R.M.K GROUP OF ENGINEERING INSTITUTIONS

R.M.K
GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

Please read this disclaimer before proceeding:

22CY701
INTRUSION DETECTION AND INTERNET SECURITY
(Lab Integrated)

UNIT I
INTRODUCTION TO INTRUSION DETECTION

Department : **CSE(CS)**

Batch/Year : **2022 - 2026 /IV**

Created by :**Dr. Dharini N**

Date : **10.06.2025**

Table of Contents

Course Objectives

22CY701 INTRUSION DETECTION AND INTERNET SECURITY

(Lab Integrated)

COURSE OBJECTIVES

- To Understand when, where, how, and why to apply Intrusion Detection tools and techniques in order to improve the security posture of an enterprise.
- To Apply knowledge of the fundamentals and history of Intrusion Detection in order to avoid common pitfalls in the creation and evaluation of new Intrusion Detection Systems
- To Analyze intrusion detection alerts and logs to distinguish attack types from false alarms
- To Understand the fundamentals of network security, including the MAC layer, Internet Protocol, and common attacks targeting these layers.
- To Gain an understanding of key internet security mechanisms, including firewalls, virtual private networks (VPNs), and TLS/SSL VPNs.

Prerequisite

22CY701 INTRUSION DETECTION AND INTERNET SECURITY

PREREQUISITE

1. 22CY401-CYBER SECURITY ESSENTIALS
2. 22CS501 – COMPUTER NETWORKS
3. 22CS901 – ETHICAL HACKING

Syllabus

22CY701 – INTRUSION DETECTION AND INTERNET SECURITY (Lab Integrated)

SYLLABUS **3 0 2 4**

UNIT I INTRODUCTION TO INTRUSION DETECTION

History of Intrusion detection, Audit, Concept and definition, Internal and external threats to data, attacks, Need and types of IDS, Information sources Host based information sources, Network based information sources.

List of Exercise/Experiments

1. Install Snort and configure it to monitor network traffic.

2. Deploy Snort as a Network Intrusion Detection System (NIDS).

UNIT II INTRUSION DETECTION AND PREVENTION TECHNIQUES

Intrusion Prevention Systems, Network IDs protocol based IDs , Hybrid IDs, Analysis schemes, thinking about intrusion. A model for intrusion analysis, techniques Responses requirement of responses, types of responses mapping responses to policy Vulnerability analysis, credential analysis non credential analysis

List of Exercise/Experiments

1. Write and implement custom Snort rules to detect specific traffic patterns.

2. Integrate Snort with MySQL to log alerts to a database.

UNIT III SNORT

Introduction to Snort, Snort Installation Scenarios, Installing Snort, Running Snort on Multiple Network Interfaces, Snort Command Line Options. Step-By-Step Procedure to Compile and Install Snort Location of Snort Files, Snort Modes Snort Alert Modes-Working with Snort Rules, Rule Headers, Rule Options, The Snort Configuration File etc. Plugins, Preprocessors and Output Modules, Using Snort with MySQL

List of Exercise/Experiments

1.Enhance Snort's functionality using preprocessors and plugins.

2.Set up advanced alerting and logging mechanisms.

UNIT IV ESSENTIALS OF INTERNET SECURITY

Network Security basics-The MAC Layer and Attacks- The Internet Protocol and Attacks- Packet Sniffing and Spoofing-Attacks on TCP Protocol- DNS Attacks Overview - Local DNS Cache Poisoning Attack- Remote DNS Cache Poisoning attack- Replay forgery attacks-DNS Rebinding attack- DoS on DNS Servers- DNSSEC-Securing DNS

List of Exercise/Experiments

1.Conducting TCP SYN Flood Attack

2.Sniffing Packets on Network Interfaces

3.Spoofing Source IP Address in Packets

4.Investigating DNSSEC Implementation and Validation

UNIT V INTERNET SECURITY MECHANISMS

Firewall-Virtual Private Network-Overview of How TLC/SSL VPN Works- Creating and using the TUN Interface- Implementing the IP Tunnel- Testing VPN- Tunneling and Firewall Evasion- -BGP and Attacks- The Heartbleed bug and attack- Reverse Shell

List of Exercise/Experiments

1. Configuring Linux Firewall using IP tables
2. Setting Up VPN Tunnels
3. Exploring BGP Session Hijacking
4. Simulating Heartbleed Attack Scenario

Course Outcomes

R.M.K GROUP OF INSTITUTIONS

COURSE OUTCOMES

- CO1: Understand fundamental concepts and demonstrate skills in capturing and analyzing network packets.
- CO2: Utilize various protocol analyzers and Network Intrusion Detection Systems (NIDS) to detect network attacks and troubleshoot network problems.
- CO3: Develop the ability to proficiently use the Snort tool for detecting and mitigating network attacks
- CO4: Demonstrate knowledge of network security basics, including MAC layer vulnerabilities and attacks, as well as common attacks targeting the Internet Protocol.
- CO5: Demonstrate understanding of firewall, VPN, and TLS/SSL VPN principles and functionalities in network security.
- CO6: Apply the concepts of Intrusion Detection and internet security protocols to develop cyber security mechanisms.

CO – PO/ PSO Mapping

R.M.K GROUP OF INSTITUTIONS

CO-PO MAPPING

COs	PO's/PSO's														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	3	3	1	2	2	–	–	–	–	1	–	2	2	3	1
CO2	3	3	1	2	2	–	–	–	–	1	–	2	2	3	2
CO3	2	2	2	1	3	–	–	1	1	2	1	2	1	3	3
CO4	3	2	1	1	1	–	–	2	–	1	–	3	1	3	1
CO5	3	2	1	1	2	1	–	2	1	2	1	2	1	3	2
CO6	2	2	3	2	3	1	–	2	2	2	2	3	3	3	3

1 – Low, 2 – Medium, 3 – Strong

Lecture Plan

R.M.K
GROUP OF
INSTITUTIONS

LECTURE PLAN

S No	Topics	No of periods	Proposed date	Actual Lecture Date	Pert aining CO	Taxo nomy level	Mode of delivery
1	History of Intrusion detection	1	14.07.2025		CO1	K1	ICT Tools
2	Audit, Concept and definition	1	15.07.2025		CO1	K2	ICT Tools
3	Audit, Concept and definition	1	16.07.2025		CO1	K3	ICT Tools
4	Internal and external threats to data, attack	1	17.07.2025		CO1	K2	ICT Tools
5	Internal and external threats to data, attack	1	18.07.2025		CO1	K2	ICT Tools
6	Need and types of IDS	1	19.07.2025		CO1	K2	ICT Tools
7	Need and types of IDS	1	21.07.2025		CO1	K2	ICT Tools
8	Information sources Host based information sources	1	22.07.2025		CO1	K2	ICT Tools
9	Network based information sources.	1	23.07.2025		CO1	K3	ICT Tools
10	Install Snort and configure it to monitor network traffic.	1	24.07.2025		CO1	K3	Lecture and Practical
11	Install Snort and configure it to monitor network traffic.	1	25.07.2025		CO1	K3	Lecture and Practical
12	Install Snort and configure it to monitor network traffic.	1	26.07.2025		CO1	K3	Lecture and Practical

13	Deploy Snort as a Network Intrusion Detection System (NIDS).	1	29.07.2025		CO1	K3	Lecture and Practical
14	Deploy Snort as a Network Intrusion Detection System (NIDS).	1	30.07.2025		CO1	K3	Lecture and Practical
15	Deploy Snort as a Network Intrusion Detection System (NIDS).	1	01.08.2025		CO1	K3	Lecture and Practical

Activity Based Learning

Collaborative Concept Mapping + Structured Debate

Host-based IDS is more effective than Network-based IDS in modern systems

R.M.K GROUP OF INSTITUTIONS

Lecture Notes

R.M.K GROUP OF INSTITUTIONS

UNIT I

Intrusion detection systems (IDSs) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for signs of security problems. As network attacks have increased in number and severity over the past few years, intrusion detection systems have become a necessary addition to the security infrastructure of most organizations.

1. History of Intrusion Detection Systems

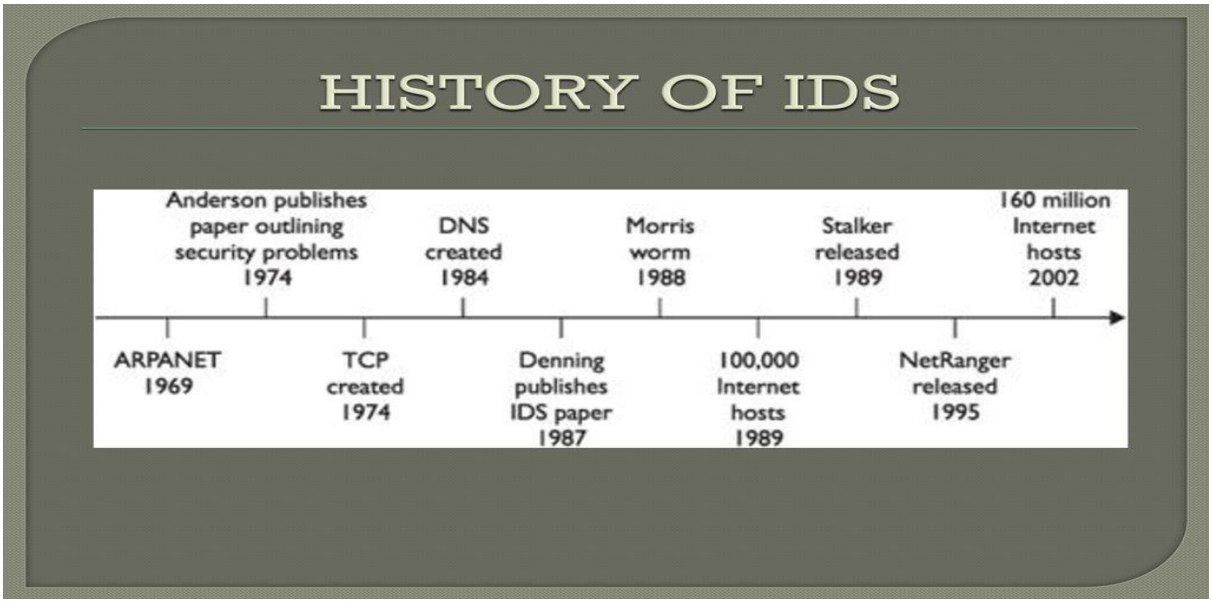
Originally, system administrators performed intrusion detection by sitting in front of a console and monitoring user activities. They might detect intrusions by noticing, for example, that a vacationing user is logged in locally or that a seldom-used printer is unusually active. Although effective enough at the time, this early form of intrusion detection was ad hoc and not scalable.

The next step in intrusion detection involved audit logs, which system administrators reviewed for evidence of unusual or malicious behavior. In the late '70s and early '80s, administrators typically printed audit logs on fan-folded paper, which were often stacked four- to five-feet high by the end of an average week. Searching through such a stack was obviously very time-consuming. With this overabundance of information and only manual analysis, administrators mainly used audit logs as a forensic tool to determine the cause of a particular security incident after the fact. There was little hope of catching an attack in progress.

As storage became cheaper, audit logs moved online and researchers developed programs to analyze the data. However, analysis was slow and often computationally intensive and therefore intrusion detection programs were usually run at night when the system's user load was low. Therefore, most intrusions were still detected after they occurred.

In the early '90s, researchers developed real-time intrusion detection systems that reviewed audit data as it was produced. This enabled the detection of attacks and attempted attacks as they occurred, which in turn allowed for real-time response, and, in some cases, attack preemption.

More recent intrusion detection efforts have centered on developing products that users can effectively deploy in large networks. This is no easy task, given increasing security concerns, countless new attack techniques, and continuous changes in the surrounding computing environment.



History of IDS

The "History of IDS" diagram outlines the chronological development of Intrusion Detection Systems, highlighting key milestones in network security evolution. It begins in 1969 with the creation of ARPANET, the precursor to the modern internet. In 1974, the TCP protocol was developed, laying the groundwork for network communication. That same year, James P. Anderson published a paper identifying key computer security issues, which is considered one of the earliest conceptual frameworks for IDS. In 1984, the Domain Name System (DNS) was created, adding complexity to internet infrastructure and expanding the attack surface. Dorothy Denning's influential 1987 paper introduced a formal intrusion detection model, providing a strong theoretical foundation. The 1988 release of the Morris

worm—a major early malware incident—highlighted the need for real-time detection tools. In 1989, the commercial IDS tool "Stalker" was released, and the internet had grown to 100,000 hosts, showing the need for scalable security. The 1995 release of NetRanger, a network-based IDS, marked a leap in real-time threat detection capabilities. Finally, by 2002, the number of internet hosts had exploded to 160 million, emphasizing the importance of IDS in securing vast digital infrastructures. This timeline reflects how IDS evolved in response to technological growth, security research, and real-world cyber threats.

Intrusion Detection Overview:

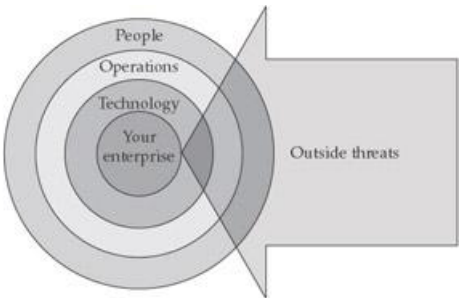
The goal of intrusion detection is seemingly simple: to detect intrusions. However, the task is difficult, and in fact intrusion detection systems do not detect intrusions at all—they only identify evidence of intrusions, either while they're in progress or after the fact.

Such evidence is sometimes referred to as an attack's "manifestation." If there is no manifestation, if the manifestation lacks sufficient information, or if the information it contains is untrustworthy, then the system cannot detect the intrusion.

For example, suppose a house monitoring system is analyzing camera output that shows a person fiddling with the front door. The camera's video data is the manifestation of the occurring intrusion. If the camera lens is dirty or out of focus, the system will be unable to determine whether the person is a burglar or the owner.

It is also important to note that IDSs and IPSs are just two of many methods that should be employed in a strong security program. Using a layered approach, or defense in depth, based on careful risk analysis is critical in any information protection program because a network is only as secure as its weakest link. This means that a network should have multiple layers of security, each with its own function, to complement the overall security strategy of the organization. Figure below illustrates a defense-in-

depth approach that will protect a network on many levels.



Defense in depth

IDSs work at the network layer of the OSI model, and passive network sensors are typically positioned at choke points on the network. They analyse packets to find specific patterns in network traffic—if they find such a pattern in the traffic, an alert is logged, and a response can be based on the data recorded. IDSs are similar to antivirus software in that they use known signatures to recognize traffic patterns that may be malicious in intent.

Layer	Function	Protocols
Application (user interface)	This layer is used for applications, such as HTTP, specifically written to run over the network and allows accesses to network services. It handles issues like network transparency, resource allocation, and problem partitioning. The application layer is concerned with the user's view of the network, like formatting. In addition, this layer allows access to services that support applications and handle network access, flow, and recovery.	DNS, FTP, TFTP, BOOTP, SNMP, RLOGIN, SMTP, MIME, NFS, FINGER, TELNET, APPC, AFP,
Presentation (translation)	The presentation layer helps to translate between the application and the network formats. This is also where protocol conversion takes place.	Named Pipes, Mail Slots, RPC, NCP, SMB
Session	The session layer helps to establish, maintain, and end sessions across the network.	NetBios
Transport (packets; flow control and error-handling)	The transport layer manages the flow control of data between parties across the network.	TCP, ARP, RARP, SPX, NWLink, ATP, NetBEUI
Network (addressing; routing)	The network layer translates logical network addresses and names to their physical addresses and is responsible for addressing and managing network problems such as packet switching, data congestion, and routing.	IP, ARP, RARP, ICMP, RIP, OSFP, IGMP, IPX, NWLink, OSI, DDP, DECnet
Data link (data frames to bits)	The data-link layer turns packets into raw bits on the sending end, and at the receiving end turns bits into packets. It handles data frames between the network and physical layers.	
Physical (hardware; raw bit stream)	The physical layer transmits the raw bit stream over the physical cable or airwaves (when dealing with wireless). It defines cables, cards, and other physical aspects.	IEEE 802, IEEE 802.2, ISO 2110, ISDN

Data collection issues

For accurate intrusion detection, we must have reliable and complete data about the target system's activities. Reliable data collection is a complex issue in itself. Most operating systems offer some form of auditing that provides an operations log for different users. These logs might be limited to the security-relevant events (such as failed login attempts) or they might offer a complete report on every system call invoked by every process. Similarly, routers and firewalls provide event logs for network activity. These logs might contain simple information, such as network connection openings and closings, or a complete record of every packet that appeared on the wire.

