A SYNOPSIS ON

INTRUSION DETECTION SYSTEM

Submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

Computer Science & Engineering

Submitted by:

Divyanshu Tewari	**2261198**
Anuj Bhatt	**2261108**
Sagar Joshi	**2261498**
Gokul Chopra	**2261233**

Under the Guidance of

Mr. Prince Kumar

Assistant Professor

Project Team ID: 48

Department of Computer Science & Engineering

Graphic Era Hill University, Bhimtal, Uttarakhand

March-2025

Graphic Era Hill University
BHIMTAL CAMPUS

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the Synopsis entitled **"Intrusion Detection System"** in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science & Engineering of the Graphic Era Hill University, Bhimtal campus and shall be carried out by the undersigned under the supervision of **Mr. Prince Kumar, Assistant Professor**, Department of Computer Science & Engineering, Graphic Era Hill University, Bhimtal.

Divyanshu Tewari	2261198
Anuj Bhatt	2261108
Sagar Joshi	2261498
Gokul Chopra	2261233

The above mentioned students shall be working under the supervision of the undersigned on the **"Intrusion Detection System"**

Signature

Supervisor

Signature

Head of the Department

Internal Evaluation (By DPRC Committee)

Status of the Synopsis: Accepted / Rejected

Any Comments:

Name of the Committee Members: **Signature with Date**

1.

2.

Table of Contents

Chapter 1

Introduction

In the modern digital era, computer systems and networks are highly susceptible to various security threats such as malware, unauthorized access, privilege escalation, and zero-day attacks. With the growing complexity and interconnectivity of systems, traditional defence mechanisms like firewalls and antivirus software are often insufficient to combat sophisticated attacks. Therefore, there is an increasing need for more intelligent and adaptive security solutions that can detect both known and unknown threats effectively.

An **Intrusion Detection System (IDS)** is a crucial component of modern cybersecurity frameworks. It monitors system activities and detects potential security breaches by analysing patterns of behaviour or known attack signatures. Among various types of IDS, a **Host-Based Intrusion Detection System (HIDS)** focuses specifically on monitoring and analysing activities within a single host (e.g., a computer or server). It leverages logs, file integrity, system calls, and user behaviour to detect anomalies.

This project proposes the development of a **Hybrid HIDS** that combines **signature-based detection** (for known attacks) and **anomaly-based detection** (for unknown or zero-day attacks). The system is designed using fundamental **operating system concepts** such as process management, system calls, and user activity logging. It also integrates **machine learning algorithms** to detect abnormal patterns in system behaviour, thereby improving detection accuracy and reducing false positives.

The objective of this project is to provide a lightweight, efficient, and extensible HIDS solution that can be integrated into existing systems for enhanced security monitoring. The hybrid approach not only enables rapid detection of well-known threats but also equips the system with adaptive intelligence to identify new and emerging security risks.

Problem Statement

In the current digital era, computer systems and networks are increasingly vulnerable to a wide range of security threats, including malware attacks, unauthorized access and data breaches. Despite the deployment of firewalls and antivirus software, many systems remain exposed to sophisticated and evolving cyber-attacks that bypass traditional security mechanisms.

Challenges-

1. **Detection of Unknown or Zero-Day Attacks**
    - Signature-based IDS fail to detect attacks that do not match known patterns or signatures.
2. **High False Positive Rate in Anomaly Detection**
    - Anomaly-based systems often raise false alarms for legitimate user behaviour.
3. **Real-Time Monitoring of System Activities**
    - Continuous tracking of process execution, file access, and system calls without lag is technically complex.
4. **Ensuring Low System Overhead**
    - The HIDS must not degrade system performance significantly during real-time monitoring.
5. **Efficient Alert Management**
    - Generating too many alerts can overwhelm system administrators and reduce response effectiveness.

Addressing the Challenges

1. **Hybrid Detection Mechanism**
    - Combine signature-based and anomaly-based techniques to detect both known and unknown attacks.
2. **Advanced Anomaly Filtering**
    - Use machine learning models and behaviour profiling to reduce false positives.
3. **Efficient Monitoring Tools**
    - Leverage OS tools like auditd, sysmon, or inotify for lightweight and real-time system tracking.
4. **Smart Alerting and Visualization**
    - Prioritize and aggregate alerts, and provide dashboards or logs for better usability.

