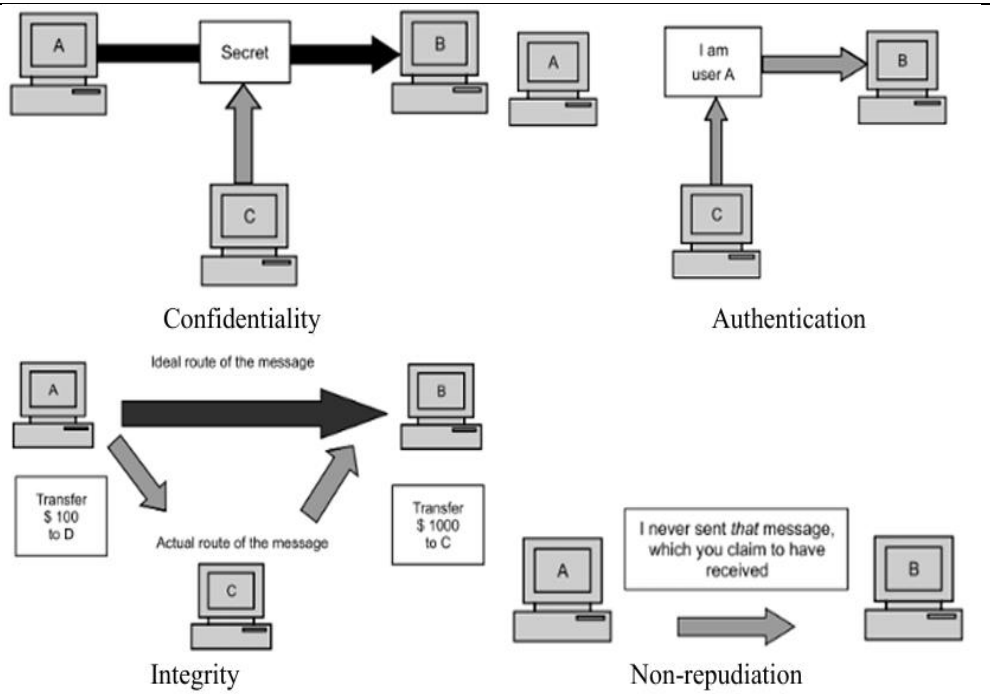
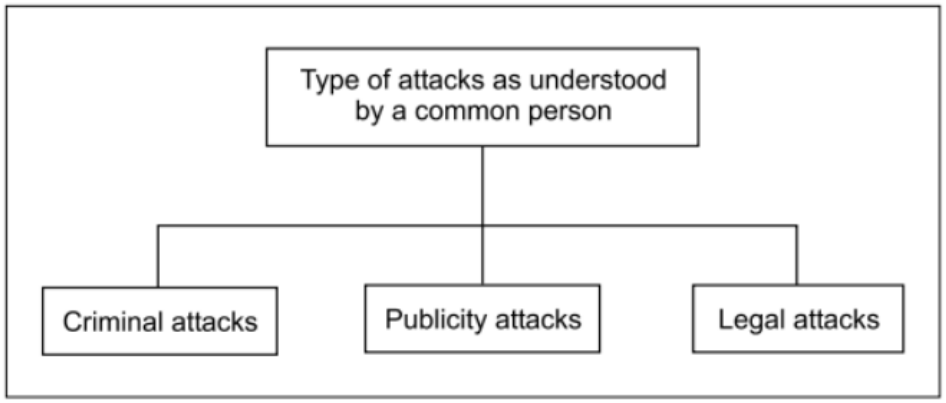
BEC714B

VTU Question Paper Solution Seventh semester BE Dec-2025 - Jan-2026
Computer and Network Security

Q.1 a)	**Define the need for computer security. Explain the principles of security.**	**5**	**CO1**	**L1**
Ans	● Business & different types of transactions are being conducted to a large extent over the Internet. ● Inadequate or improper *security mechanisms can bring the whole business down or play havoc with people's lives!* ● *Since Electronic Documents & Messages are now becoming equivalent to proper documents in terms of their legal validity & binding.* ● Businesses collect mass amounts of data about their customers, employees, and competitors. ● Most of this data is stored on computers and transmitted across networks. ● If this information should fall into the hands of a competitor, the result could be loss of business, lawsuits and bankruptcy. ● Protecting corporate data is no longer an option, it is a requirement. Principles of security: The 4 chief principles of security are 1. Confidentiality: - Is msg seen by someone else? 2. Authentication: - Do you trust the sender of msg? 3. Integrity: - Is the meg changed during transmission? 4. Non-repudiation: - Can the sender refuse the message?			