The amount of system activity information a system collects is a trade-off between overhead and effectiveness. A system that records every action in detail could have substantially degraded performance and require enormous disk storage. For example, collecting a complete log of a 100-Mbit Ethernet link's network packets could require hundreds of Gigabytes per day.

Collecting information is expensive, and collecting the right information is important. Determining what information to log and where to collect it is an open problem. For example, having your house alarm system monitor the water for pollution levels is an expensive activity that doesn't help detect burglars. On the other hand, if the house's threat model includes terrorist attacks, monitoring the pollution level might be reasonable.

2. **IDS Audit**

An IDS Audit refers to the process of systematically reviewing and analyzing the logs, alerts, and data generated by an Intrusion Detection System (IDS) to assess its effectiveness, accuracy, and compliance with security policies. The audit helps ensure that the IDS is correctly detecting intrusions, minimizing false positives/negatives, and providing actionable security insights.

Purpose of IDS Audit

Verify IDS Effectiveness:

To check if the IDS correctly identifies real attacks or malicious activities on the network or host systems.

Reduce False Positives and False Negatives:

False positives are benign activities mistakenly flagged as intrusions, while false negatives are actual intrusions that the IDS misses. An audit helps tune the IDS rules and algorithms to minimize these.

Compliance and Accountability:

Many industries and organizations have regulatory or policy requirements that mandate regular security audits, including IDS audits.

Incident Investigation:

IDS audit logs are crucial during forensic investigations after a security incident to understand the attack vector, timeline, and impact.

Performance Improvement:

Helps in identifying performance bottlenecks or gaps in IDS coverage.

IDS audit include

1. **Review of IDS Logs and Alerts:**
   - Analyze recorded intrusion events.
   - Check patterns and frequency of alerts.
   - Validate whether alerts correspond to real security events.
2. **Evaluation of IDS Configuration:**
   - Verify that IDS sensors are properly placed in the network for optimal coverage.
   - Check detection rules and signatures for relevance and currency.
   - Assess whether the IDS is updated regularly with latest threat signatures.
3. **Assessment of IDS Response:**
   - Check how alerts are handled — are they escalated properly?
   - Evaluate the workflow for incident response triggered by IDS alerts.

4. **Analysis of False Positives/Negatives:**
   - Identify events that were wrongly classified.
   - Understand the reasons and update IDS signatures or tuning parameters accordingly.
5. **System Performance Review:**
   - Examine if the IDS can handle traffic loads without dropping packets.
   - Assess resource utilization (CPU, memory).
6. **Compliance Verification:**
   - Ensure audit trails meet organizational or regulatory standards.
   - Check retention periods and access controls for IDS logs.

Types of IDS Audit

- **Manual Audit:** Security analysts manually review IDS logs and reports.
- **Automated Audit:** Using audit tools and software that parse IDS data, detect anomalies, and generate audit reports.
- **Hybrid Audit:** Combines automated tools with human review for greater accuracy.

Tools and Techniques Used in IDS Audit

- **Log Analysis Tools:** Tools like Splunk, ELK Stack (Elasticsearch, Logstash, Kibana), or proprietary SIEM (Security Information and Event Management) systems to aggregate and analyze IDS data.
- **Statistical Analysis:** Identifying trends and patterns in IDS alerts to detect abnormal behavior.
- **Signature and Rule Review:** Regularly checking the IDS detection rules for outdated or missing signatures.
- **Simulation Testing:**

Running penetration tests or simulated attacks to check if IDS detects them.

Benefits of IDS Audit

- Ensures IDS is effective and reliable.
- Improves overall security posture by identifying vulnerabilities.
- Helps in proactive threat detection and faster incident response.
- Maintains compliance with security standards and regulations.
- Reduces unnecessary alert noise, saving analyst time and effort.

Challenges in IDS Audit

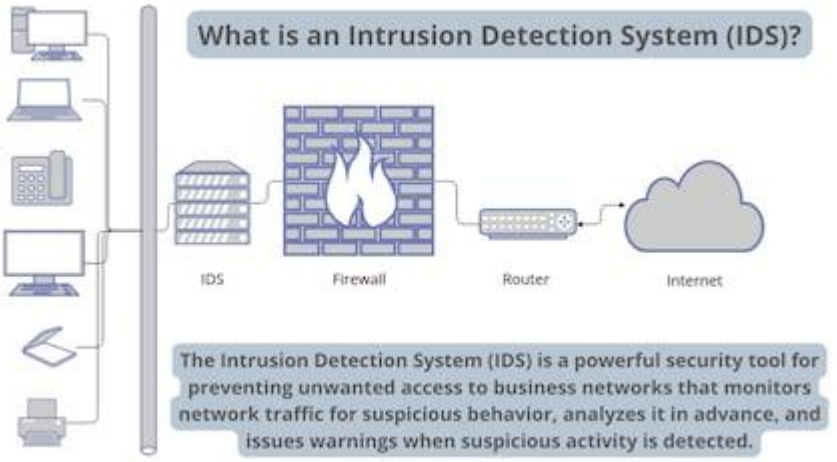
- High volume of data and alerts to analyze.
- Distinguishing between real attacks and benign anomalies.
- Keeping IDS signatures and rules updated.
- Balancing detection sensitivity to avoid excessive false alarms.
- Skilled personnel required to interpret audit findings correctly.

An **IDS Audit** is a critical security activity focused on reviewing and validating the operation and output of an Intrusion Detection System. By analyzing logs, configurations, and performance, organizations can ensure their IDS effectively protects against threats, maintains compliance, and provides useful insights for incident handling. Regular IDS audits improve detection accuracy, reduce false alarms, and strengthen overall cybersecurity defenses.

3. **IDS Concept and Definition**

Security Intrusion: A security event, or a combination of multiple security events, that constitutes a security incident in which an intruder gains, or attempts to gain, access to a system (or system resource) without having authorization to do so.

Intrusion Detection: A security service that monitors and analyzes system events for the purpose of finding, and providing real-time or near real-time warning of, attempts to access system resources in an unauthorized manner.



Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network. Intrusions are caused by attackers accessing the systems from the Internet, authorized users of the systems who attempt to gain additional privileges for which they are not authorized, and authorized users who misuse the privileges given them. Intrusion Detection Systems (IDSs) are software or hardware products that automate this monitoring and analysis process.

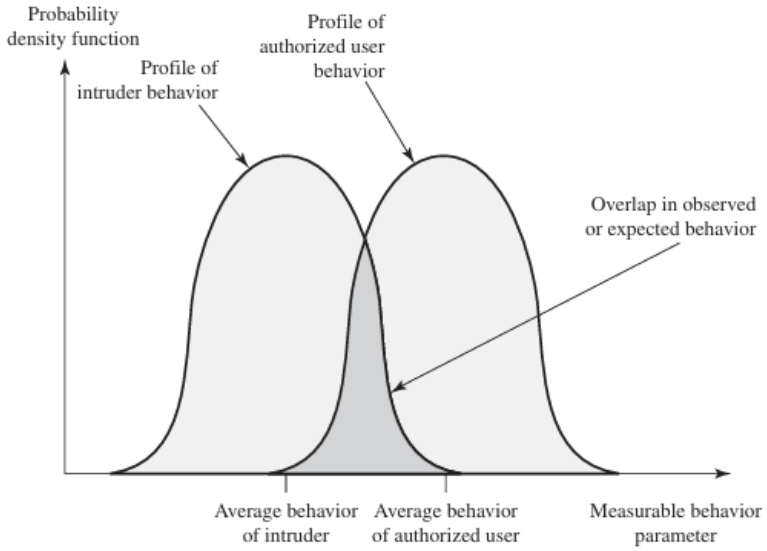
Authentication facilities, access control facilities, and firewalls all play a role in countering intrusions. Another line of defense is intrusion detection, and this has been the focus of much research in recent years. This interest is motivated by a number of considerations, including the following:

1. If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised. Even if the detection is not sufficiently timely to preempt the intruder, the sooner that the intrusion is detected, the less the amount of damage and the more quickly that recovery can be achieved.

2. An effective IDS can serve as a deterrent, thus acting to prevent intrusions.

3. Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen intrusion prevention measures. Intrusion detection is based on the assumption that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified. Of course, we cannot expect that there will be a crisp, exact distinction between an attack by an intruder and the normal use of resources by an authorized user. Rather, we must expect that there will be some overlap.

Figure below suggests, in abstract terms, the nature of the task confronting the designer of an IDS. Although the typical behavior of an intruder differs from the typical behavior of an authorized user, there is an overlap in these behaviors. Thus, a loose interpretation of intruder behavior, which will catch more intruders, will also lead to a number of false positives , or authorized users identified as intruders. On the other hand, an attempt to limit false positives by a tight interpretation of intruder behavior will lead to an increase in false negatives , or intruders not identified as intruders. Thus, there is an element of compromise and art in the practice of intrusion detection.



Profiles of Behavior of Intruders and Authorized Users

It was postulated that one could, with reasonable confidence, distinguish between a masquerader and a legitimate user. Patterns of legitimate user

behavior can be established by observing past history, and significant deviation from such patterns can be detected. Anderson suggests that the task of detecting a misfeasor (legitimate user performing in an unauthorized fashion) is more difficult, in that the distinction between abnormal and normal behavior may be small. Anderson concluded that such violations would be undetectable solely through the search for anomalous behavior. However, misfeasor behavior might nevertheless be detectable by intelligent definition of the class of conditions that suggest unauthorized use. Finally, the detection of the clandestine user was felt to be beyond the scope of purely automated techniques. These observations, which were made in 1980, remain true today.

Requirements of IDS

- Run continually with minimal human supervision.
- Be fault tolerant in the sense that it must be able to recover from system crashes and reinitializations.
- Resist subversion. The IDS must be able to monitor itself and detect if it has been modified by an attacker.
- Impose a minimal overhead on the system where it is running.
- Be able to be configured according to the security policies of the system that is being monitored.
- Be able to adapt to changes in system and user behavior over time.
- Be able to scale to monitor a large number of hosts.
- Provide graceful degradation of service in the sense that if some components of the IDS stop working for any reason, the rest of them should be affected as little as possible.
- Allow dynamic reconfiguration; that is, the ability to reconfigure the IDS without having to restart it.

IDS Architecture

There exist many IDSs, based on different conceptual frameworks. It is

still possible, however, to recognize a common architecture that underlies all intrusion detection systems. We will present the main components of IDSs and their functionality. To this end, we use the terminology introduced by the working group on the Common Intrusion Detection Framework (CIDF)

CIDF models an IDS as an aggregate of four components with specific roles:

1. *Event boxes (E-boxes).* The role of event boxes is to generate events by processing raw audit data produced by the computational environment. A common example of an E-boxes is a program that filters audit data generated by an operating system evaluated at the C2 level of TCSEC [U.S. Department of Defense, 1985]. Another example is a network sniffer that generates events based on the network traffic.
2. *Analysis boxes (A-boxes).* The role of an analysis box is to analyze the events provided by other components. The results of the analysis are sent back to the system as additional events, typically representing alarms. Usually A-boxes analyze simple events supplied by E-boxes. Some A-boxes analyze events produced by other A-boxes and operate at a higher level of abstraction.
3. *Database boxes (D-boxes).* Database boxes simply store events, guaranteeing persistence and allowing postmortem analysis.
4. *Response boxes (R-boxes).* Response boxes consume messages that carry directives about actions to be performed as a reaction to a detected intrusion. Typical actions include killing processes, resetting network connections, and modifying firewall settings.

Figure below presents an IDS system where two E-boxes monitor the environment and deliver audit events to two A-boxes. These A-boxes analyze the audit data and provide their conclusions to a third A-box that correlates the alerts, stores them in a D-box and controls an R-box.

Dashed lines are used to indicate exchange of events, solid lines represent the exchanging of raw audit data. Note that components are logical entities which produce or consume events. The CIDF model does not mandate what their implementation should be. It only states their roles and interactions. They can be realized as a single process on a single computer or as a collection of cooperating processes spanning multiple computers.

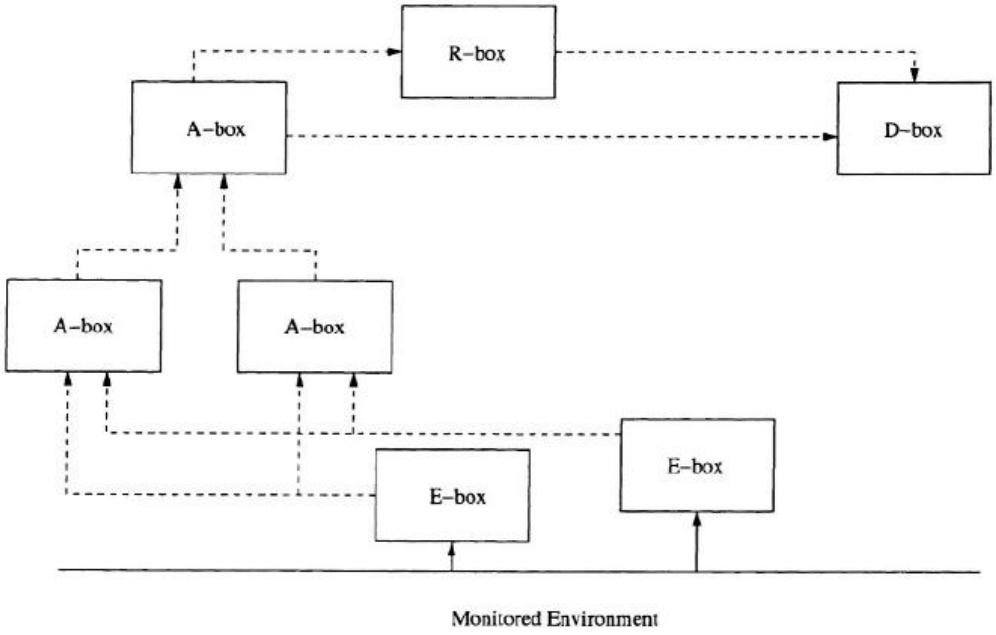


Figure 2.3. CIDF Description of an IDS System

Taxonamy

Intrusion detection systems may be classified according to different characteristics

Detection method. It defines the philosophy on which the A-box is built. Two approaches have been proposed. When the IDS defines what is "normal" in the environment and flags as attacks deviations from normality, it is qualified as ***anomaly-based.*** When the IDS explicitly defines what is "abnormal", using specific knowledge about the attacks in order to detect them, it is called ***misuse-based.***

Behavior on detection. It defines a characteristic of the R-box. It is said to be passive if the system just issues an alert when an attack is detected. If more proactive actions are taken (e.g., disconnecting users, shutting down network connections), it is said to be active.

Audit source location. It specifies where the E-box takes audit data from. We distinguish between **host-based IDSs**, which deal with audit data generated on a single host, e.g., a C2 audit trail; **application-based IDSs,** which work on audit records produced by a specific application; and **network-based IDSs,** which monitor network traffic.

Usage frequency. It discriminates between systems that analyze the data in real time and those that are run periodically (offline). It specifies how often the A-box analyzes data collected by other parts of the system.