Chapter 2

Background/ Literature Survey

With the advancement of technology and the growing reliance on computer systems, security threats have become increasingly complex and dangerous. Traditional defence mechanisms such as firewalls, antivirus programs, and access control systems, although essential, are often not sufficient to combat modern cyberattacks that use stealthy, polymorphic, and zero-day techniques. This has led to increased interest in Intrusion Detection Systems (IDS) as a second line of defence.

---

1. Evolution of IDS

The concept of IDS was first introduced by James P. Anderson in 1980, focusing on monitoring audit logs for abnormal user behaviour. Since then, IDS has evolved to include a variety of techniques and mechanisms that can monitor system activities and detect both known and unknown intrusions.

There are two primary types of IDS:

- **Network-based IDS (NIDS):** Monitors and analyses network traffic.
- **Host-based IDS (HIDS):** Operates on individual systems and observes operating system activities like process behaviour, system calls, and file access**.**

Modern IDS systems are often hybrid, combining both types to improve detection accuracy.

---

2. Detection Techniques in IDS

- **Signature-Based Detection:**
  - Compares observed activity against known attack patterns or signatures.
  - Fast and accurate for known attacks.
  - Ineffective against unknown or zero-day threats.
- **Anomaly-Based Detection:**
  - Establishes a baseline of normal behaviour and flags deviations.
  - Capable of detecting novel attacks.
  - Suffers from high false positive rates.
- **Hybrid Detection:**
  - Combines both signature and anomaly-based methods.
  - Aims to improve detection accuracy while reducing false positives.
  - Requires effective integration and tuning to balance both techniques.

3. Role of Operating System Concepts in HIDS

HIDS utilizes various OS-level features to identify malicious activity**:**

OS Concept	IDS Usage
System Calls	Monitor file I/O, network access, and process actions.
Process Table	Detect suspicious or unknown processes.
File System	Check for unauthorized modifications or hidden files.
User Sessions	Observe failed logins, privilege escalation attempts.
Logs & Audits	Analyse logs for malicious patterns or repeated errors.

4. Previous Research and Studies

- Snort (Roesch, 1999): A widely-used open-source NIDS that introduced signature-based detection at the packet level.
- Bro/Zeek IDS: Supports anomaly detection and real-time analysis of network traffic.
- Research by Forrest et al. (1996): Introduced system call-based anomaly detection for UNIX systems, using short sequences of calls to detect intrusions.
- Host-based IDS using Machine Learning (Various Studies): Many recent works apply ML models (SVM, Random Forest, Deep Learning) to classify system behaviour as benign or malicious based on logs or syscall patterns.

5. Gaps and Challenges

- High false positives in anomaly-based systems.
- Poor detection of unknown attacks in signature-based systems.
- Performance degradation due to constant monitoring.
- Difficulty in real-time detection and automated response.

6. Motivation for the Project

Given the above challenges, there is a need to develop a light, real-time, hybrid HIDS that:

- Uses OS-level data (like system calls and file access patterns).
- Combines both signature and anomaly detection techniques.
- Can trigger alerts or responses automatically upon detecting intrusions.

Chapter 3

Objectives

The objectives of the proposed work are as follows:

1. **Develop a Real-Time Host-Based IDS:**
    - Design an IDS that monitors system activities (processes, files, network, logs) in real time.
    - Detect suspicious system behaviours and unauthorized access attempts.
2. **Implement a Hybrid Detection Approach:**
    - Combine signature-based detection for known attacks and anomaly-based detection for unknown threats.
    - Reduce false positives while ensuring accurate intrusion detection.
3. **Utilize OS-Level Monitoring for Detection:**
    - Track system calls, file system modifications, process behaviour, and user authentication logs.
    - Use OS auditing tools and log analysis for intrusion detection.
4. **Ensure Lightweight and Efficient Performance:**
    - Optimize monitoring techniques to minimize system overhead and prevent resource exhaustion.
    - Implement efficient algorithms for real-time detection and alerting.
5. **Enhance Security with Automated Responses:**
    - Implement alert mechanisms (log alerts, email notifications, system lockdown).
    - Support automatic responses like blocking malicious processes or users.