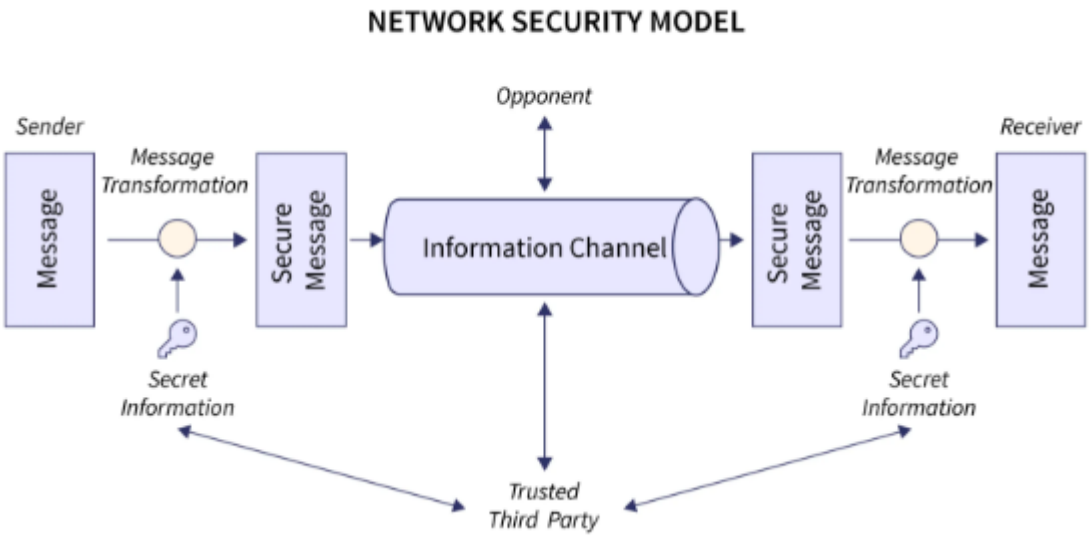
b)	**Explain different types of attacks with examples.**	**5**	**CO1**	**L2**
Ans	Attacks shall be classified into three categories in general, as shown in			



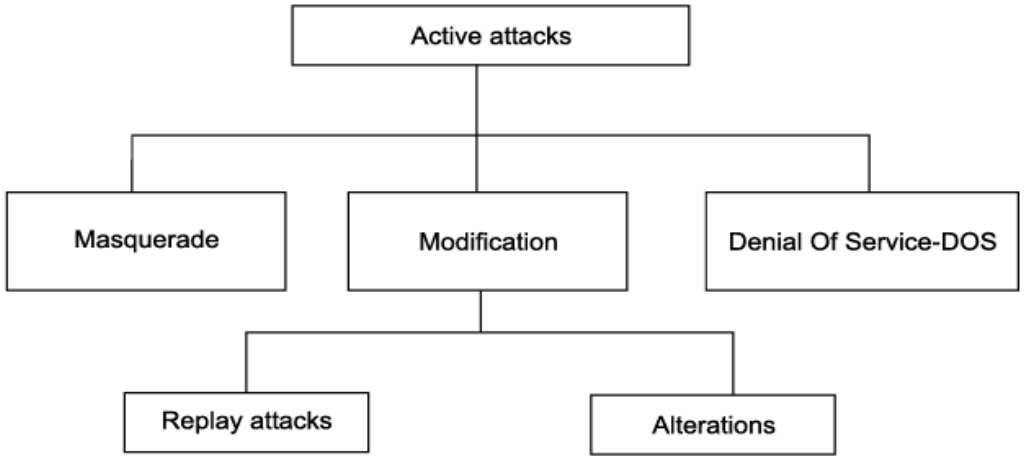
Criminal Attacks: Criminal Attackers sole aim is to maximize financial gain by attacking computer systems. Table below lists some forms of criminal attacks.

Publicity Attacks: Publicity attacks occur because the attackers want to see their names appear on television news channels and newspapers.

Attack	Description
Fraud	Modern fraud attacks concentrate on manipulating some aspects of electronic currency, credit cards, electronic stock certificates, cheques, letters of credit, purchase order, ATMs etc.
Scams	Some forms of scams are sales of services, auctions, multi- level marketing schemes, general merchandise and business opportunities etc. People are tempted to send money in return of great profits, but end up losing their money.
Destruction	The main motive behind these attacks is some sort of grudge. Example: Some unhappy employees attack their own organization; terrorists strike at bigger levels
Identity theft	An attacker does not steal anything from a legitimate user, instead he becomes that legitimate user! Example, it is much easier to manage to get the password of someone else's bank account or to actually be able to get a credit card on someone else's name. That privilege can be misused until it gets detected.

		Intellectual property theft	Intellectual property theft ranges from stealing companies' trade secrets, databases, digital music and videos, electronic documents and books, Identity theft, Intellectual property theft software and etc.		
		Brand theft	It is quite easy to set up fake Web sites that look like real Web sites. It is difficult for a common user to know if she is visiting the real Bank site or an attacker's site? Innocent users end up providing their secrets and personal details on these fake sites to the attackers. The attackers use these details to then access the real site, causing an identity theft.		
c)	Describe a model for Network Security and its components.			10	CO1	L2
Ans	 • The sender is the source of the original message. • It wants to transmit data securely to the intended recipient. • Message is the original plain data generated by the sender. • Before transmission, it is vulnerable to attacks if sent directly. • The sender applies a security operation such as encryption or digital signature.				