4. **Internal and external threats to data**

INTERNAL THREATS

Internal threats originate from individuals who have or have had authorized access to the network. This could be a disgruntled employee, an opportunistic employee, or an unhappy past employee whose access is still active. In the case of a past network employee, even if their account is gone, they could be using a compromised account or one they set up before leaving for just this purpose. Many surveys and studies show that internal attacks can be significant in both the number and the size of any losses. If dishonest employees steal inventory or petty cash, or set up elaborate paper-invoicing schemes, why wouldn't they learn to use the computer systems to further their ambitions? With access to the right systems, a trusted employee can devastate an unsuspecting organization. All too often, employers fail to prosecute this type of activity. The reasons range from fear of the activity becoming public knowledge to knowing that, quite often, record-keeping systems haven't been developed either to provide adequate evidence or to prove that the transactions, no matter how ludicrous, weren't authorized.

EXTERNAL THREATS

External threats are threats from individuals outside the organization, often using the Internet or dial-up access. These attackers don't have authorized access to

the systems. In trying to categorize a specific threat, the result could possibly be a combination of two or more threats. The attack might be structured from an external source, but a serious crime might have one or more compromised employees on the inside actively furthering the endeavor.

UNSTRUCTURED THREATS

Unstructured threats often involve unfocused assaults on one or more network systems, often by individuals with limited or developing skills. The systems being attacked and infected are probably unknown to the perpetrator. These attacks are often the result of people with limited integrity and too much time on their hands. Malicious intent might or might not exist, but there is always indifference to the resulting damage caused to others. The Internet has many sites where the curious can select program codes, such as a virus, worm, or Trojan horse, often with instructions that can be modified or redistributed as is. In all cases, these items are small programs written by a human being. They aren't alive and they can't evolve spontaneously from nothing.The person launching an unstructured attack is often referred to as a script kiddy because that person often lacks the skills to develop the threat themselves, but can pass it on anonymously (they think) and gain some perverse sense of satisfaction from the result. E-mail delivery methods have replaced —shared‖ game disks as the vehicle of choice for distributing this type of attack. The term —script kiddy is a common derogatory term and should be used with caution, if at all. Script kiddy is included here so you know what it means. Remember, the difference between an unstructured attack and a series of all-out denial-of-service attacks might be that the latter attacker is offended or angry. Unstructured attacks involving code that reproduces itself and mails a copy to everyone in the person's e-mail address book can easily circle the globe in a few hours, causing problems for networks and individuals all over the world. While the original intent might have been more thoughtless than malicious, the result can be a loss of user access while systems are being protected, a loss of reputation if the news that a company's site has been attacked, or a loss of user freedoms as more-restrictive policies and practices are implemented to defend against additional attacks. In some organizations, if the network is down, entire groups of people can't do their jobs, so they're either sent home or they sit and wait without pay because their

income is tied to sales. So even if the hacker —thought‖ no one would be hurt, the result is often that they just beat some single parent or new hire out of a day's pay. Each of these results can be quantified in currency and often result in large numbers if and when the perpetrator is prosecuted.

STRUCTURED THREATS

A **structured threat** refers to a highly organized and targeted cyberattack carried out by skilled individuals or groups with a specific objective in mind. Unlike random or opportunistic attacks, structured threats follow a systematic and well-planned approach, often involving multiple stages such as reconnaissance, weaponization, delivery, exploitation, installation, command and control, and execution of the intended objective. These threats are commonly associated with cybercriminal organizations, hacktivist groups, or nation-state actors who use advanced tools, custom malware, and zero-day vulnerabilities to breach security systems. Structured threats are particularly dangerous because they are stealthy, persistent, and difficult to detect using traditional security measures. A well-known example of a structured threat is the Stuxnet worm, which specifically targeted Iran's nuclear facilities using multiple zero-day exploits and a sophisticated delivery mechanism. Other examples include Advanced Persistent Threat (APT) campaigns like APT29 (Cozy Bear), which engage in long-term cyber espionage, and targeted ransomware attacks such as those launched by the Maze group against corporate networks. The danger posed by structured threats lies in their ability to evade detection, remain within systems for extended periods, and cause extensive damage—ranging from data theft to operational disruption. Defending against such threats requires a combination of advanced intrusion detection systems, endpoint protection, regular vulnerability assessments, security awareness training, and the integration of threat intelligence to anticipate and mitigate potential attacks.

Category	Description	Example
Internal Threat	Comes from inside the organization	Employee stealing data
External Threat	Comes from outside	Hacker phishing

Category	Description	Example
	the organization	attack
Structured Attack	Well-planned and targeted	Nation-state ransomware
Unstructured Attack	Random and unskilled	Amateur trying DDoS tools

Intruders

· **Masquerader:** An individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account

· **Misfeasor:** A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges

· **Clandestine user:** An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection

Examples of intrusion:

· Performing a remote root compromise of an e-mail server

· Defacing a Web server

· Guessing and cracking passwords

· Copying a database containing credit card numbers

· Viewing sensitive data, including payroll records and medical information, without authorization

· Running a packet sniffer on a workstation to capture usernames and passwords

· Using a permission error on an anonymous FTP server to distribute pirated software and music files

· Dialing into an unsecured modem and gaining internal network access

· Posing as an executive, calling the help desk, resetting the executive's e-mail password, and learning the new password

· Using an unattended, logged-in workstation without permission

Intruder Behavior Patterns

The techniques and behavior patterns of intruders are constantly shifting, to exploit newly discovered weaknesses and to evade detection and countermeasures. Even so, intruders typically follow one of a number of recognizable behavior patterns, and these patterns typically differ from those of ordinary users.

HACKERS : Traditionally, those who hack into computers do so for the thrill of it or for status. The hacking community is a strong meritocracy in which status is determined by level of competence. Thus, attackers often look for targets of opportunity and then share the information with others.

The intruder took advantage of the fact that the corporate network was running unprotected services, some of which were not even needed. In this case, the key to the break-in was the pcAnywhere application. The manufacturer, Symantec, advertises this program as a remote control solution that enables secure connection to remote devices. But the attacker had an easy time gaining access to pcAnywhere; the administrator used the same three-letter username and password for the program. In this case, there was no intrusion detection system on the 700-node corporate network. The intruder was only discovered when a vice president walked into her office and saw the cursor moving files around on her Windows workstation. Benign intruders might be tolerable, although they do consume resources and may slow performance for legitimate users. However, there is no way in advance to know whether an intruder will be benign or malign. Consequently, even for systems with no particularly sensitive resources, there is a motivation to control this problem.

(a) Hacker

1. Select the target using IP lookup tools such as NSLookup, Dig, and others.
2. Map network for accessible services using tools such as NMAP.
3. Identify potentially vulnerable services (in this case, pcAnywhere).
4. Brute force (guess) pcAnywhere password.
5. Install remote administration tool called DameWare.
6. Wait for administrator to log on and capture his password.
7. Use that password to access remainder of network.

(b) Criminal Enterprise

1. Act quickly and precisely to make their activities harder to detect.
2. Exploit perimeter through vulnerable ports.
3. Use Trojan horses (hidden software) to leave back doors for reentry.
4. Use sniffers to capture passwords.
5. Do not stick around until noticed.
6. Make few or no mistakes.

(c) Internal Threat

1. Create network accounts for themselves and their friends.
2. Access accounts and applications they wouldn't normally use for their daily jobs.
3. E-mail former and prospective employers.
4. Conduct furtive instant-messaging chats.
5. Visit Web sites that cater to disgruntled employees, such as f'dcompany.com.
6. Perform large downloads and file copying.
7. Access the network during off hours.

Some Examples of Intruder Patterns of Behavior

Organizations can consider restricting remote logons to specific IP addresses and/or use virtual private network technology. One of the results of the growing awareness of the intruder problem has been the establishment of a number of computer emergency response teams (CERTs). These cooperative ventures collect information about system vulnerabilities and disseminate it to systems managers. Hackers also routinely read CERT reports. Thus, it is important for system administrators to quickly install all software patches to discovered vulnerabilities. Unfortunately, given the complexity of many IT systems, and the rate at which patches are released, this is increasingly difficult to achieve without automated updating. Even then, there are problems caused by incompatibilities resulting from the updated software. Hence the need for multiple layers of

defense in managing security threats to IT systems.

CRIMINALS Organized groups of hackers have become a widespread and common threat to Internet-based systems. These groups can be in the employ of a corporation or government but often are loosely affiliated gangs of hackers. Typically, these gangs are young, often Eastern European, Russian, or southeast Asian hackers who do business on the Web. They meet in underground forums with names like DarkMarket.org and theftservices.com to trade tips and data and coordinate attacks. A common target is a credit card file at an e-commerce server. Attackers attempt to gain root access. The card numbers are used by organized crime gangs to purchase expensive items and are then posted to carder sites, where others can access and use the account numbers; this obscures usage patterns and complicates investigation. Whereas traditional hackers look for targets of opportunity, criminal hackers usually have specific targets, or at least classes of targets in mind. Once a site is penetrated, the attacker acts quickly, scooping up as much valuable information as possible and exiting. IDSs and IPSs can also be used for these types of attackers but may be less effective because of the quick in-and-out nature of the attack. For e-commerce sites, database encryption should be used for sensitive customer information, especially credit cards. For hosted e-commerce sites (provided by an outsider service), the e-commerce organization should make use of a dedicated server (not used to support multiple customers) and closely monitor the provider's security services.

INSIDER ATTACKS Insider attacks are among the most difficult to detect and prevent. Employees already have access and knowledge about the structure and content of corporate databases. Insider attacks can be motivated by revenge of simply a feeling of entitlement. An example of the former is the case of Kenneth Patterson, fired from his position as data communications manager for American Eagle Outfitters. Patterson disabled the company's ability to process credit card purchases during five days of the holiday season of 2002. As for a sense of entitlement, there have always been many employees who felt entitled to take extra office supplies for home use, but this now extends to corporate data. An example is that of a vice president of sales for a stock analysis firm who quit to go to a competitor. Before she left, she copied the customer database to take

with her. The offender reported feeling no animus toward her former employee; she simply wanted the data because it would be useful to her.

Although IDS and IPS facilities can be useful in countering insider attacks, other more direct approaches are of higher priority. Examples include the following:

· Enforce least privilege, only allowing access to the resources employees need to do their job.

· Set logs to see what users access and what commands they are entering.

· Protect sensitive resources with strong authentication.

· Upon termination, delete employee's computer and network access.

· Upon termination, make a mirror image of employee's hard drive before reissuing it. That evidence might be needed if your company information turns up at a competitor.

5. **Need for IDS**

1. Detects Unauthorized Access

◇ Example: A hacker tries to access your company server using stolen login credentials.

➤ The IDS detects this unauthorized login attempt and alerts the security team.

2. Provides Early Alerts

◇ Example: A Distributed Denial of Service (DDoS) attack starts flooding your network.

➤ The IDS raises an alert as soon as abnormal traffic spikes are noticed

3. Monitors Network and Host Behavior

◇ Example: A user starts transferring unusually large files at midnight.

➤ The IDS flags this as unusual behavior based on historical patterns.

4. Complements Other Security Tools

◇ Example: A firewall allows web traffic on port 80, but someone hides a malware payload inside it.

➤ The IDS inspects the content of allowed traffic and detects the hidden malware.

5. Detects Insider Threats

◇ Example: A disgruntled employee tries to delete sensitive customer data from a shared drive.

➤ The IDS logs and alerts about this unauthorized file access and deletion attempt.

6. Supports Incident Response

◇ Example: During a ransomware attack, the IDS helps trace how the malware entered the system.

➤ This speeds up containment and recovery.

7. Enables Forensic Analysis

◇ Example: After a data breach, logs from the IDS reveal the attacker's IP address and the time of the breach.

➤ Investigators use this data to understand the attack and prevent future ones.

8. Assists in Regulatory Compliance

◇ Example: A healthcare company needs to meet HIPAA requirements.

➤ The IDS helps track all access to patient records and raises alerts on violations.

9. Reduces Downtime and Data Loss

◇ Example: A worm starts spreading across systems, threatening to shut down operations.

➤ The IDS catches it early, helping stop the spread and avoid major disruption.

10. Detects Known and Unknown Attacks

◇ Example:

Known Attack: An IDS detects a SQL Injection using a known signature.

Unknown Attack: An anomaly-based IDS notices abnormal user activity and flags it as a zero-day attack.

➤ Both are stopped before causing harm.

11. Preventing problems by increasing the perceived risk of discovery and punishment of attackers

A fundamental goal of computer security management is to affect the behavior of individual users in a way that protects information systems from security problems. Intrusion detection systems help organizations accomplish this goal by increasing the perceived risk of discovery and punishment of attackers. This serves as a significant deterrent to those who would violate security policy

12. Detecting problems that are not prevented by other security measures

- In many legacy systems, the operating systems cannot be patched or updated.
- Even in systems in which patches can be applied, administrators sometimes have neither sufficient time nor resource to track and install all the necessary patches. This is a common problem, especially in environments that include a large number of hosts or a wide range of different hardware or software environments.
- Users can have compelling operational requirements for network services and protocols that are known to be vulnerable to attack.
- Both users and administrators make errors in configuring and using systems.
- In configuring system access control mechanisms to reflect an organization's procedural computer use policy, discrepancies almost always occur. These disparities allow legitimate users to perform actions that are ill advised or that overstep their authorization.

In an ideal world, commercial software vendors would minimize vulnerabilities in their products, and user organizations would correct all reported vulnerabilities quickly and reliably. However, in the real world, this seldom happens thanks to our reliance on commercial software where new flaws and vulnerabilities are discovered on a daily basis. Given this state of affairs, intrusion detection can represent an excellent approach to protecting a system. An IDS can detect when an attacker has penetrated a system by

exploiting an uncorrected or uncorrectable flaw. Furthermore, it can serve an important function in system protection, by bringing the fact that the system has been attacked to the attention of the administrators who can contain and recover any damage that results. This is far preferable to simply ignoring network security threats where one allows the attackers continued access to systems and the information on them.

13.Detecting the preambles to attacks (often experienced as network probes and other tests for existing vulnerabilities)

When adversaries attack a system, they typically do so in predictable stages. The first stage of an attack is usually probing or examining a system or network, searching for an optimal point of entry. In systems with no IDS, the attacker is free to thoroughly examine the system with little risk of discovery or retribution. Given this unfettered access, a determined attacker will eventually find a vulnerability in such a network and exploit it to gain entry to various systems. The same network with an IDS monitoring its operations presents a much more formidable challenge to that attacker. Although the attacker may probe the network for weaknesses, the IDS will observe the probes, will identify them as suspicious, may actively block the attacker's access to the target system, and will alert security personnel who can then take appropriate actions to block subsequent access by the attacker. Even the presence of a reaction to the attacker's probing of the network will elevate the level of risk the attacker perceives, discouraging further attempts to target the network.

14.Documenting the existing threat

When you are drawing up a budget for network security, it often helps to substantiate claims that the network is likely to be attacked or is even currently under attack. Furthermore, understanding the frequency and characteristics of attacks allows you to understand what security measures are appropriate to protect the network against those attacks. IDSs verify, itemize, and characterize the threat from both outside and inside your organization's network, assisting you in making sound decisions regarding your allocation of computer security resources. Using IDSs in this manner is important, as many people mistakenly deny that anyone (outsider or insider) would be interested in breaking into their networks. Furthermore, the information that IDSs give you regarding the source

and nature of attacks allows you to make decisions regarding security strategy driven by demonstrated need, not guesswork or folklore.

15.Quality control for security design and administration

When IDSs run over a period of time, patterns of system usage and detected problems can become apparent. These can highlight flaws in the design and management of security for the system, in a fashion that supports security management correcting those deficiencies before they cause an incident.

16.Providing useful information about actual intrusions

Even when IDSs are not able to block attacks, they can still collect relevant, detailed, and trustworthy information about the attack that supports incident handling and recovery efforts. Furthermore, this information can, under certain circumstances, enable and support criminal or civil legal remedies. Ultimately, such information can identify problem areas in the organization's security configuration or policy.

Major types of IDSs

There are several types of IDSs available today, characterized by different monitoring and analysis approaches. Each approach has distinct advantages and disadvantages.

Process model for Intrusion Detection

Many IDSs can be described in terms of three fundamental functional components:

· *Information Sources* – the different sources of event information used to determine whether an intrusion has taken place. These sources can be drawn from different levels of the system, with network, host, and application monitoring most common.