Chapter 4

Hardware and Software Requirements

4.1 Hardware Requirements

Sl. No	Name of the Hardware	Specification
1.	Processor	Intel Core i5 / AMD Ryzen 5 or higher
2.	RAM	Minimum 8GB (Recommended: 16GB)
3.	Storage	Minimum 256GB SSD (Recommended: 512GB)
4.	Network Interface Card (NIC)	Gigabit Ethernet / Wi-Fi Adapter

4.2 Software Requirements

Sl. No	Name of the Software	Specification
1.	Operating System	Linux (Ubuntu 20.04+ / CentOS 7+) or Windows 10/11
2.	Programming Language	Python 3.8+ / C++ / Java (for system monitoring)
3.	IDS Framework	OSSEC, Snort, Suricata (for reference & integration)
4.	Database	MySQL / PostgreSQL (for logging intrusion data)
5.	Machine Learning Libraries	Scikit-learn, TensorFlow, PyTorch (for anomaly detection)
6.	IDE / Code Editor	PyCharm, VS Code, or Eclipse (for development)

Chapter 5

Possible Approach/ Algorithms

1. Signature-Based Approach (Misuse Detection)

- Compares incoming data (logs, system calls, network traffic) against a database of **known attack patterns (signatures)**.
- Works like an **antivirus**, detecting threats with **predefined rules**.
- Example: **Snort**, **OSSEC**, and **Suricata** use this approach.

➢ **Algorithms Used**

Algorithm	**Functionality**
Pattern Matching Algorithms	Detects attack patterns in logs/system calls using string matching.
Aho-Corasick Algorithm	Efficient multi-pattern string searching. Used in Snort IDS.
Boyer-Moore Algorithm	Fast substring search used in intrusion detection.

Table 4.1 Implementation Example (Python - Simple Signature Matching)

```
import re
# Predefined attack signatures
signatures = [
    "rootkit",
    "malware",
    "unauthorized access",
    "buffer overflow",
]
def signature_detection(log_data):
    for signature in signatures:
        if re.search(signature, log_data, re.IGNORECASE):
```

```
            print(f"Alert: Suspicious activity detected - {signature}")
            return True
    return False

# Sample log data
log_entry = "User attempted unauthorized access to /etc/shadow"
signature_detection(log_entry)
```

2. Anomaly-Based Approach (Machine Learning & Statistical Methods)

- Learns normal system behaviour and flags anything unusual.
- Detects zero-day and unknown attacks.
- Uses machine learning (ML) and statistical analysis.

➢ **Algorithms Used**

Algorithm	**Functionality**
Z-Score / Standard Deviation	Identifies values outside normal ranges.
Chi-Square Test	Compares observed vs. expected behaviors.
Support Vector Machine (SVM)	Classifies normal vs. abnormal activities.
Random Forest (RF)	Uses multiple decision trees for attack classification.
K-Nearest Neighbors (KNN)	Compares new data with past observations.
Naïve Bayes	Probabilistic detection of anomalies.
Artificial Neural Networks (ANN)	Deep learning model for attack detection.

Table 4.1 Implementation Example (Python - Anomaly Detection with SVM)

```
import re
# Predefined attack signatures
signatures = [
    "rootkit",
    "malware",
    "unauthorized access",
    "buffer overflow",
]
def signature_detection(log_data):
    for signature in signatures:
        if re.search(signature, log_data, re.IGNORECASE):
            print(f"Alert: Suspicious activity detected - {signature}")
            return True
    return False

# Sample log data
log_entry = "User attempted unauthorized access to /etc/shadow"
signature_detection(log_entry)
```

References

I. Debar, H., Dacier, M., & Wespi, A. (1999). Towards a taxonomy of intrusion-detection systems. *Computer Networks*, *31*(8), 805–822. https://doi.org/10.1016/s1389-1286(98)00017-6

II. Scarfone, K. A., & Mell, P. M. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS)*. https://doi.org/10.6028/nist.sp.800-94

III. Denning, D. (1987). An Intrusion-Detection model. *IEEE Transactions on Software Engineering*, *SE-13*(2), 222–232. https://doi.org/10.1109/tse.1987.232894

IV. Baker, A. R., Caswell, B., & Poor, M. (2004). Snort 2.1 intrusion detection. In *Syngress Publishing eBooks*. http://ci.nii.ac.jp/ncid/BB01942289

V. OSSEC – Open Source HIDS . https://www.ossec.net

VI. Scikit-learn Documentation (for ML algorithms). https://scikit-learn.org/stable/

VII. Ghosh, A. K., Schwartzbard, A., & Schatz, M. (1999). Learning program behavior profiles for intrusion detection. *Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring*.

VIII. Bace, R., & Mell, P**.** *Intrusion Detection Systems.* National Institute of Standards and Technology (NIST), 2001. https://csrc.nist.gov/publications/detail/nist-ir/7298/rev-3/final