	• This transformation protects the message from unauthorized access or modification. • Secret information refers to cryptographic keys or credentials. • These keys are used in the security transformation and must remain confidential. • After applying the security transformation, the message becomes a secure message. • This encrypted or signed data is safe to transmit over an insecure network. • The information channel represents the communication medium (e.g., Internet). • It is assumed to be insecure and open to interception. • The opponent is an unauthorized entity attempting to intercept, read, modify, or disrupt the message. • The security model is designed to protect against such attackers. • The receiver applies a reverse security process such as decryption or signature verification. • This process uses the corresponding secret information. • The recipient is the intended destination of the message. • After successful transformation, the original message is recovered securely.			
Q.2 (a)	Describe security approaches and security services.	5	CO1	L1
Ans	●A trusted system is a computer system that can be trusted to a specified extent to enforce a specified security policy. ●Security models: starts from No Security; Security through Obscurity (nobody knows about its existence and contents), Host Security (the security for each host is enforced individually), Network Security(the focus is to control network access to various hosts and their services). ●Security-Management Practices: A good security policy generally takes care of four key aspects, as follows. I)Affordability: How much money and effort does this security implementation cost? ii) Functionality: What is the mechanism of providing security? iii) Cultural Issues: Does the policy complement the people's expectations, working style and beliefs) Legality: Does the policy meet the legal requirements? Security Services (X.800) ● Authentication ● Access Control ● Data Confidentiality ● Data Integrity ● Nonrepudiation.			
b)	Explain how attacks are categorized in terms of passive and active attacks.	5	CO1	L2

Ans	The attacks grouped into passive attacks and active attacks caused due to interruption problems to a hardware device, erasing program, data, or operating-system components. Passive attacks are those wherein the attacker indulges in eavesdropping or monitoring of data transmission. In other words, the attacker aims to obtain information that is in transit. The term passive indicates that the attacker does not attempt to perform any modifications to the data. Release of message contents is quite simple to understand. When you send a confidential email message to your friend, you desire that only he/she be able to access it. Otherwise, the contents of the message are released against our wishes to someone else. Using certain security mechanisms, we can prevent the release of message contents. For example, we can encode messages using a code language, so that only the desired parties understand the contents of a message, because only they know the code language. However, if many such messages are passing through, a passive attacker could try to figure out similarities between them to come up with some sort of pattern that provides her some clues regarding the communication that is taking place. Such attempts of analysing (encoded) messages to come up with likely patterns are the work of the traffic-analysis attack. Active Attacks: Unlike passive attacks, the active attacks are based on the modification of the original message in some manner, or in the creation of a false message.  Masquerade is caused when an unauthorized entity pretends to be another entity. As we have seen, user C might pose as user A and send a message to user B. User B might be led to believe that the message indeed came from user A. In masquerade attacks, an entity poses as another entity. In masquerade attacks, usually some other forms of active attacks are also embedded. As an instance, the attack may involve capturing the user's authentication sequence (e.g. user ID and password). Later, those details can be replayed to gain illegal access to the computer system.

	In a replay attack, a user captures a sequence of events, or some data units, and re-sends them. For instance, suppose user A wants to transfer some amount to user C's bank account. Both users A and C have accounts with bank B. User A might send an electronic message to bank B, requesting for the funds transfer. User C could capture this message, and send a second copy of the same to bank B. Bank B would have no idea that this is an unauthorized message, and would treat this as a second, and different, funds transfer request from user A. Therefore, user C would get the benefit of the funds transfer twice: once authorized, once through a replay attack. Alteration of messages involves some change to the original message. For instance, suppose user A sends an electronic message Transfer $1000 to D's account to bank B. User C might capture this, and change it to Transfer $10000 to C's account. Note that both the beneficiary and the amount have been changed— instead, only one of these could have also caused alteration of the message. Denial Of Service (DOS) attacks make an attempt to prevent legitimate users from accessing some services, which they are eligible for. For instance, an unauthorized user might send too many login requests to a server using random user ids in quick succession, so as to flood the network and deny other legitimate users to use the network facilities.			
(c)	Illustrate a network security model with a diagram and explanation.	10	CO1	L2
Ans	Same as Q.1(c)			
Q.3 a)	Explain the characteristics of Trojan horses, viruses and worms.	6	CO2	L1
Ans	A Trojan horse is a program with an overt (documented or known) effect and a covert (undocumented or unexpected) effect. **cp /bin/sh /tmp/.xxsh** **chmod o+s,w+x /tmp/.xxsh** In this example the overt purpose is to list the files in a directory. The covert purpose is to create a shell that is setuid to the user executing the script. Hence, this program is a Trojan horse. Trojan horses can make copies of themselves. One of the earliest Trojan horses was a version of the game animal. When this game was played, it created an extra copy of itself. These copies spread, taking up much room. The program was modified to delete one copy of the earlier version and create two copies of the modified program. Because it spread even more rapidly than the earlier version, the modified version of animal soon completely supplanted the earlier version. After a preset date, each copy of the later version deleted itself after it was played. This is a propagating Trojan horse (also called a replicating Trojan horse) is a Trojan horse that creates a copy of itself. The Trojan horse modifies the compiler to insert itself into specific programs, including future versions of the compiler itself.			
b)	Describe defences against malicious logic.	6	CO2	L2
Ans	Defending against malicious logic takes advantage of several different characteristics of malicious logic to detect, or to block, its execution. (Write any five) Malicious Logic Acting as Both Data and Instructions: Some malicious logic acts as both data and instructions. A computer virus inserts code into another program. During this writing, the object being written into the file (the set of virus instructions) is data. The			