· *Analysis* – the part of intrusion detection systems that actually organizes and makes sense of the events derived from the information sources, deciding when those events indicate that intrusions are occurring or have already taken place. The most common analysis approaches are *misuse detection* and *anomaly detection*.

· *Response* – the set of actions that the system takes once it detects intrusions. These are typically grouped into active and passive measures, with active measures involving some automated intervention on the part of the system, and passive measures involving reporting IDS findings to humans, who are then expected to take action based on those reports.

How do I distinguish between different Intrusion Detection approaches?

There are several design approaches used in Intrusion Detection. These drive the features provided by a specific IDS and determine the detection capabilities for that system. For those who must evaluate different IDS candidates for a given system environment, these approaches can help them determine what goals are best addressed by each IDS.

Architecture

The architecture of an IDS refers to how the functional components of the IDS are arranged with respect to each other. The primary architectural components are the Host, the system on which the IDS software runs, and the Target, the system that the IDS is monitoring for problems.

Host-Target Co-location

In early days of IDSs, most IDSs ran on the systems they protected. This was due to the fact that most systems were mainframe systems, and the cost of computers made a separate IDS system a costly extravagance. This presented a problem from a security point of view, as any attacker that successfully attacked the target system could simply disable the IDS as an integral portion of the attack.

Host-Target Separation

With the advent of workstations and personal computers, most IDS architects moved towards running the IDS control and analysis systems on a separate system, hence separating the IDS host and target systems. This improved the security of the IDS as this made it much easier to hide the existence of the IDS from attackers.

Goals of IDS

1. **Accountability**

Accountability is the capability to link a given activity or event back to the party responsible for initiating it. This is essential in cases where one wishes to bring criminal charges against an attacker. The goal statement associated with accountability is: *"I can deal with security attacks that occur on my systems as long as I know who did it (and where to find them.)"* Accountability is difficult in TCP/IP networks, where the protocols allow attackers to forge the identity of source addresses or other source identifiers. It is also extremely difficult to enforce accountability in any system that employs weak identification and authentication mechanisms.

2. Response

Response is the capability to recognize a given activity or event as an attack and then taking action to block or otherwise affect its ultimate goal. The goal statement associated with response is *"I don't care who attacks my system as long as I can recognize that the attack is taking place and block it."* Note that the requirements of detection are quite different for response than for accountability.

3. Control Strategy

Control Strategy describes how the elements of an IDS is controlled, and furthermore, how the input and output of the IDS is managed.

i. Centralized

Under centralized control strategies, all monitoring, detection and reporting is controlled directly from a central location

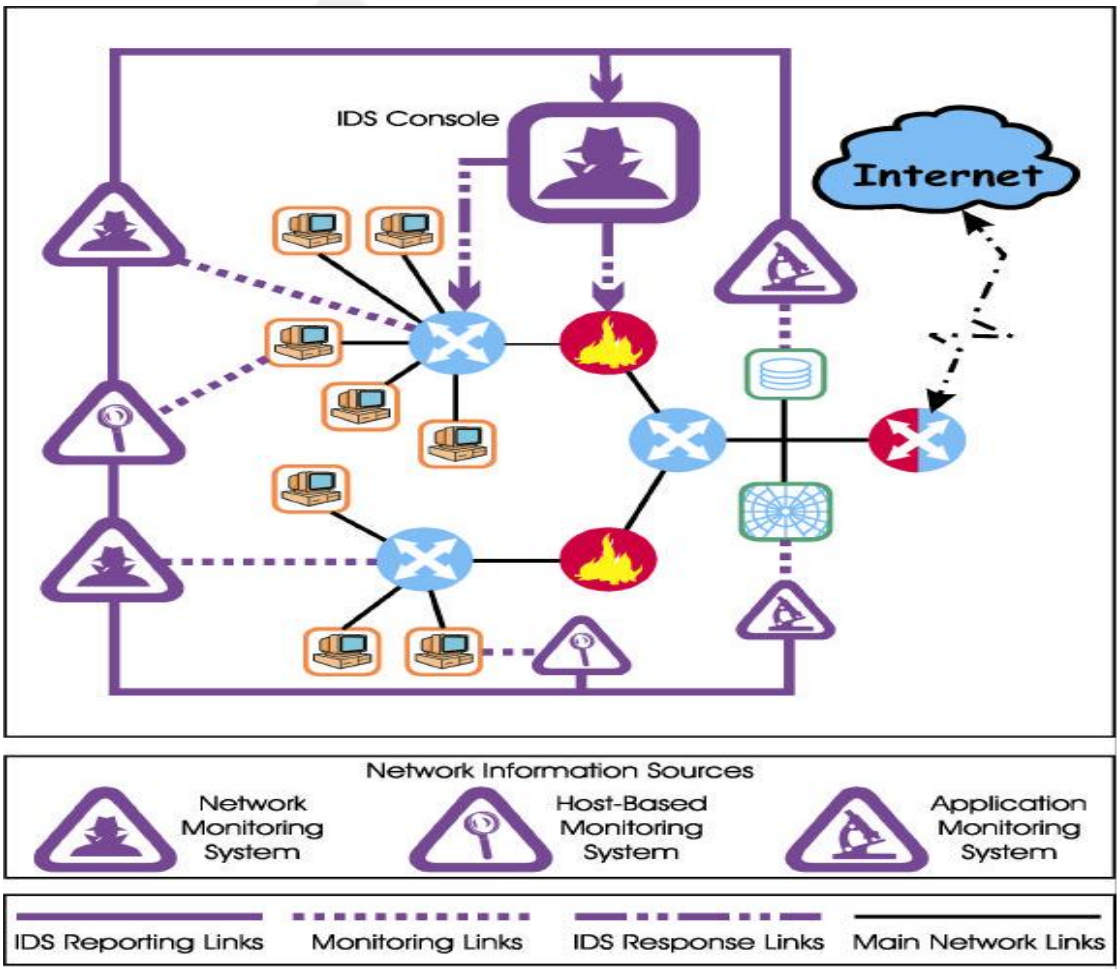


Figure 1: Centralized Control

ii. Partially Distributed

Monitoring and detection is controlled from a local control node, with hierarchical reporting to one or more central location(s).

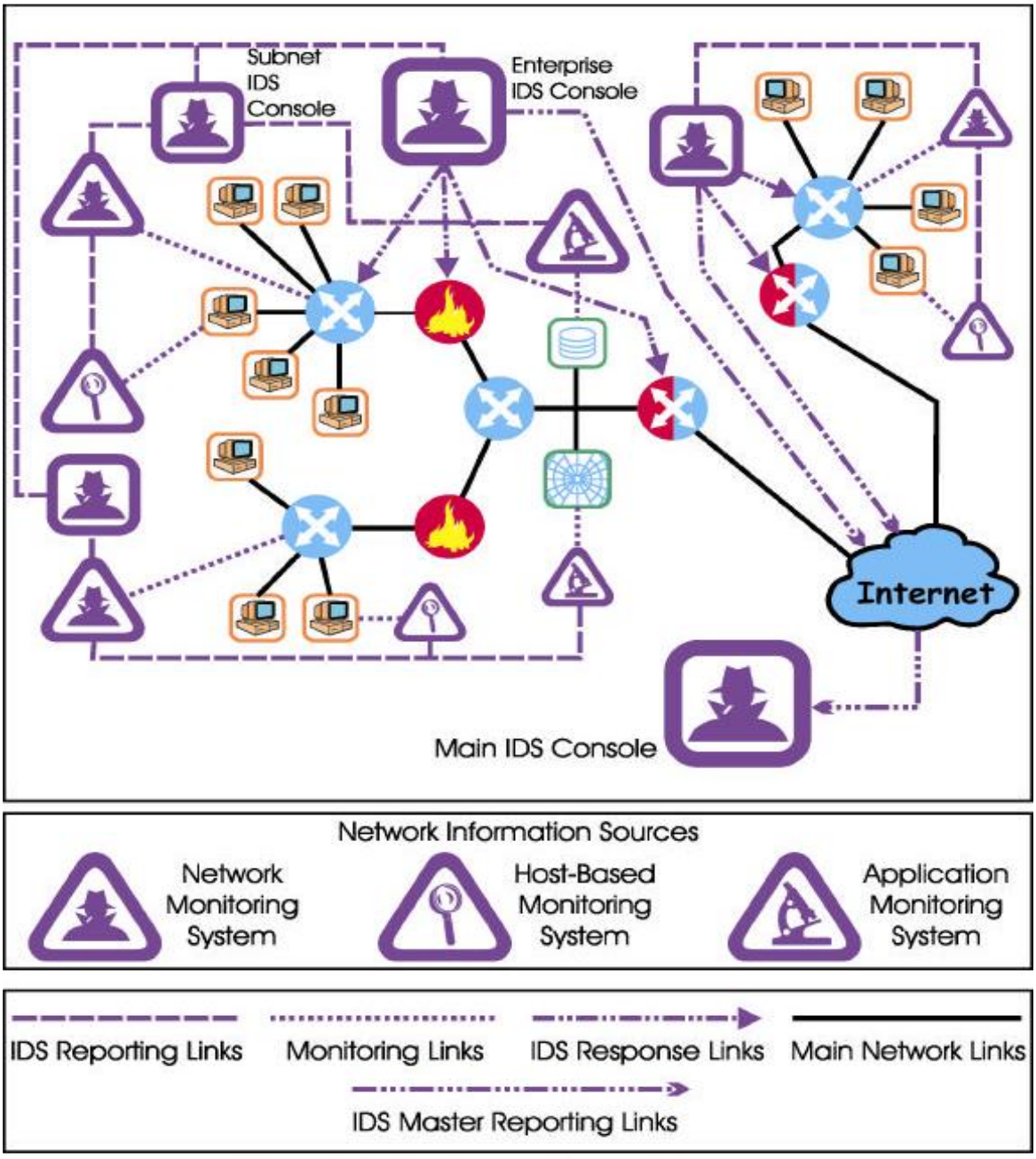


Figure 2: Distributed Control Strategy

iii. Fully Distributed

Monitoring and detection is done using an agent-based approach, where response decisions are made at the point of analysis.

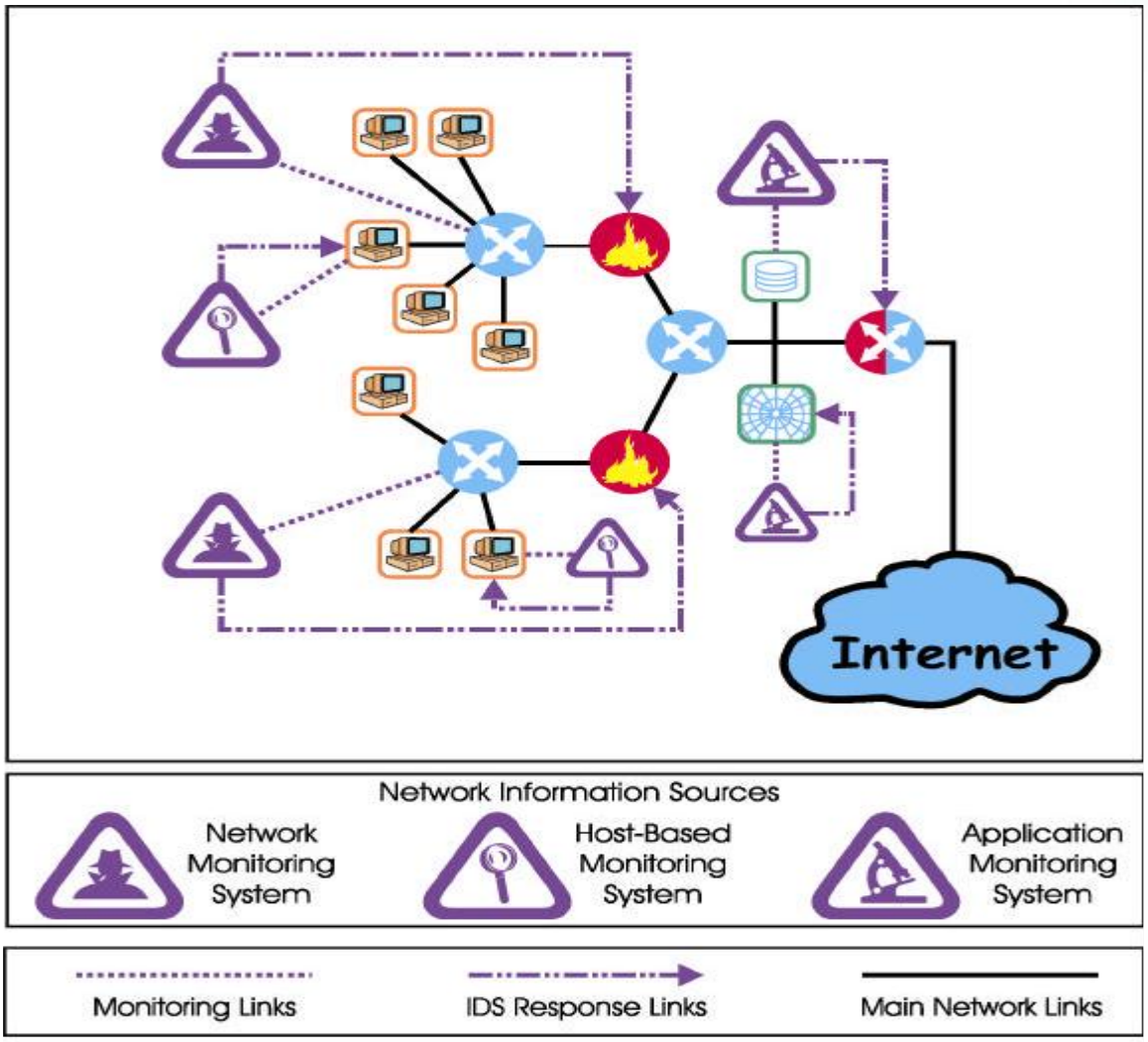


Figure 3: Fully Distributed (Agent-Based) Control

a. Timing

Timing refers to the elapsed time between the events that are monitored and the analysis of those events.

1. Interval-Based (Batch Mode)

In interval-based IDSs, the information flow from monitoring points to analysis engines is not continuous. In effect, the information is handled in a fashion similar to "store and forward" communications schemes. Many early host-based IDSs used this timing scheme, as they relied on operating system audit trails, which were generated as files. Intervalbased IDSs are precluded from performing active responses.

2. Real-Time(Continuous)

Real-time IDSs operate on continuous information feeds from information

sources. This is the predominant timing scheme for networkbased IDSs, which gather information from network traffic streams. In this document, we use the term "real-time" as it is used in process control situations. This means that detection performed by a "real-time" IDS yields results quickly enough to allow the IDS to take action that affects the progress of the detected attack.

6. **IDS Techniques**

There are two primary approaches to analyzing events to detect attacks: misuse detection and anomaly detection. Misuse detection, in which the analysis targets something known to be "bad", is the technique used by most commercial systems. Anomaly detection, in which the analysis looks for abnormal pattern of activity, has been, and continues to be, the subject of a great deal of research. Anomaly detection is used in limited form by a number of IDSs. There are strengths and weaknesses associated with each approach, and it appears that the most effective IDSs use mostly misuse detection methods with a smattering of anomaly detection components.

a. Rule-Based Detection(Misuse Detection)

Misuse detectors analyze system activity, looking for events or sets of events that match a predefined pattern of events that describe a known attack. As the patterns corresponding to known attacks are called *signatures*, misuse detection is sometimes called "signature-based detection." The most common form of misuse detection used in commercial products specifies each pattern of events corresponding to an attack as a separate signature. However, there are more sophisticated approaches to doing misuse detection (called "state-based" analysis techniques) that can leverage a single signature to detect groups of attacks.

Rule-based detection, also referred to as *signature detection, pattern matching* and *misuse detection*, is the first scheme that was used in early intrusion-detection systems. Rule-based detection uses pattern matching to detect known attack patterns.