	virus then executes itself. The instructions it executes are the same as what it has just written. Here, the object is treated as an executable set of instructions. Protection mechanisms based on this property treat all programs as type "data" until some certifying authority changes the type to "executable" (instructions). Both new systems designed to meet strong security policies and enhancements of existing systems use these methods. Malicious Logic Assuming the Identity of a User: Because a user (unknowingly) executes malicious logic, that code can access and affect objects within the user's protection domain. So, limiting the objects accessible to a given process run by the user is an obvious protection technique. Information Flow Metrics: Information is accessible only while its flow distance is less than some particular value. Reducing the Rights: The user can reduce her associated protection domain when running a suspect program. This follows from the principle of least privilege. Sandboxing: Sandboxes and virtual machines implicitly restrict process rights. A common implementation of this approach is to restrict the program by modifying it. Usually, special instructions inserted into the object code cause traps whenever an instruction violates the security policy. If the executable dynamically loads libraries, special libraries with the desired restrictions replace the standard libraries. Malicious Logic Crossing Protection Domain Boundaries by Sharing: Inhibiting users in different protection domains from sharing programs or data will inhibit malicious logic from spreading among those domains. This takes advantage of the separation implicit in integrity policies. A more general proposal suggests that programs to be protected be placed at the lowest possible level of an implementation of a multilevel security policy. Because the mandatory access controls will prevent those processes from writing to objects at lower levels, any process can read the programs, but no process can write to them. Such a scheme would have to be combined with an integrity model to provide protection against viruses to prevent both disclosure and file corruption Malicious Logic Altering Files: Mechanisms using manipulation detection codes (or MDCs) apply some function to a file to obtain a set of bits called the signature block and then protect that block. If, after recomputing the signature block, the result differs from the stored signature block, the file has changed, possibly because of malicious logic altering the file. This mechanism relies on selection of good cryptographic checksums Malicious Logic Performing Actions Beyond Specification: Fault-tolerant techniques keep systems functioning correctly when the software or hardware fails to perform to specifications. Proof-Carrying Code: Necula has proposed a technique that combines specification and integrity checking. His method, called proof-carrying code (PCC), requires a "code consumer" (user) to specify a safety requirement. The "code producer" (author) generates a proof that the code meets the desired safety property and integrates that proof with the executable code. This produces a PCC binary. The binary is delivered (through the network or other means) to the consumer. The consumer then validates the safety proof and, if it is correct, can execute the code knowing that it honors that policy. Malicious Logic Altering Statistical Characteristics: Like human languages, programs have specific statistical characteristics that malicious logic might alter. Detection of such changes may lead to detection of malicious logic.			
c)	Explain penetration studies and Vulnerability classification.	8	CO2	L2

Ans	A penetration study is a test for evaluating the strengths of all security controls on the computer system. It provides a methodology for testing the system in toto, once it is in place. Unlike other testing and verification technologies, it examines procedural and operational controls as well as technological controls. The Flaw Hypothesis Methodology was developed at System Development Corporation and provides a framework for penetration studies It consists of five steps 1. Information gathering. In this step, the testers become familiar with the system's functioning. They examine the system's design, its implementation, its operating procedures, and its use. The testers become as familiar with the system as possible. 2. Flaw hypothesis. Drawing on the knowledge gained in the first step, and on knowledge of vulnerabilities in other systems, the testers hypothesize flaws of the system under study. 3. Flaw testing. The testers test their hypothesized flaws. If a flaw does not exist (or cannot be exploited), the testers go back to step 2. If the flaw is exploited, they proceed to the next step. 4. Flaw generalization. Once a flaw has been successfully exploited, the testers attempt to generalize the vulnerability and find others like it. They feed their new understanding (or new hypothesis) back into step 2 and iterate until the test is concluded. 5. Flaw elimination. The testers suggest ways to eliminate the flaw or to use procedural controls to ameliorate it. The goal of vulnerability analysis is to develop methodologies that provide the following abilities. 1. The ability to specify, design, and implement a computer system without vulnerabilities. 2. The ability to analyze a computer system to detect vulnerabilities (which feeds into the Flaw Hypothesis Methodology step of penetration testing). 3. The ability to address any vulnerabilities introduced during the operation of the computer system (possibly leading to a redesign or reimplementation of the flawed components). 4. The ability to detect attempted exploitation of vulnerabilities.			
Q.4 a)	Define Malicious logic and its types.	6	CO2	L1
Ans	Malicious logic is a set of instructions that cause a site's security policy to be violated. Types: Trojan Horse a program with an overt (documented or known) effect and a covert (undocumented or unexpected) effect. Computer viruses: A computer virus is a program that inserts itself into one or more files and then performs some (possibly null) action.. The types of computer viruses are Boot Sector Infectors, Executable Infectors, Multipartite Viruses, terminate and stay resident (TSR) virus, Stealth Viruses, Encrypted Viruses, Polymorphic Viruses and Macro Viruses. Computer Worms: A computer worm is a program that copies itself from one computer to another. Macro worm was written in a high-level job control language, which the IBM systems interpreted. Other malicious logic: Rabbits and Bacteria, Logic Bombs.			
b)	Explain Vulnerability analysis framework with examples.	6	CO2	L2

Ans	A vulnerability classification framework is used to identify and categorize security flaws in a system. The structure of the framework depends on its goal. For example, a framework designed for attack detection focuses on how vulnerabilities are exploited, whereas a framework designed for secure software development focuses on programming and design errors that create vulnerabilities. Many frameworks classify vulnerabilities as an n-tuple, where each element represents a specific category of vulnerability. Some frameworks use a single category, while others use multiple dimensions to analyse different characteristics of vulnerabilities. One important framework is the RISOS (Research into Secure Operating Systems) framework, which was developed to help system managers and developers understand operating system security flaws. It classifies vulnerabilities into seven types. **RISOS Vulnerability Classes** 1. **Incomplete Parameter Validation** Occurs when input parameters are not properly checked before use. *Example:* Buffer overflow attacks occur when input data exceeds the allocated memory. 2. **Inconsistent Parameter Validation** Happens when different programs check input data using different formats, causing inconsistencies. *Example:* In a database, one program accepts colons or newlines in data while another interprets them as field separators, creating incorrect records. 3. **Implicit Sharing of Privileged or Confidential Data** Occurs when the operating system fails to properly isolate processes or users. *Example:* A user being able to recover another user's file password. 4. **Inadequate Identification, Authentication, or Authorization** Occurs when the system incorrectly identifies users or allows unauthorized access. *Example:* Accounts without passwords or Trojan horse programs. 5. **Violable Prohibition or Limit** Occurs when system limits or boundaries are not properly enforced. *Example:* Allowing user programs to access restricted memory locations. 6. **Exploitable Logic Errors** Security flaws caused by incorrect program logic or improper resource allocation. *Example:* Incorrect error handling leading to unintended system behaviour.			
c)	Discuss how penetration testing helps identify vulnerabilities in a system.	8	CO2	L2
Ans	Penetration testing (penetration study) is an authorized attempt to violate specific constraints defined by a system's security or integrity policy. Its purpose is to evaluate the effectiveness of all security controls — technological, procedural, and operational — by simulating attacks from an attacker's perspective. It helps identify vulnerabilities in the following ways:			

	1) A penetration study examines the system as a whole. Unlike structured design testing, it evaluates whether implemented security mechanisms enforce the stated security policy. It tests real-world behaviour rather than theoretical correctness. 2) Penetration testing begins with precise goals such as: ● Gaining unauthorized read/write access to files ● Obtaining specific privileges (e.g., administrator access) ● Disrupting system availability These clearly defined goals provide a measurable metric of success and help identify weaknesses in security mechanisms. 3) Penetration testing simulates different attacker environments: ● External attacker with no knowledge – Tests exposure and social engineering risks. ● External attacker with access – Tests password guessing, server flaws, and network vulnerabilities. ● Internal attacker with access – Tests privilege escalation and unauthorized information access. This layered approach reveals vulnerabilities at different access levels. 4) The Flaw Hypothesis Methodology was developed at System Development corporation and provides a framework for penetration studies. ● Information Gathering – Studying system design, implementation, policies, and procedures to find inconsistencies or unclear specifications. ● Flaw Hypothesis – Formulating possible weaknesses based on system knowledge and past vulnerabilities. ● Flaw Testing – Attempting controlled exploitation to verify whether flaws exist. ● Flaw Generalization – Identifying similar weaknesses and combining flaws for deeper compromise. ● Flaw Elimination – Suggesting corrective measures. This structured approach ensures systematic identification of vulnerabilities. 5) Penetration testing exposes vulnerabilities arising from: ● Poor system design ● Implementation errors ● Incomplete specifications ● Weak administrative procedures ● Excessive privilege use ● Account sharing			
Q.5 a)	Define Auditing. Explain anatomy of an auditing system.	6	CO3	L1
Ans	Auditing is the analysis of log records to present information about the system in a clear and understandable manner. Anatomy of an Auditing System: An auditing system consists of three components: the logger, the analyzer, and the notifier. These components collect data, analyze it, and report the results.			

	Logger: Logging mechanisms record information. The type and quantity of information are dictated by system or program configuration parameters. The mechanisms may record information in binary or human-readable form or transmit it directly to an analysis mechanism. A log-viewing tool is usually provided if the logs are recorded in binary form, so a user can examine the raw data or manipulate it using text-processing tools. Analyzer: An analyser takes a log as input and analyses it. The results of the analysis may lead to changes in the data being recorded to detection of some event or problem, or both. E.g. A database query control mechanism that uses prior queries to determine whether to answer contains both a logger and an analyzer. The logger records queries. When a user makes a new query, the analyser examines the answers to past queries. If there are too many answers in common, the analyser determines whether the overlap is within acceptable limits. Notifier: The analyser passes the results of the analysis to the notifier. The notifier informs the analyst, and other entities, of the results of the audit. The entities may take some action in response to these results.			
b)	Describe designing an auditing system and a posteriori design.	6	CO3	L2
Ans	The goals of the auditing process determine the type of information that must be logged in a system. Auditing is primarily intended to detect violations of security policy. Policy constraints can be represented in the form "action ⇒ condition." This means that when a specific action occurs, a corresponding condition must hold true. Auditing involves examining recorded actions and verifying whether the associated conditions are satisfied. If the action in the audit record does not match the constraint's action, the constraint is considered satisfied. However, if the constraint is false and the operation succeeds, a security violation has occurred. Thus, effective auditing requires logging relevant actions and their outcomes to ensure that policy constraints are upheld, and security breaches can be detected. A posteriori design: Needs to design auditing mechanisms for systems not built with security in mind. • Goal of auditing – Detect any violation of a stated policy • Focus is on policy and actions designed to violate policy; specific actions may not be known – Detect actions known to be part of an attempt to breach security • Focus on specific actions that have been determined to indicate attacks.			
(c)	Explain intrusion detection models and architecture of IDS.	8	CO3	L3
Ans	Intrusion detection systems determine if actions constitute intrusions on the basis of one or more models of intrusion. A model classifies a sequence of states or actions, or a characterization of states or actions, as "good" (no intrusions) or "bad" (possible intrusions). • Anomaly detection			

	– What is usual, is known – What is unusual, is bad • Misuse detection – What is bad, is known – What is not bad, is good • Specification-based detection – What is good, is known – What is not good, is bad An intrusion detection system consists of three parts. The agent corresponds to the logger. It acquires information from a target (such as a computer system). The director corresponds to the analyzer. It analyzes the data from the agents as required (usually to determine if an attack is in progress or has occurred). The director then passes this information to the notifier, which determines whether, and how, to notify the requisite entity. The notifier may communicate with the agents to adjust the log ging if appropriate. Figure 22–1 illustrates this. Hosts A, B, and C are general-purpose computers, each running an agent that monitors local activity. Host N contains a network-monitoring agent that collects data from the Internet. All agents send the gathered information to a central component called the Director. The Director analyzes the data and sends alerts or reports to a Notifier component.