The four phases of the analysis process are applied in a rule-based detection system:

1. **Preprocessing** The first step is to collect data about intrusions, vulnerabilities, and attacks, and put them into a classification scheme or pattern descriptor. From the classification scheme, a behavioral model is

built, and then put into a common format:

¨ **Signature Name** The given name of a signature

¨ **Signature ID** A unique ID for the signature

¨ **Signature Description** Description of the signature and what it does

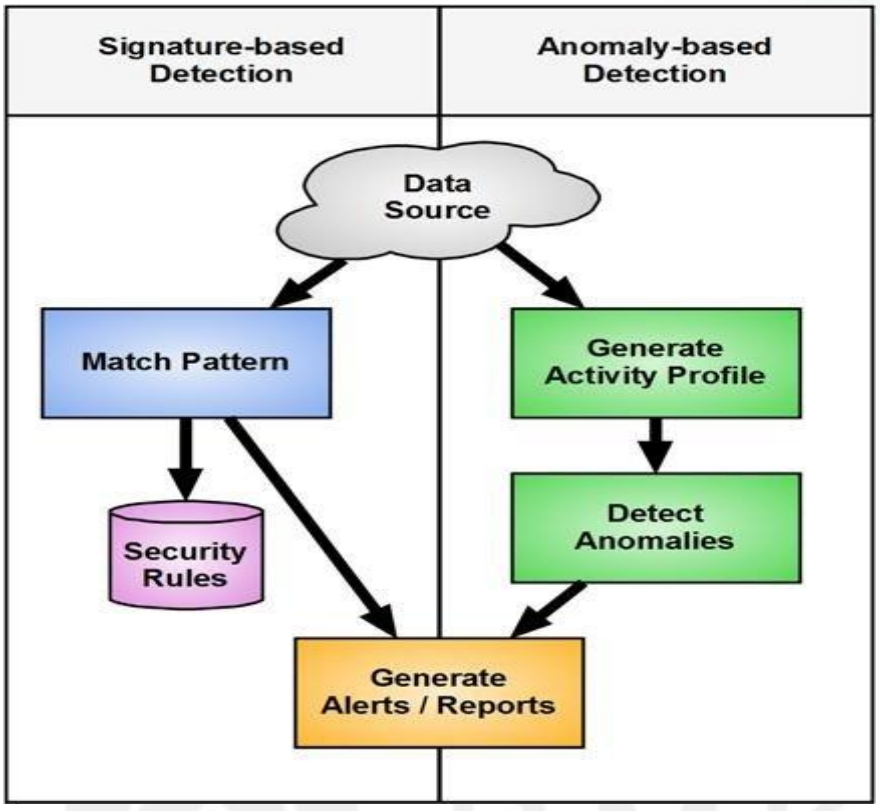
Possible False Positive Description An explanation of any "false positives" that may appear to be an exploit but are actually normal network activity.

Related Vulnerability Information This field has any related vulnerability information

User Notes This field allows a security professional to add specific notes related to their network

The pattern descriptors are typically either content-based signatures, which examine the payload and header of a packet, or context-based signatures that evaluate *only* the packet headers to identify an alert. Note that pattern descriptors can be atomic (single) or composite (multiple) descriptors. An atomic descriptor requires only one packet to be inspected to identify an alert, while a composite descriptor requires multiple packets to be inspected to identify an alert. The pattern descriptors are then put into a knowledge base that contains the criteria for analysis.

2. **Analysis** The event data are formatted and compared against the knowledge base by using a pattern-matching analysis engine. The analysis engine looks for defined patterns that are known as attacks.
3. **Response** If the event matches the pattern of an attack, the analysis engine sends an alert. If the event is a partial match, the next event is examined. Note that partial matches can only be analyzed with a stateful detector, which has the ability to maintain state, as many IDS systems do. Different responses. can be returned depending on the specific event records.
4. **Refinement** Refinement of pattern-matching analysis comes down to updating signatures, because an IDS is only as good as its latest signature update. This is one of the drawbacks of pattern-matching analysis. Most IDSs allow automatic and manual updating of attack signatures.



Advantages:

· Misuse detectors are very effective at detecting attacks without generating an overwhelming number of false alarms.

Misuse detectors can quickly and reliably diagnose the use of a specific attack tool or technique. This can help security managers prioritize corrective measures.

· Misuse detectors can allow system managers, regardless of their level of security expertise, to track security problems on their systems, initiating incident handling procedures.

Disadvantages:

· Misuse detectors can only detect those attacks they know about – therefore they must be constantly updated with signatures of new attacks.

· Many misuse detectors are designed to use tightly defined signatures that prevent them from detecting variants of common attacks. State-based misuse detectors can overcome this limitation, but are not commonly used in commercial IDSs.

b. Profile Based Detection (Anomaly Detection)

Anomaly detectors identify abnormal unusual behavior (anomalies) on a host or network. They function on the assumption that attacks are different from "normal" (legitimate) activity and can therefore be detected by systems that

identify these differences. Anomaly detectors construct profiles representing normal behavior of users, hosts, or network connections. These profiles are constructed from historical data collected over a period of normal operation. The detectors then collect event data and use a variety of measures to determine when monitored activity deviates from the norm.

The measures and techniques used in anomaly detection include:

· Threshold detection, in which certain attributes of user and system behavior are expressed in terms of counts, with some level established as permissible. Such behavior attributes can include the number of files accessed by a user in a given period of time, the number of failed attempts to login to the system, the amount of CPU utilized by a process, etc. This level can be static or heuristic (*i.e.,* designed to change with actual values observed over time)

· Statistical measures, both parametric, where the distribution of the profiled attributes is assumed to fit a particular pattern, and non-parametric, where the distribution of the profiled attributes is "learned" from a set of historical values, observed over time.

· Rule-based measures, which are similar to non-parametric statistical measures in that observed data defines acceptable usage patterns, but differs in that those patterns are specified as rules, not numeric quantities

· Other measures, including neural networks, genetic algorithms, and immune system models.

Only the first two measures are used in current commercial IDSs.

Unfortunately, anomaly detectors and the IDSs based on them often produce a large number of false alarms, as normal patterns of user and system behavior can vary wildly. Despite this shortcoming, researchers assert that anomaly-based IDSs are able to detect new attack forms, unlike signature-based IDSs that rely on matching patterns of past attacks.

Furthermore, some forms of anomaly detection produce output that can in turn be used as information sources for misuse detectors. For example, a threshold-based anomaly detector can generate a figure representing the "normal" number of files accessed by a particular user; the misuse detector can use this figure as part of a detection signature that says "if the number of files accessed by this user exceeds this "normal" figure by ten percent, trigger an alarm."

Although some commercial IDSs include limited forms of anomaly detection, few, if any, rely solely on this technology.The anomaly detection that exists in commercial systems usually revolves around detecting network or port scanning. However, anomaly detection remains an active intrusion detection research area and may play a greater part in future IDSs.

One key distinction between anomaly detection and other analysis schemes is that anomaly-based schemes not only define activities that are *not allowed*, but also activities that are *allowed*. In addition, anomaly detection is typically used for its ability to collect statistical behavior and characteristic behavior. Statistics are quantitative and characteristics are more qualitative. For example, "This server's UDP traffic never exceeds 25 percent of capacity" describes a statistical behavior, and "User Stan321 does not normally FTP files outside of the company" describes a characteristic behavior.

Anomaly-based schemes fall into three main categories: behavioral, traffic pattern, and protocol. *Behavioral analysis* looks for anomalies in the types of behavior that have been statistically baselined, such as relationships in packets and what is being sent over a network. *Traffic-pattern analysis* looks for specific patterns in network traffic. *Protocol analysis* looks for network protocol violations or misuse based on RFC-based behavior. Protocol analysis has the benefit of identifying possible attacks that are not yet publicized or that there is no known signature or remedy for.

The analysis model in the context of anomaly detection:

1. **Preprocessing** The first step in the analysis process is collecting the data in which behavior considered normal on the network is baselined over a period of time. The data are put into a numeric form and is then formatted. Then the information is classified into a statistical profile that is based on different algorithms in the knowledge base.
2. **Analysis** The event data are typically reduced to a profile vector, which is then compared to the knowledge base. The contents of the profile vector are compared to a historical record for that particular user, and any data that fall

outside of the baseline normal activity is labeled a deviation.

3. **Response** At this point, a response can be triggered either automatically or manually.
4. **Refinement** The data records must be kept updated. The profile vector history will typically be deleted after a specific number of days. In addition, different weighting systems can be used to add more weight to recent behaviors than past behaviors.

Advantages:

· IDSs based on anomaly detection detect unusual behavior and thus have the ability to detect symptoms of attacks without specific knowledge of details.

· Anomaly detectors can produce information that can in turn be used to define signatures for misuse detectors.

Disadvantages:

· Anomaly detection approaches usually produce a large number of false alarms due to the unpredictable behaviors of users and networks.

· Anomaly detection approaches often require extensive "training sets" of system event records in order to characterize normal behavior patterns.

c. Host-based IDSs

Host-based IDSs add a specialized layer of security software to vulnerable or sensitive systems; examples include database servers and administrative systems. The host-based IDS monitors activity on the system in a variety of ways to detect suspicious behavior. In some cases, an IDS can halt an attack before any damage is done, but its primary purpose is to detect intrusions, log suspicious events, and send alerts.

The primary benefit of a host-based IDS is that it can detect both external and internal intrusions, something that is not possible either with network-based IDSs or firewalls.

Host-based IDSs follow one of two general approaches to intrusion detection:

1. Anomaly detection: Involves the collection of data relating to the behavior of legitimate users over a period of time. Then statistical tests are applied to observed behavior to determine with a high level of confidence whether that behavior is not legitimate user behavior. The following are two approaches to statistical anomaly detection:

a. Threshold detection: This approach involves defining thresholds, independent of user, for the frequency of occurrence of various events.

b. Profile based: A profile of the activity of each user is developed and used to detect changes in the behavior of individual accounts.

2. Signature detection: Involves an attempt to define a set of rules or attack patterns that can be used to decide that a given behavior is that of an intruder.

In essence, anomaly approaches attempt to define normal, or expected, behavior, whereas signature-based approaches attempt to define proper behavior. In terms of the types of attackers listed earlier, anomaly detection is effective against masqueraders, who are unlikely to mimic the behavior patterns of the accounts they appropriate. On the other hand, such techniques may be unable to deal with misfeasors. For such attacks, signature-based approaches may be able to recognize events and sequences that, in context, reveal penetration. In practice, a system may employ a combination of both approaches to be effective against a broad range of attacks.

Audit Records

A fundamental tool for intrusion detection is the audit record. 1 Some record of ongoing activity by users must be maintained as input to an IDS. Basically, two plans are used:

· **Native audit records:** Virtually all multiuser operating systems include accounting software that collects information on user activity. The advantage of using this information is that no additional collection software is needed. The disadvantage is that the native audit records may not contain the needed information or may not contain it in a convenient form.

· **Detection-specific audit records:** A collection facility can be implemented that generates audit records containing only that information required by the IDS. One advantage of such an approach is that it could be made vendor independent and ported to a variety of systems. The disadvantage is the extra overhead involved in having, in effect, two accounting packages running on a machine. A good example of detection-specific audit records is one developed by Dorothy Denning [DENN87]. Each audit record contains the following fields:

· **Subject:** Initiators of actions. A subject is typically a terminal user but might also be a process acting on behalf of users or groups of users. All activity arises

through commands issued by subjects. Subjects may be grouped into different access classes, and these classes may overlap.

· **Action:** Operation performed by the subject on or with an object; for example, login, read, perform I/O, execute.

· **Object:** Receptors of actions. Examples include files, programs, messages, records, terminals, printers, and user- or program-created structures. When a subject is the recipient of an action, such as electronic mail, then that subject is considered an object. Objects may be grouped by type. Object granularity may vary by object type and by environment. For example, database actions may be audited for the database as a whole or at the record level.

· **Exception-Condition:** Denotes which, if any, exception condition is raised on return.

· **Resource-Usage:** A list of quantitative elements in which each element gives the amount used of some resource (e.g., number of lines printed or displayed, number of records read or written, processor time, I/O units used, session elapsed time).

· **Time-Stamp:** Unique time-and-date stamp identifying when the action took place.

Most user operations are made up of a number of elementary actions. For example, a file copy involves the execution of the user command, which includes doing access validation and setting up the copy, plus the read from one file, plu the write to another file. Consider the command

COPY GAME.EXE TO <Library>GAME.EXE issued by Smith to copy an executable file GAME from the current directory to the directory. The following audit records may be generated:

Smith	execute	<Library>COPY.EXE	0	CPU = 00002	11058721678
Smith	read	<Smith>GAME.EXE	0	RECORDS = 0	11058721679
Smith	execute	<Library>COPY.EXE	write-viol	RECORDS = 0	11058721680

In this case, the copy is aborted because Smith does not have write permission to < Library> .

The decomposition of a user operation into elementary actions has three

advantages:

1. Because objects are the protectable entities in a system, the use of elementary actions enables an audit of all behavior affecting an object. Thus, the system can detect attempted subversions of access controls (by noting an abnormality in the number of exception conditions returned) and can detect successful subversions by noting an abnormality in the set of objects accessible to the subject.

2. Single-object, single-action audit records simplify the model and the implementation.

3. Because of the simple, uniform structure of the detection-specific audit records, it may be relatively easy to obtain this information or at least part of it by a straightforward mapping from existing native audit records to the detection-specific audit records.

Anomaly Detection

As was mentioned, anomaly detection techniques fall into two broad categories: threshold detection and profile-based systems. Threshold detection involve counting the number of occurrences of a specific event type over an interval of time. If the count surpasses what is considered a reasonable number that one might expect to occur, then intrusion is assumed. Threshold analysis, by itself, is a crude and ineffective detector of even moderately sophisticated attacks. Both the threshold and the time interval must be determined. Because of the variability across users, such thresholds are likely to generate either a lot of false positives or a lot of false negatives. However, simple threshold detectors may be useful in conjunction with more sophisticated techniques.

Profile-based anomaly detection focuses on characterizing the past behavior of individual users or related groups of users and then detecting significant deviations. A profile may consist of a set of parameters, so that deviation on just a single parameter may not be sufficient in itself to signal an alert.

The foundation of this approach is an analysis of audit records. The audit records provide input to the intrusion detection function in two ways. First, the designer must decide on a number of quantitative metrics that can be used to measure user behavior. An analysis of audit records over a period of time can be used to determine the activity profile of the average user. Thus, the audit records serve

to define typical behavior. Second, current audit records are the input used to detect intrusion. That is, the intrusion detection model analyzes incoming audit records to determine deviation from average behavior. Examples of metrics that are useful for profile-based intrusion detection are the following:

· **Counter:** A nonnegative integer that may be incremented but not decremented until it is reset by management action. Typically, a count of certain event types is kept over a particular period of time. Examples include the number of logins by a single user during an hour, the number of times a given command is executed during a single user session, and the number of password failures during a minute.

· **Gauge:** A nonnegative integer that may be incremented or decremented. Typically, a gauge is used to measure the current value of some entity. Examples include the number of logical connections assigned to a user application and the number of outgoing messages queued for a user process.

· **Interval timer:** The length of time between two related events. An example is the length of time between successive logins to an account.

· **Resource utilization:** Quantity of resources consumed during a specified period. Examples include the number of pages printed during a user session and total time consumed by a program execution. Given these general metrics, various tests can be performed to determine whether current activity fits within acceptable limits. [DENN87] lists the following approaches that may be taken:

· Mean and standard deviation

· Multivariate

· Markov process

· Time series

· Operational

The simplest statistical test is to measure the **mean and standard deviation** of a parameter over some historical period. This gives a reflection of the average behavior and its variability. The use of mean and standard deviation is applicable to a wide variety of counters, timers, and resource measures. But these measures, by themselves, are typically too crude for intrusion detection purposes.

A **multivariate** model is based on correlations between two or more variables.

Intruder behavior may be characterized with greater confidence by considering such correlations (e.g., processor time and resource usage, or login frequency and session elapsed time). ious states. As an example, this model might be used to look at transitions between certain commands.

A **time series** model focuses on time intervals, looking for sequences of events that happen too rapidly or too slowly. A variety of statistical tests can be applied to characterize abnormal timing.