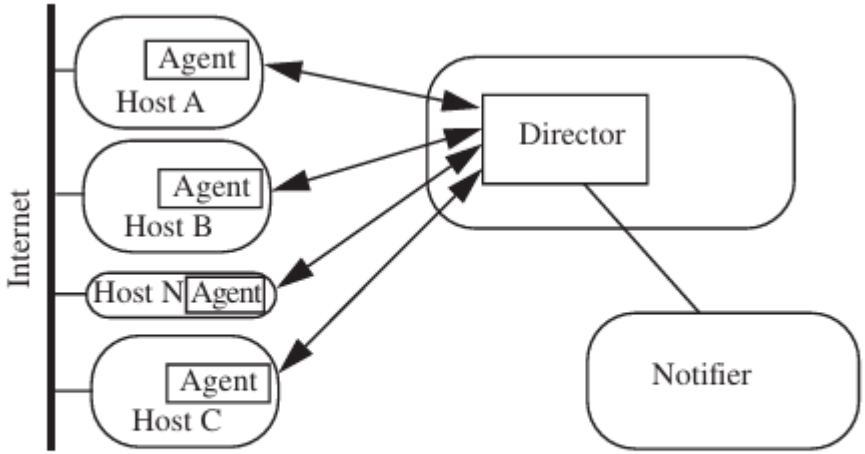


Figure 22–1 Architecture of an intrusion detection system. Hosts A, B, and C are general-purpose computers, and the agents monitor activity on them. Host N is designed for network monitoring, and its agent reports data gleaned from the Net to the director.

Q.6(a)	Explain Auditing Mechanisms with examples.	6	L1	CO3
Ans	Auditing mechanisms record system activities to detect security violations and monitor access to resources. Auditing in Secure Systems: Secure systems are designed with auditing integrated into their architecture. They provide configurable audit subsystems that allow administrators to: Monitor specific events. Track access by users (subjects). Monitor access to specific files or objects. Only relevant actions are recorded to avoid unnecessary logging. Auditing in Nonsecure Systems: In systems not designed for security, auditing is mainly used for accounting purposes. Such systems record limited details and may not capture sufficient information to detect security violations. To improve security monitoring, additional auditing			

	subsystems must be added. Example: Auditing File Systems n shared file systems like NFS, logging was added after design (a posteriori auditing), so it lacks detailed security tracking. In contrast, the Logging and Auditing File System (LAFS) was designed with auditing in mind. LAFS records user-level file actions and uses a policy language to detect violations automatically.			
b)	Discuss Intrusion Response Techniques in Network Security	6	L2	CO3
Ans	Intrusion response deals with protecting the system after an attack is detected. Its goal is to minimize damage and restore the system according to the security policy. Incident Prevention: Prevention requires detecting the attack before it completes. Intrusion detection systems (IDS) monitor activities and may automatically or manually block attacks. Early identification helps stop attacks before serious damage occurs. Intrusion Handling Phases: Intrusion handling consists of six phases: Preparation – Establish procedures and mechanisms before attacks occur. Identification – Detect and confirm the attack. Containment – Limit attacker's access and reduce damage. Eradication – Stop the attack and block similar future attacks. Recovery – Restore the system to a secure state. Follow-up – Take action against the attacker and record lessons learned.			
c)	Illustrate organization of an intrusion detection system with a diagram.	8	L3	CO3
Ans	An Intrusion Detection System (IDS) can be organized in different ways to monitor network traffic, host activities, or both. Three major paradigms are NSM, DIDS, and distributed directors. Network-Based IDS – NSM: The Network Security Monitor (NSM) monitors network traffic only. It builds a profile of normal network usage and compares current traffic with expected behavior. It uses a matrix of source, destination, and service to detect anomalies. It also supports signature-based detection for known attack patterns. Anomalies or suspicious traffic are reported to the security officer.			

	Combined Host and Network IDS – DIDS: The Distributed Intrusion Detection System (DIDS) combines: Host-based monitoring (log analysis) Network-based monitoring A centralized analysis engine (DIDS Director) Host agents and network agents send events to the DIDS Director. The Director uses an expert system (rule-based engine) to analyze events. It correlates activities across systems using a Network Identification Number (NID). Threats are classified as abuse, misuse, or suspicious acts. Distributed IDS Architecture: GrIDS extends DIDS to wide-area networks. It uses multiple directors arranged hierarchically. Each director reduces data from agents and sends summarized information upward. This improves scalability, reliability, and coordinated response.			
Q.7(a)	Explain Network Security Policies and Development.	6	L1	CO4
Ans	Network security policy defines rules to protect sensitive data and control information flow within an organization. The Drib's policy has three main goals: Keep company plans and development data confidential. Protect customer data such as credit card information. Ensure sensitive data is released only with proper authorization. To implement this, data is classified into five categories: Public Data (PD), Development Data for Existing Products (DDEP), Development Data for Future Products (DDFP), Corporate Data (CpD), and Customer Data (CuD). This classification follows the principles of least privilege and separation of privilege. Users are also classified as Outsiders, Developers, Corporate Executives, and Employees, each with specific access rights. Information flow between classes is strictly controlled. For example, moving data from DDFP to DDEP requires agreement of both developers and executives. Releasing corporate data to the public requires approval of at least two executives. Thus, network security policy development involves defining security goals, classifying data and users, and enforcing controlled information flow to protect confidentiality and integrity.			
(b)	Discuss Network Flooding and measures to anticipate attacks.	6	L2	CO4
Ans	Network flooding is a denial-of-service attack that attempts to make systems unavailable.			