Finally, an **operational model** is based on a judgment of what is considered abnormal, rather than an automated analysis of past audit records. Typically, fixed limits are defined and intrusion is suspected for an observation that is outside the limits. This type of approach works best where intruder behavior can be deduced from certain types of activities. For example, a large number of login attempts over a short period suggests an attempted intrusion.

Measure	Model	Type of Intrusion Detected
Login and Session Activity		
Login frequency by day and time	Mean and standard deviation	Intruders may be likely to log in during off hours.
Frequency of login at different locations	Mean and standard deviation	Intruders may log in from a location that a particular user rarely or never uses.
Time since last login	Operational	Break-in on a "dead" account.
Elapsed time per session	Mean and standard deviation	Significant deviations might indicate masquerader.
Quantity of output to location	Mean and standard deviation	Excessive amounts of data transmitted to remote locations could signify leakage of sensitive data.
Session resource utilization	Mean and standard deviation	Unusual processor or I/O levels could signal an intruder.
Password failures at login	Operational	Attempted break-in by password guessing.
Failures to login from specified terminals	Operational	Attempted break-in.
Command or Program Execution Activity		
Execution frequency	Mean and standard deviation	May detect intruders, who are likely to use different commands, or a successful penetration by a legitimate user, who has gained access to privileged commands.
Program resource utilization	Mean and standard deviation	An abnormal value might suggest injection of a virus or Trojan horse, which performs side effects that increase I/O or processor utilization.
Execution denials	Operational model	May detect penetration attempt by individual user who seeks higher privileges.
Fil Access Activity		
Read, write, create, delete frequency	Mean and standard deviation	Abnormalities for read and write access for individual users may signify masquerading or browsing.
Records read, written	Mean and standard deviation	Abnormality could signify an attempt to obtain sensitive data by inference and aggregation.
Failure count for read, write, create, delete	Operational	May detect users who persistently attempt to access unauthorized files.

The main advantage of the use of statistical profiles is that a prior knowledge of security flaws is not required. The detector program learns what is "normal" behavior and then looks for deviations. The approach is not based on system-dependent characteristics and vulnerabilities. Thus, it should be readily portable among a variety of systems.

Signature Detection

Signature techniques detect intrusion by observing events in the system and

applying a set of rules that lead to a decision regarding whether a given pattern of activity is or is not suspicious. In very general terms, we can characterize all approaches as focusing on either anomaly detection or penetration identification, although there is some overlap in these approaches.

Rule-based anomaly detection is similar in terms of its approach and strengths to statistical anomaly detection. With the rule-based approach, historical audit records are analyzed to identify usage patterns and to generate automatically rules that describe those patterns. Rules may represent past behavior patterns of users, programs, privileges, time slots, terminals, and so on. Current behavior is then observed, and each transaction is matched against the set of rules to determine if it conforms to any historically observed pattern of behavior. As with statistical anomaly detection, rule-based anomaly detection does not require knowledge of security vulnerabilities within the system. Rather, the scheme is based on observing past behavior and, in effect, assuming that the future will be like the past. In order for this approach to be effective, a rather large database of rules will be needed. For example, a scheme described in [VACC89] contains anywhere from 10 4 to 10 6 rules.

Rule-based penetration identification takes a very different approach to intrusion detection. The key feature of such systems is the use of rules for identifying known penetrations or penetrations that would exploit known weaknesses. Rules can also be defined that identify suspicious behavior, even when the behavior is within the bounds of established patterns of usage. Typically, the rules used in these systems are specific to the machine and operating system. The most fruitful approach to developing such rules is to analyze attack tools and scripts collected on the Internet. These rules can be supplemented with rules generated by knowledgeable security personnel. In this latter case, the normal procedure is to interview system administrators and security analysts to collect a suite of known penetration scenarios and key events that threaten the security of the target system.

A simple example of the type of rules that can be used is found in NIDX, an early system that used heuristic rules that can be used to assign degrees of suspicion to activities [BAUE88]. Example heuristics are the following:

1. Users should not read files in other users' personal directories.

2. Users must not write other users' files.

3. Users who log in after hours often access the same files they used earlier.

4. Users do not generally open disk devices directly but rely on higher-level operating system utilities.

5. Users should not be logged in more than once to the same system.

6. Users do not make copies of system programs.

The penetration identification scheme used in IDES is representative of the strategy followed. Audit records are examined as they are generated, and they are matched against the rule base. If a match is found, then the user's *suspicion rating* is increased. If enough rules are matched, then the rating will pass a threshold that results in the reporting of an anomaly.

The IDES approach is based on an examination of audit records. A weakness of this plan is its lack of flexibility. For a given penetration scenario, there may be a number of alternative audit record sequences that could be produced, each varying from the others slightly or in subtle ways. It may be difficult to pin down all these variations in explicit rules. Another method is to develop a higher-level model independent of specific audit records. An example of this is a state transition model known as USTAT [VIGN02, ILGU95]. USTAT deals in general actions rather than the detailed specific actions recorded by the UNIX auditing mechanism. USTAT is implemented on a SunOS system that provides audit records on 239 events. Of these, only 28 are used by a preprocessor, which maps these onto 10 general actions (Table 8.3). Using just these actions and the parameters that are invoked with each action, a state transition diagram is developed that characterizes suspicious activity. Because a number of different auditable events map into a smaller number of actions, the rule-creation process is simpler. Furthermore, the state transition diagram model is easily modified to accommodate newly learned intrusion behaviors.

The Base-Rate Fallacy

To be of practical use, an IDS should detect a substantial percentage of intrusions while keeping the false alarm rate at an acceptable level. If only a modest percentage of actual intrusions are detected, the system provides a false sense of security. On the other hand, if the system frequently triggers an alert when there

Table 8.3 USTAT Actions versus SunOS Event Types

USTAT Action	SunOS Event Type
Read	open_r, open_rc, open_rtc, open_rwc, open rwtc, open_rt, open_rw, open_rwt
Write	truncate, ftruncate, creat, open_rtc, open_rwc, open_rwtc, open_rt, open_rw, open_rwt, open_w, open_wt, open_wc, open_wct
Create	mkdir, creat, open_rc, open_rtc, open_rwc, open_rwtc, open_wc, open_wtc, mknod
Delete	rmdir, unlink
Execute	exec, execve
Exit	exit
Modify_Owner	chown, fchown
Modify_Perm	chmod, fchmod
Rename	rename
Hardlink	link

is no intrusion (a false alarm), then either system managers will begin to ignore the alarms, or much time will be wasted analyzing the false alarms. Unfortunately, because of the nature of the probabilities involved, it is very difficult to meet the standard of high rate of detections with a low rate of false alarms. In general, if the actual numbers of intrusions is low compared to the number of legitimate uses of a system, then the false alarm rate will be high unless the test is extremely discriminating. This is an example of a phenomenon known as the *base-rate fallacy.* A study of existing IDSs, reported in [AXEL00], indicated that current systems have not overcome the problem of the base-rate fallacy.

d. Distributed Host Based IDS

Traditionally, work on host-based IDSs focused on single-system stand-alone facilities. The typical organization, however, needs to defend a distributed collection of hosts supported by a LAN or internetwork. Although it is possible to mount a defense by using stand-alone IDSs on each host, a more effective defense can be achieved by coordination and cooperation among IDSs across the network.

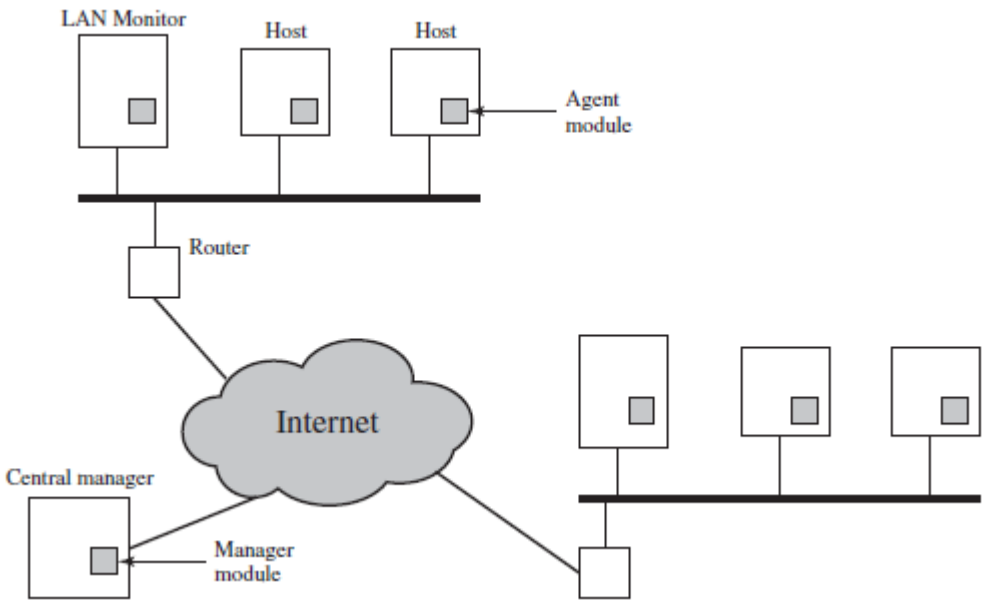
Major issues in the design of a distributed IDS

· A distributed IDS may need to deal with different audit record formats. In a heterogeneous environment, different systems will employ different native audit collection systems and, if using intrusion detection, may employ different formats for security-related audit records.

· One or more nodes in the network will serve as collection and analysis points for the data from the systems on the network. Thus, either raw audit data or summary data must be transmitted across the network. Therefore, there is a requirement to assure the integrity and confidentiality of these data. Integrity is required to prevent an intruder from masking his or her activities by altering the transmitted audit information. Confidentiality is required because the transmitted audit information could be valuable.

· Either a centralized or decentralized architecture can be used. With a centralized architecture, there is a single central point of collection and analysis of all audit data. This eases the task of correlating incoming reports but creates a potential bottleneck and single point of failure. With a decentralized architecture, there is more than one analysis center, but these must coordinate their activities and exchange information.

Figure below shows the overall architecture, which consists of three main components:



Architecture for Distributed Intrusion Detection

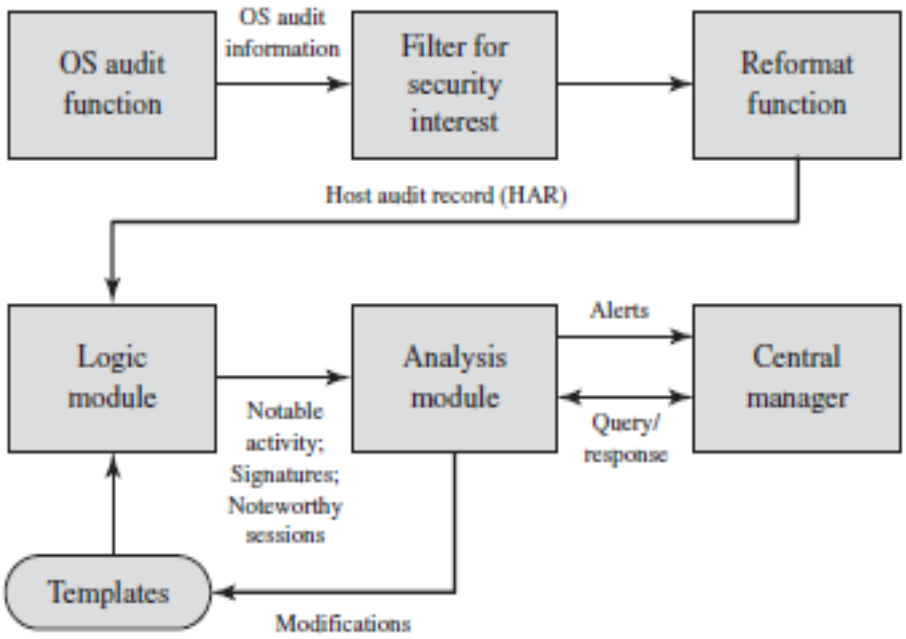
· **Host agent module:** An audit collection module operating as a background process on a monitored system. Its purpose is to collect data on securityrelated events on the host and transmit these to the central manager.

· **LAN monitor agent module:** Operates in the same fashion as a host agent module except that it analyzes LAN traffic and reports the results to the central manager.

· **Central manager module:** Receives reports from LAN monitor and host agents and processes and correlates these reports to detect intrusion.

The scheme is designed to be independent of any operating system or system auditing implementation. Figure 8.3 shows the general approach that is taken. The agent captures each audit record produced by the native audit collection system. A filter is applied that retains only those records that are of security interest. These records are then reformatted into a standardized format referred to as the host audit record (HAR). Next, a template-driven logic module analyzes the records for suspicious activity. At the lowest level, the agent scans for notable events that are of interest independent of any past events. Examples include failed files, accessing system files, and changing a file's access control. At the next higher level, the agent looks for sequences of events, such as known attack patterns (signatures). Finally, the agent looks for anomalous behavior of an individual user based on a historical profile of that user, such as number of programs executed, number of files accessed, and the like.

When suspicious activity is detected, an alert is sent to the central manager. The central manager includes an expert system that can draw inferences from received data. The manager may also query individual systems for copies of HARs to correlate with those from other agents. The LAN monitor agent also supplies information to the central manager. The LAN monitor agent audits host-host connections, services used, and volume of traffic. It searches for significant events, such as sudden changes in network load, the use of security-related services, and network activities such as *rlogin* . It offers a foundation for a machine-independent approach that can expand from stand-alone intrusion detection to a system that is able to correlate activity from a number of sites and networks to detect suspicious activity that would otherwise remain undetected.



Agent Architecture

e. Network Based IDS

A network-based IDS (NIDS) monitors traffic at selected points on a network or interconnected set of networks. The NIDS examines the traffic packet by packet in real time, or close to real time, to attempt to detect intrusion patterns. The NIDS may examine network-, transport-, and/or application-level protocol activity. Note the contrast with a host-based IDS; a NIDS examines packet traffic directed toward potentially vulnerable computer systems on a network. A host-based system examines user and software activity on a host.

A typical NIDS facility includes a number of sensors to monitor packet traffic, one or more servers for NIDS management functions, and one or more management consoles for the human interface. The analysis of traffic patterns to detect intrusions may be done at the sensor, at the management server, or some combination of the two.

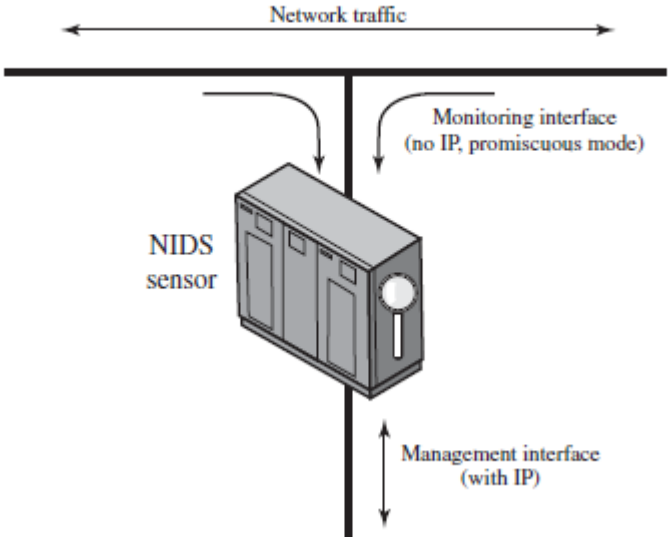
Types of Network Sensors

Sensors can be deployed in one of two modes: inline and passive. An **inline sensor** is inserted into a network segment so that the traffic that it is monitoring must pass through the sensor. One way to achieve an inline sensor is to combine NIDS sensor logic with another network device, such as a firewall or LAN switch. This approach has the advantage that no additional separate hardware

devices are needed; all that is required is NIDS sensor software. An alternative is a stand-alone inline NIDS sensor. The primary motivation for the use of inline sensors is to enable them to block an attack when one is detected. In this case the device is performing both intrusion detection and intrusion prevention functions.

More commonly, **passive sensors** are used. A passive sensor monitors a copy of network traffic; the actual traffic does not pass through the device. From the point of view of traffic flow, the passive sensor is more efficient than the inline sensor, because it does not add an extra handling step that contributes to packet delay.

Figure below illustrates a typical passive sensor configuration. The sensor connects to the network transmission medium, such as a fiber optic cable, by a direct physical tap. The tap provides the sensor with a copy of all network traffic being carried by the medium. The network interface card (NIC) for this tap usually does not have an IP address configured for it. All traffic into this NIC is simply collected with no protocol interaction with the network. The sensor has a second NIC that connects to the network with an IP address and enables the sensor to communicate with a NIDS management server.



Passive NIDS

NIDS Sensor Deployment

Consider an organization with multiple sites, each of which has one or more LANs, with all of the networks interconnected via the Internet or some other

WAN technology. For a comprehensive NIDS strategy, one or more sensors are needed at each site. Within a single site, a key decision for the security administrator is the placement of the sensors.

Figure 8.5 illustrates a number of possibilities. In general terms, this configuration is typical of larger organizations. All Internet traffic passes through an external firewall that protects the entire facility. 2 Traffic from the outside world, such as customers and vendors that need access to public services, such as Web and mail, is monitored. The external firewall also provides a degree of protection for those parts of the network that should only be accessible by users from other corporate sites. Internal firewalls may also be used to provide more specific protection to certain parts of the network.

A common location for a NIDS sensor is just inside the external firewall (**location 1** in the figure). This position has a number of advantages:

· Sees attacks, originating from the outside world, that penetrate the network's perimeter defenses (external firewall).

· Highlights problems with the network firewall policy or performance.

· Sees attacks that might target the Web server or ftp server.

· Even if the incoming attack is not recognized, the IDS can sometimes recognize the outgoing traffic that results from the compromised server.

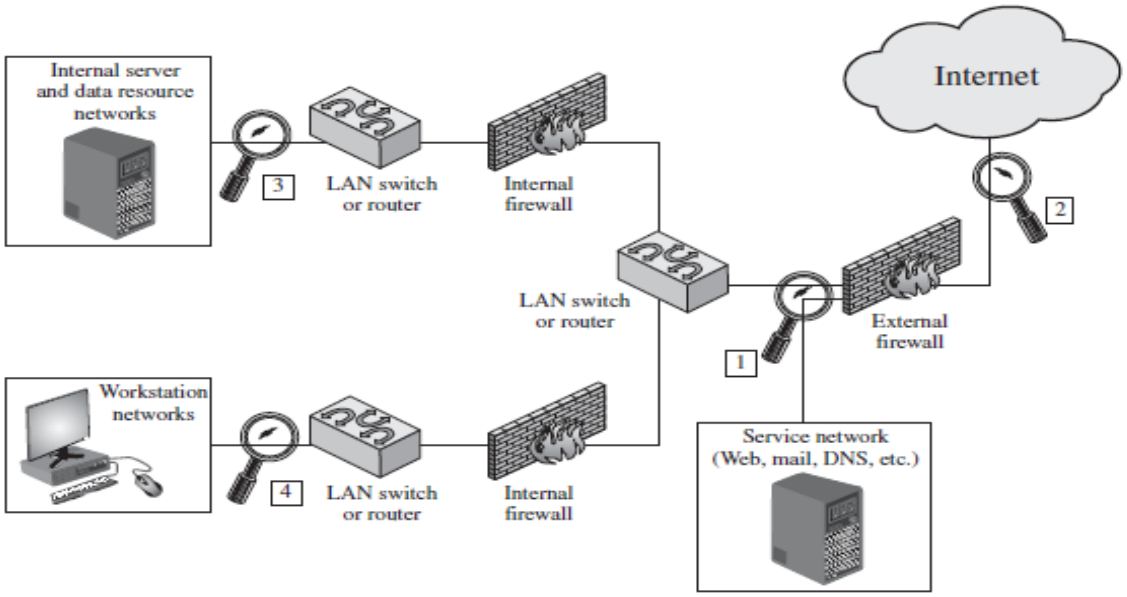


Figure 8.5 **Example of NIDS Sensor Deployment**

Instead of placing a NIDS sensor inside the external firewall, the security administrator may choose to place a NIDS sensor between the external firewall and the Internet or WAN (**location 2**). In this position, the sensor can monitor all network traffic, unfiltered. The advantages of this approach are as follows:

· Documents number of attacks originating on the Internet that target the network

· Documents types of attacks originating on the Internet that target the network

A sensor at location 2 has a higher processing burden than any sensor located elsewhere on the site network. In addition to a sensor at the boundary of the network, on either side of the external firewall, the administrator may configure a firewall and one or more sensors to protect major backbone networks, such as those that support internal servers and database resources (**location 3**). The benefits of this placement include the following:

· Monitors a large amount of a network's traffic, thus increasing the possibility of spotting attacks

· Detects unauthorized activity by authorized users within the organization's security perimeter

Thus, a sensor at location 3 is able to monitor for both internal and external attacks. Because the sensor monitors traffic to only a subset of devices at the site, it can be tuned to specific protocols and attack types, thus reducing the processing burden. Finally, the network facilities at a site may include separate LANs that support user workstations and servers specific to a single department. The administrator could configure a firewall and NIDS sensor to provide additional protection for all of these networks or target the protection to critical subsystems, such as personnel and financial networks (**location 4**). A sensor used in this latter fashion provides the following benefits:

· Detects attacks targeting critical systems and resources

· Allows focusing of limited resources to the network assets considered of greatest value

As with a sensor at location 3, a sensor at location 4 can be tuned to specific protocols and attack types, thus reducing the processing burden.

Intrusion Detection Techniques

As with host-based intrusion detection, network-based intrusion detection makes use of signature detection and anomaly detection.

SIGNATURE DETECTION The following are types of attacks that are suitable for signature detection:

· **Application layer reconnaissance and attacks:** Most NIDS technologies analyze several dozen application protocols. Commonly analyzed ones include Dynamic Host Configuration Protocol (DHCP), DNS, Finger, FTP, HTTP, Internet Message Access Protocol (IMAP), Internet Relay Chat (IRC), Network File System (NFS), Post Office Protocol (POP), rlogin/rsh, Remote Procedure Call (RPC), Session Initiation Protocol (SIP), Server Message Block (SMB), SMTP, SNMP, Telnet, and Trivial File Transfer Protocol (TFTP), as well as database protocols, instant messaging applications, and peer-to-peer file sharing software. The NIDS is looking for attack patterns that have been identified as targeting these protocols. Examples of attack include buffer overflows, password guessing, and malware transmission.

· **Transport layer reconnaissance and attacks:** NIDSs analyze TCP and UDP traffic and perhaps other transport layer protocols. Examples of attacks are unusual packet fragmentation, scans for vulnerable ports, and TCP-specific attacks such as SYN floods.

· **Network layer reconnaissance and attacks:** NIDSs typically analyze IPv4, ICMP, and IGMP at this level. Examples of attacks are spoofed IP addresses and illegal IP header values.

· **Unexpected application services:** The NIDS attempts to determine if the activity on a transport connection is consistent with the expected application protocol. An example is a host running an unauthorized application service.

· **Policy violations:** Examples include use of inappropriate Web sites and use of forbidden application protocols.

ANOMALY DETECTION TECHNIQUES The following are types of attacks that are suitable for anomaly detection:

· **Denial-of-service (DoS) attacks:** Such attacks involve either significantly increased packet traffic or significantly increase connection attempts, in an attempt to overwhelm the target system. These attacks are analyzed in Chapter

7 . Anomaly detection is well suited to such attacks.

· **Scanning :** A scanning attack occurs when an attacker probes a target network or system by sending different kinds of packets. Using the responses received from the target, the attacker can learn many of the system's characteristics and vulnerabilities. Thus, a scanning attack acts as a target identification tool for an attacker. Scanning can be detected by atypical flow patterns at the application layer (e.g., banner grabbing 3), transport layer (e.g., TCP and UDP port scanning), and network layer (e.g., ICMP scanning).

· **Worms:** Worms 4 spreading among hosts can be detected in more than one way. Some worms propagate quickly and use large amounts of bandwidth. Worms can also be detected because they can cause hosts to communicate with each other that typically do not, and they can also cause hosts to use ports that they normally do not use. Many worms also perform scanning.

Logging of Alerts

When a sensor detects a potential violation, it sends an alert and logs information related to the event. The NIDS analysis module can use this information to refine intrusion detection parameters and algorithms. The security administrator can use this information to design prevention techniques. Typical information logged by a NIDS sensor includes the following:

· Timestamp (usually date and time)

· Connection or session ID (typically a consecutive or unique number assigned to each TCP connection or to like groups of packets for connectionless protocols)

· Event or alert type

· Rating (e.g., priority, severity, impact, confidence)

· Network, transport, and application layer protocols

· Source and destination IP addresses

· Source and destination TCP or UDP ports, or ICMP types and codes

· Number of bytes transmitted over the connection

· Decoded payload data, such as application requests and responses

· State-related information (e.g., authenticated username)

NIDS	HIDS
Broad in scope (watches all network activities)	Narrow in scope (watches only specific host activities)
Easier setup	More complex setup
Better for detecting attacks from the outside	Better for detecting attacks from the inside
Less expensive to implement	More expensive to implement
Detection is based on what can be recorded on the entire network	Detection is based on what any single host can record
Examines packet headers	Does not see packet headers
Near real-time response	Usually only responds after a suspicious log entry has been made
OS-independent	OS-specific
Detects network attacks as payload is analyzed	Detects local attacks before they hit the network
Detects unsuccessful attack attempts	Verifies success or failure of attacks

7. Information Sources

An **Intrusion Detection System (IDS)** is only as effective as the data it receives. To detect and respond to suspicious activity, an IDS gathers data from multiple **information sources**, which act as the eyes and ears of the system. These sources provide critical input about system behavior, network traffic, application usage, and security events. Based on this information, the IDS analyzes activity to identify potential intrusions, policy violations, or malicious actions.

Below is a detailed explanation of the **main types of information sources** used in IDS:

1. Host-Based Information Sources

These sources are found on individual computers or servers. A **Host-Based IDS (HIDS)** uses this data to detect changes or suspicious activity at the system level.

Common sources include:

- ◇ **System logs** – Records of user logins, command history, or system events.
- ◇ **Audit trails** – Tracks user activities, such as file access or administrative actions.
- ◇ **File integrity checking tools** – Detects unauthorized changes to system or config files (e.g., Tripwire).
- ◇ **Process monitoring** – Observes running processes and memory usage.

- ◇ **Registry changes** – On Windows systems, changes to the registry can indicate malware activity.

Example:

If a critical system file is modified without authorization, the HIDS detects and flags the change.

2. Network-Based Information Sources

These sources capture and analyze data from the **network layer**. A **Network-Based IDS (NIDS)** monitors real-time network traffic and communication between systems.

Common sources include:

- ◇ **Packet captures (PCAP)** – Full or partial snapshots of network traffic for in-depth inspection.
- ◇ **Firewall logs** – Shows allowed or blocked traffic, source and destination IPs, and port access.
- ◇ **Router and switch logs** – Records routing behavior and traffic patterns.
- ◇ **NetFlow or sFlow data** – Summarized traffic statistics from routers.
- ◇ **DNS logs** – Helps detect domain spoofing or exfiltration via DNS tunnels.

Example:

If a large number of failed login attempts are detected from a single IP, the NIDS can raise an alert for a brute-force attack.

3. Application-Based Information Sources

These sources are specific to **software applications** and services running on a system. An **Application Protocol-Based IDS (APIDS)** focuses on detecting abnormal behavior in application-level protocols.

Common sources include:

- ◇ **Web server logs (Apache, Nginx)** – Tracks page visits, errors, and request patterns.
- ◇ **Database access logs** – Records queries and transactions.
- ◇ **Email server logs** – Useful for detecting spam, phishing, or malware-laden emails.

- ◇ **Authentication logs** – Records user login attempts, failures, and session durations.

Example:

If a user sends thousands of requests with SQL syntax to a web server, the APIDS may flag this as a SQL injection attempt.

4. Security Device and System Logs

Modern IDS implementations often integrate with other **security systems** to correlate events and gain broader visibility.

Common sources include:

- ◇ **Antivirus and EDR logs** – Detect malware infections or suspicious user behavior.
- ◇ **Firewall and VPN logs** – Monitor perimeter activity and remote access attempts.
- ◇ **SIEM (Security Information and Event Management) tools** – Aggregate logs from various systems for correlation and alerting.
- ◇ **Intrusion Prevention System (IPS) logs** – May provide dropped packet info or real-time blocking data.

Example:

A combination of VPN logs showing login from an unusual country, and antivirus logs indicating malware, could trigger a high-priority alert.

Information sources play a crucial role in determining the effectiveness of an Intrusion Detection System (IDS). The type and quality of data fed into an IDS directly influence what the system can observe and how accurately it can detect threats. The more diverse and relevant the sources—ranging from host-based logs and network traffic to application-level data and security device outputs—the better the IDS can perform. High-quality, comprehensive data helps minimize false positives and false negatives, ensuring that genuine threats are identified without overwhelming the system with noise. Furthermore, integrating multiple information sources provides a broader context, allowing the IDS to detect complex or multi-stage attacks that might otherwise go unnoticed. In essence, host-based sources reveal internal system changes, network sources monitor communication patterns, application sources expose misuse of services, and

security devices contribute additional layers of validation. Together, these information sources form the backbone of a robust and intelligent intrusion detection strategy.

Lab Exercises

1. Install Snort and configure it to monitor network traffic.
2. Deploy Snort as a Network Intrusion Detection System (NIDS).

Lecture Slides

Lecture Slides
https://drive.google.com/drive/folders/1MdIU8YLJpG_n3byErvlaSDCn4JTDGb23?usp=drive_link

Assignment

Assignment

e. Given a simple network scenario (e.g., office with users, servers, internet)

- Draw the network.
  - Identify where to place HIDS and NIDS.
  - Justify their placement.

f. Create a poster (physical or digital) on one IDS topic:

- Types of IDS
- IDS vs IPS
- IDS Architecture
  - Choose two known cyberattacks (e.g., SQL injection and insider data theft). Explain which technique (misuse or anomaly) would detect them better and why.

Part A Q & A

PART -A

1. Define an Intrusion Detection System (IDS). (CO1, K1 – Remember)

An IDS is a security mechanism that monitors system or network activities for signs of possible incidents or violations. It helps detect malicious actions like unauthorized access, policy violations, and system misuse.

2. List two main types of IDS. (CO1, K1 – Remember)

The two primary types of IDS are Host-Based IDS (HIDS), which monitors individual system activities, and Network-Based IDS (NIDS), which analyzes traffic across the network.

3. State any two goals of IDS. (CO1, K1 – Remember)

The main goals of IDS include detecting intrusions accurately, providing alerts or responses, and maintaining accountability through logging and auditing.

4. What is the primary function of an IDS? (CO1, K2 – Understand)

The core function of an IDS is to detect unauthorized or suspicious activities in systems or networks. It aids in identifying breaches before significant damage occurs.

5. Differentiate between HIDS and NIDS. (CO1, K2 – Understand)

HIDS monitors events on individual hosts like file changes or logins, while NIDS inspects network traffic to detect malicious communication across systems.

6. Mention any two internal threats to data. (CO1, K1 – Remember)

Internal threats include disgruntled employees who misuse access and former staff who retain unauthorized credentials. They often have inside knowledge of the system.

7. Explain the term 'structured threat' with an example. (CO1, K2 – Understand)

Structured threats are well-planned attacks by skilled individuals or groups. For example, the Stuxnet worm was a structured attack targeting nuclear facilities.

8. What is a false positive in IDS? (CO1, K1 – Remember)

A false positive occurs when benign or legitimate activity is incorrectly flagged as a threat. This can overwhelm analysts with unnecessary alerts.

9. What is meant by audit logs in IDS? (CO1, K1 – Remember)

Audit logs are system-generated records of events like logins, file access, or configuration changes. They are essential for detecting and investigating security incidents.