	The most common example is the SYN flood, where attackers send repeated SYN requests but never complete the TCP three-way handshake. Because the final ACK is not sent, connections remain half-open. The server cannot distinguish between legitimate and attack handshakes. SYN flooding affects availability by consuming bandwidth and exhausting memory allocated for half-open connections. If memory is full, legitimate connection requests are discarded. To anticipate attacks, the Drib uses extensive logging and intrusion detection mechanisms. The DMZ log server monitors logs for known attacks and anomalous behavior. Failed attacks from the Internet are logged, but special attention is given to successful attacks or attacks within the DMZ. Both misuse detection (known signatures) and anomaly detection are used. Security staff monitor the intrusion detection system continuously. Regular log analysis and tuning of detection systems help prepare for and respond to network attacks.			
(c)	Describe System Security Policies for Networks, Users and Files	8	L2	CO4
Ans	System security policies define how networks, user accounts, and files are configured to enforce protection even if firewalls fail. Network Security Policies: In the DMZ Web server system, network access is strictly controlled. External users access the Web server only through the outer firewall proxy. Internal administrators use SSH from a trusted host through the inner firewall. All other connections are blocked and monitored. In the development system, SSH with encryption and public key authentication is required. Only necessary services (printing, logging, file access) are enabled. FTP and Web services are placed on separate servers to reduce exposure. Access control wrappers and logging provide additional protection even if firewall rules fail. User Security Policies: The DMZ Web server has minimal user accounts: A Web server user A commerce server user A system administrator Each runs with minimal privileges to reduce damage if compromised. The development system provides separate accounts for each developer. Direct root login is not allowed; administrators must log in as normal users first. User roles and groups are carefully managed to prevent privilege escalation. File Security Policies: On the DMZ Web server, system programs and configuration files are stored on CD-ROM (read-only). This prevents attackers from modifying system files. Web pages are stored on a hard drive, but CGI programs are protected on read-only media.			

	Transaction data is encrypted using public key cryptography before storage. In the development system, files and logs are backed up regularly to support recovery and investigation.			
Q.8(a)	Describe availability issues in Network Security.	6	L1	CO4
Ans	Availability ensures that network services remain accessible to users. To support availability, the network is partitioned into Internet, DMZ, and internal segments. Firewalls and proxies act as guards, controlling traffic between these segments. They enforce access control and prevent malicious traffic from disrupting services. A major availability threat is Denial of Service (DoS) and Distributed DoS (DDoS) attacks. The most common example is a SYN flood, where attackers do not complete the TCP handshake. This causes memory exhaustion by filling half-open connection queues. As a result, legitimate users cannot establish connections. Intermediate routers and filtering mechanisms can block illegitimate traffic. Techniques like SYN cookies reduce memory usage and help maintain service availability			
(b)	Explain Authentication Mechanisms in System Security.	6	L2	CO4
Ans	Authentication binds a user's identity to system processes and prevents unauthorized access. In the DMZ Web server system, SSH uses cryptographic authentication to verify that connections come only from the trusted administrative host. Public key authentication is preferred over simple IP-based verification for stronger security. If cryptographic authentication fails, password authentication is used as a fallback. The system uses PAM to allow flexible authentication methods such as smart cards without modifying core programs. Passwords are hashed using MD-5 and must meet complexity requirements. On the development network, users authenticate using reusable passwords with password aging enabled. Passwords must be changed every 90 days and are checked for strength against dictionary attacks. SSH supports public key, smart card, and password authentication, but direct root login is disabled. Administrators must first log in as normal users and then switch roles, ensuring controlled and secure authentication.			
(c)	Illustrate retrospective security techniques for files and processes	8	L2	CO4
	Retrospective security techniques help detect attacks, investigate incidents, and recover compromised systems. The Web server in the DMZ runs only a minimal set of services to reduce attack surface. Unnecessary programs are removed so attackers cannot misuse them. Most system components are stored on unalterable media to prevent modification. All administrative access is through a trusted host using SSH with public key authentication. The Web and commerce servers run with minimal privileges to limit damage. Transaction files are encrypted using public key cryptography so attackers cannot alter them. If compromised, attackers can only delete files, not modify them. Logs are maintained to track activities and support investigation. In the development system, user information is centrally maintained for consistency. Backups are performed daily to preserve files and log data. Local writable areas are also backed up to retain evidence if an attack occurs. These logging, backup, minimal service configuration, and encryption mechanisms enable detection, damage limitation, and recovery after security incidents.			
Q.9(a)	Explain user security policies for access, files and processes.	6	L1	CO4