10. State the function of E-box in IDS architecture. (CO1, K1 – Remember)

The E-box, or Event Generator, processes raw data and creates events based on observed activity. It acts as the initial point of event detection in IDS.

11. Define anomaly detection. (CO1, K1 – Remember)

Anomaly detection identifies deviations from normal system behavior. It helps detect unknown attacks that do not match predefined patterns or signatures.

12. Name two audit record fields. (CO1, K1 – Remember)

Common fields in an audit record include the subject (user or process) and the action (event like login, access, etc.). These help track user behavior.

13. Give two examples of masqueraders. (CO1, K1 – Remember)

Masqueraders might use stolen login credentials or spoof user identities to access systems unlawfully, pretending to be legitimate users.

14. Explain the term 'clandestine user'. (CO1, K2 – Understand)

A clandestine user is one who gains unauthorized, often high-level access and hides their presence by disabling logs or bypassing controls.

15. List any two requirements of an effective IDS. (CO1, K1 – Remember)

An IDS must run continuously without interruptions and be resistant to subversion. It should operate with minimal human supervision.

16. Mention any two advantages of misuse detection. (CO1, K1 – Remember)

Misuse detection is accurate for known attacks and can generate low false positives. It is reliable in detecting threats with known signatures.

17. What is the role of the D-box in IDS? (CO1, K1 – Remember)

The D-box stores generated events and logs for future reference. It helps in postmortem analysis after an intrusion is detected.

18. Explain 'profile-based anomaly detection'. (CO1, K2 – Understand)

This approach builds a behavioral profile of normal activity and flags significant deviations. It is used to identify unknown or insider threats.

19. What is the base-rate fallacy in IDS? (CO1, K2 – Understand)

It refers to the difficulty in maintaining low false alarms even with high detection accuracy, due to the rarity of real attacks in large datasets.

20. How does IDS assist in forensic analysis? (CO1, K2 – Understand)

IDS logs record events that help reconstruct attack timelines. These records are crucial in investigating and responding to breaches.

21. State the importance of real-time IDS. (CO1, K2 – Understand)

Real-time IDS enables immediate detection and alerts, allowing rapid responses to ongoing attacks and minimizing damage.

22. Give two examples of unstructured threats. (CO1, K1 – Remember)

Examples include a teenager launching a DoS attack using a free tool or an unskilled user running malware scripts found online.

23. Name two examples of audit tools. (CO1, K1 – Remember)

Tripwire is used for file integrity checking, and the ELK Stack (Elasticsearch, Logstash, Kibana) is used for log analysis and monitoring.

24. What is interval-based IDS? (CO1, K1 – Remember)

It analyzes collected data in batches or intervals rather than in real time. It is suitable for environments where instant alerts are less critical.

25. What is a credentialed vulnerability scan? (CO1, K2 – Understand)

It is a scan done with valid login credentials, providing deeper access to assess vulnerabilities inside the system, not just from the network view.

26. Define a 'signature' in misuse detection. (CO1, K1 – Remember)

A signature is a predefined pattern that matches known attack behavior. IDS uses these signatures to identify specific threats.

27. Mention two types of responses in IDS. (CO1, K1 – Remember)

IDS responses can be passive (generating alerts) or active (blocking connections, resetting sessions).

28. What is a misuse detector? (CO1, K1 – Remember)

A misuse detector matches monitored activity against known attack signatures to detect and report intrusions.

29. List any two benefits of IDS audit. (CO1, K1 – Remember)

An IDS audit helps improve detection accuracy and ensures compliance with security policies and standards.

30. What are host-based information sources? (CO1, K1 – Remember)

These include logs from user login records, file system changes, and system calls generated on individual hosts.

31. What is the purpose of the A-box in IDS? (CO1, K1 – Remember)

The A-box is the Analyzer that evaluates collected events and generates alerts if suspicious patterns are found.

32. Name two IDS detection techniques. (CO1, K1 – Remember)

The two techniques are misuse detection (based on signatures) and anomaly detection (based on behavior deviations).

33. What is an R-box in IDS architecture? (CO1, K1 – Remember)

The R-box handles the system's response to detected intrusions, such as sending alerts or executing defense actions.

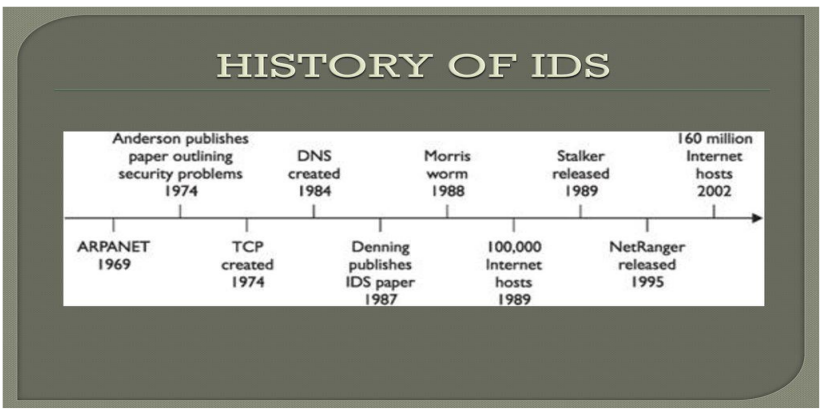
34. List two disadvantages of anomaly detection. (CO1, K1 – Remember)

It can produce a high number of false positives and requires a large amount of normal data for training and profiling.

35. Explain threshold detection. (CO1, K2 – Understand)

Threshold detection flags an alert when behavior (e.g., failed logins) exceeds a predefined limit, indicating possible intrusion.

36. Illustrate the evolution/history of Intrusion Detection Systems. (CO1,K1–Remember)



37. Compare internal vs external vs structured vs unstructured threats(CO1,K2-Understand).

Category	Description	Example
Internal Threat	Comes from inside the organization	Employee stealing data
External Threat	Comes from outside the organization	Hacker phishing attack
Structured Attack	Well-planned and targeted	Nation-state ransomware
Unstructured Attack	Random and unskilled	Amateur trying DDoS tools

38. Give Some Examples of Intruder Patterns of Behavior (CO1,K2-Understand)

(a) Hacker

1. Select the target using IP lookup tools such as NSLookup, Dig, and others.
2. Map network for accessible services using tools such as NMAP.
3. Identify potentially vulnerable services (in this case, pcAnywhere).
4. Brute force (guess) pcAnywhere password.
5. Install remote administration tool called DameWare.
6. Wait for administrator to log on and capture his password.
7. Use that password to access remainder of network.

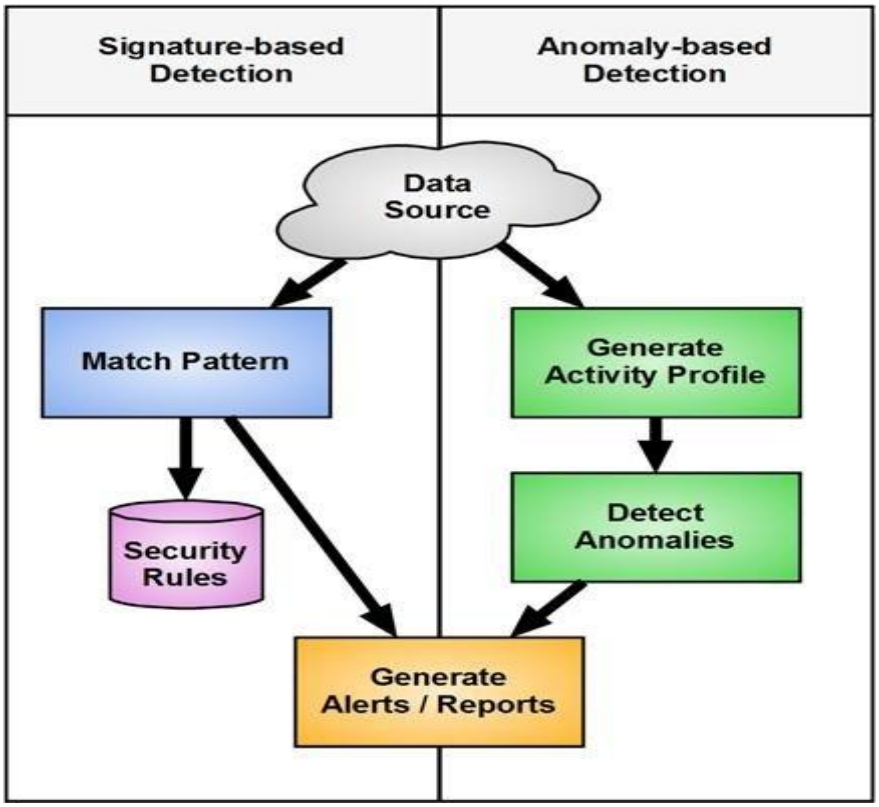
(b) Criminal Enterprise

1. Act quickly and precisely to make their activities harder to detect.
2. Exploit perimeter through vulnerable ports.
3. Use Trojan horses (hidden software) to leave back doors for reentry.
4. Use sniffers to capture passwords.
5. Do not stick around until noticed.
6. Make few or no mistakes.

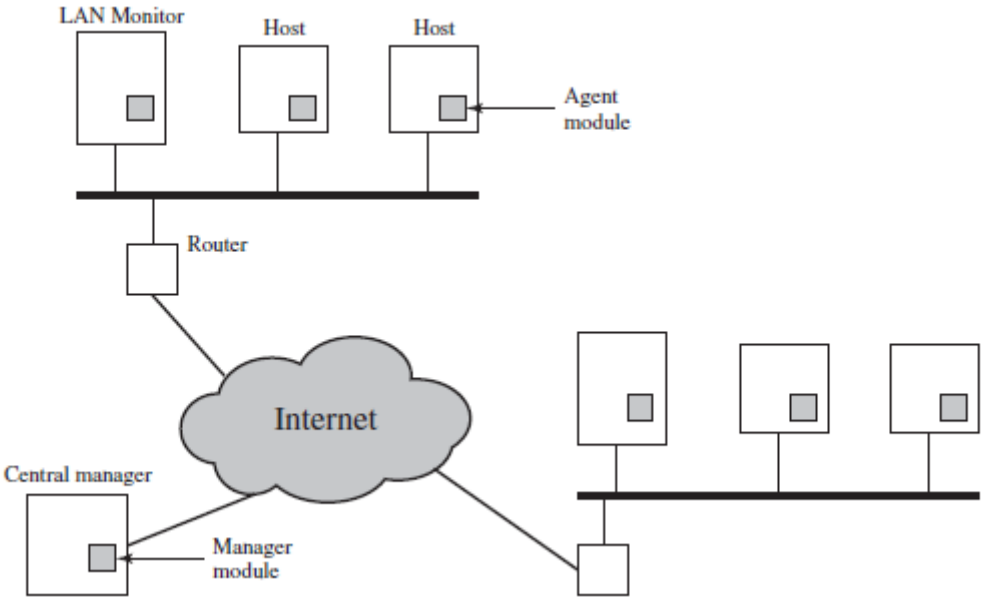
(c) Internal Threat

1. Create network accounts for themselves and their friends.
2. Access accounts and applications they wouldn't normally use for their daily jobs.
3. E-mail former and prospective employers.
4. Conduct furtive instant-messaging chats.
5. Visit Web sites that cater to disgruntled employees, such as f'dcompany.com.
6. Perform large downloads and file copying.
7. Access the network during off hours.

39. Compare Signature vs Anamaly based detection through illustration (CO1,K2-Understand)



40. Illustrate the architecture of Distributed IDS(CO1,K2-Understand)



Part B Q

PART -B

1. Analyze the differences between Host-based IDS and Network-based IDS. Explain how their information sources affect detection efficiency(CO1,K4).

2. Evaluate the effectiveness of anomaly detection and misuse detection techniques in modern intrusion detection systems. Justify your recommendation for a hybrid approach(CO1,K4).

3. Design a distributed IDS architecture for a university campus network. Justify your design decisions with respect to scalability, audit record handling, and response strategy(CO1,K4).

4. Critically analyze the role of IDS audit in improving organizational security. How does it help reduce false positives and maintain compliance? (CO1,K4)

5. Design an IDS alert escalation policy for an organization. Include thresholds, alert types, and suitable responses. Justify your choices. (CO1,K4)

6. Compare centralized, partially distributed, and fully distributed control strategies for IDS. Which is suitable for cloud-based environments? Justify(CO1,K4).

7. Analyze the CIDF (Common Intrusion Detection Framework) architecture and evaluate the interaction between its components (E-box, A-box, D-box, and R-box). How do they collectively contribute to intrusion response? (CO1,K4)

8. Using a given attack scenario, map how different types of IDS (HIDS, NIDS, APIDS) would respond at each stage of the intrusion lifecycle. Highlight strengths and limitations in your analysis(CO1,K4).

Supportive Online Certification courses

R.M.K GROUP OF INSTITUTIONS

SUPPORTIVE ONLINE COURSES

S No	Course provider	Course title	Link
1	Udemy	The Complete Cyber Security Course : Network Security	https://www.udemy.com/course/network-security-course/?couponCode=ST12MT90625AI
2	Udemy	Snort Intrusion Detection, Rule Writing, and PCAP Analysis	https://www.udemy.com/course/snort-intrusion-detection-rule-writing-and-pcap-analysis/?couponCode=ST12MT90625AI
3	NPTEL	Network Security	https://onlinecourses.nptel.ac.in/noc25_ee54/preview

Real life Applications in day to day life and to Industry

REAL TIME APPLICATIONS IN DAY TO DAY LIFE AND TO INDUSTRY

Industry	IDS Application
Healthcare	Monitoring unauthorized access to patient data and connected devices
Banking	Detecting fraud, phishing links, and abnormal access patterns
Education	Preventing student misuse of lab systems and safeguarding resources
E-Commerce	Monitoring traffic for account takeovers and fraud attempts
Manufacturing	Securing robotic process controls and industrial IoT systems
Smart Cities	Detecting anomalies in traffic management, CCTV, and power systems

Content beyond Syllabus

Contents beyond the Syllabus

Deception Technologies in Intrusion Detection
https://www.zscaler.com/resources/security-terms-glossary/what-is-deception-technology

Assessment Schedule

FIAT: 14.08.2025
SIAT: 23.09.2025
Model Exam: 28.10.2025

R.M.K
GROUP OF
INSTITUTIONS

Prescribed Text books & Reference books

PRESCRIBED TEXT BOOKS AND REFERENCE BOOKS

TEXT BOOKS

1. Rafeeq Rehman : “Intrusion Detection with SNORT, Apache, MySQL, PHP and ACID,” 1st Edition, Prentice Hall , 2003.

2. Internet Security: A Hands-on Approach, by Wenliang Du, Third Edition, 2019

REFERENCE BOOKS

1. Christopher Kruegel, Fredrik Valeur, Giovanni Vigna: “Intrusion Detection and Correlation Challenges and Solutions”, 1st Edition, Springer, 2005.
2. Carl Endorf, Eugene Schultz and Jim Mellander “Intrusion Detection & Prevention”, 1st Edition, Tata McGraw-Hill, 2004.
3. Stephen Northcutt, Judy Novak : “Network Intrusion Detection”, 3rd Edition, New Riders Publishing, 2002.
4. T. Fahringer, R. Prodan, “A Text book on Grid Application Development and Computing Environment”. 6th Edition, Khanna Publishers, 2012.

Mini Project Suggestions

R.M.K GROUP OF INSTITUTIONS

MINI PROJECT SUGGESTIONS

1. IDS Log Analyzer using Python:
Develop a tool to read and analyze IDS logs (e.g., from Snort) to identify patterns such as port scans or repeated login failures. This project introduces students to basic log analysis and alert identification using Python.

2. Threat Classification Tool:
Design a tool (Excel-based or simple Python GUI) that classifies threats as internal or external, structured or unstructured, based on user input or predefined cases.

3. IDS Alert Visualization Dashboard:
Create a visual dashboard using Streamlit or Flask that reads IDS alert logs and displays charts showing top alert sources, frequent alert types, and timelines.

Thank you