Ans	User security policies define how users must protect their accounts, files, and running processes to maintain confidentiality and integrity. 1. Access Control Policies Use strong passwords and avoid unsafe storage of credentials. Beware of fake login prompts (Trojan login programs). Ensure login is done through trusted hosts. Do not leave sessions unattended; use screen locking mechanisms. 2. File Security Policies Set proper file permissions during file creation. Use group access carefully since group membership may change. Understand that deleting a file removes only the directory entry, not necessarily the data. Protect file confidentiality and integrity using system protection mechanisms. Improper permissions may expose sensitive data. 3. Process Security Policies Be cautious while copying or moving files, as attributes may not always be preserved. Avoid accidental overwriting of important files. Protect encryption keys and passwords, especially in multiuser systems. Ensure processes do not unintentionally expose sensitive information.			
(b)	Discuss electronic communications security for users	6	L2	CO4
Ans	Electronic communication, especially email, requires users to follow strict security precautions because messages often come from untrusted sources. 1. Email from Untrusted Sources Electronic mail may pass through firewalls, but filtering cannot detect all malicious content. Users must assume that email contents and attachments are not trustworthy. Mail programs should be configured not to automatically execute attachments or scripts. 2. Automated Email Processing Risks Some systems automatically process incoming email and execute commands. This is dangerous because malicious content may cause unintended program execution and side effects. Users must avoid automatic execution of email content. 3. Digital Signatures and Certificates Electronic signatures may appear valid, but certificates can be: Expired, Compromised and Invalid. Users must verify certificates properly and not blindly trust signed messages.			
(c)	Describe Program Security Design, Refinement and Implementation	8	L2	CO4
Ans	The program is designed using a modular structure, where each module implements a specific requirement. The framework includes User interface, Access control module, Privilege management and Command execution.			

	The program supports: Restricted access (specific command execution), Unrestricted access (full command interpreter). Access decisions are based on user identity, location, time, and requested command. 2. Refinement of Design First-Level Refinement: The high-level algorithm is converted into structured pseudocode. The design follows block-structured programming suitable for UNIX-like systems. Second-Level Refinement: The pseudocode is mapped to C language implementation on Linux. Users and roles are represented internally using integer IDs. Commands are represented as arrays of strings. Access control data is read from a protected file. This ensures consistency and secure privilege handling. 3. Implementation Details The program collects user, role, location, and time information. It checks access control records stored in a file editable only by root. On success, it changes process privileges to the role account. On failure or error, access is denied and logged (fail-safe default). Proper error handling prevents security bypass due to malformed records.			
Q.10(a)	Explain User Access Control Policies with Examples	6	L1	CO4
	User access control policies define how users gain access to accounts and how they must protect that access from misuse or attacks. These policies focus on login security, session control, file permissions, and device protection. 1. Password Policy Passwords should be strong and difficult to guess. Reusable passwords can be captured using Trojan login programs. Example: An attacker places a fake login program that captures username and password before starting the real login session. 2.Secure Login Procedure Users must provide login name and authentication information. Systems should prevent fake login prompts. Lack of mutual authentication allows attackers to impersonate the system. Example: A Trojan login screen records the user's credentials and then starts a genuine session without the user realizing the attack. 3. Session Control Policy Users must not leave sessions unattended. Screen locking programs should be used when stepping away. Example: If a user leaves the system logged in, an unauthorized person can execute commands in that user's name. 4. File Access Control			

	Users must set proper file permissions to maintain confidentiality and integrity. Default permissions on file creation should be carefully defined. Group access must be managed properly. Example:Improper group permissions may allow unauthorized users to read confidential files.			
(b)	Describe program security requirements and policy	6	L2	CO4
Ans	Program security requirements define how a system program must control access, protect accounts, and enforce security policies. 1. Access Based on Identity and Conditions. Instead of sharing passwords for role accounts (like root), a special program verifies: User identity, Location of access, Time of request. Only if all conditions are satisfied, access is granted. Requirement: Access to a role account must depend on user, location, and time. 2.. Protection of Access Control Information Only the highest privileged account (root) can modify access control information. This prevents users from granting themselves unauthorized privileges. Requirement: Only root can alter access permissions for role accounts. 4. Restricted and Unrestricted Access The program must support two types of access: Unrestricted access: Full command interpreter (complete control). Restricted access: Only specified commands are allowed. This follows the principle of least privilege and reduces misuse. 5. Protection of Role Account Files Files and directories owned by role accounts must be accessible only to: Authorized users, Trusted system installers, Root.			
(c)	Illustrate Secure Program Design with Refinements and Implementation Steps	8	L3	CO4
Ans	Secure program design uses a modular framework where the user interface and internal modules work together to meet security requirements. The interface supports two modes: unrestricted access to a role account or restricted execution of a specific command from that role. The program first obtains role, user identity, location (host/terminal), time, and requested command.An access-checking module compares these values with access control records and returns success or failure. By fail-safe defaults, any error or mismatch results in denial of access, and the attempt is logged. If access is allowed, the program fetches the role's user/group information and changes the process privileges to the role account. For restricted access, the child process is overlaid with a command interpreter that spawns only the allowed command; otherwise			

	a normal shell is started. Refinement step 1 converts the high-level design into block-structured pseudocode suitable for UNIX-like systems. Refinement step 2 maps the pseudocode to C on Linux, using integer IDs for users/roles and representing commands as an array of strings. Implementation also includes secure storage of access-control data in a file editable only by root, safe retrieval of location information, and strict error handling to prevent unsafe access.