Project Title: Automated Intrusion Detection system using deep learning for the attack in application layer

Project Context

Project Summary:

This project focuses on developing a robust Intrusion Detection System (IDS) utilizing machine learning (ML) and deep learning (DL) techniques to effectively detect and classify various application layer security threats in modern networks. It uses AI technology like machine learning and deep learning to learn about these attacks and keep us safe online. By doing this, it helps protect our private information, keeps businesses running smoothly, and saves a ton of money by stopping bad things from happening online. However, there are some challenges, like making sure it doesn't accidentally use our personal info to learn, finding enough examples of different types of attacks to teach it, and making sure it can keep up with how fast the internet is growing. Despite these challenges, the main goal is to make the internet a safer place for everyone.

Problem statement:

With the large growing use of internet and web surfing and hosting, Intrusion detection is an area of research in the area of cyber security that bears a lot of potential. Today, at this time many applications are being hosted in the internet and cloud. So there are a lot of potential vulnerabilities in the application as well as the networks. Attacks on the web application are being a hazard for the people and businesses.

Intrusion Detection Systems (IDS) were first introduced by Ames Anderson in 1980, and since then, many different IDS products have been developed to enhance network security. However, with the rapid technological advancements in the past decade, networks have grown significantly in size and can now support numerous applications, resulting in a large amount of important data being generated and exchanged. This growth has made securing these networks, data and applications more challenging due to the emergence of new and evolving attacks. Each network node is vulnerable to security threats, and weaknesses can be exploited.To address these challenges, researchers are exploring the use of machine learning (ML), deep learning (DL), or a combination of both to create effective IDS. These systems need to work in a multi-classification mode to accurately identify and respond to various types of attacks.

Deep learning techniques like CNN and RNN have a decent chance of achieving accurate detection. The model’s training and testing times rely on the computing power available, the complexity of the ML/DL algorithm, the types of datasets used, and the methods used for feature extraction and selection.

Project goals:

1. Develop a multi-classification Intrusion Detection System (IDS) using machine learning (ML) and deep learning (DL) techniques to accurately detect and classify a wide range of applications security threats, including new and evolving attacks.
2. To enhance the performance of a deep learning model used.

Application areas:

1. **Enterprise and Business Networks**: Keeps large company networks safe by spotting and stopping advanced attacks on software applications, protecting important business data and operations.

2. **Cloud Services**: Improves security for online cloud services by detecting and dealing with attacks targeting cloud-based apps, ensuring they remain secure.

3. **Online Shopping Sites**: Protects e-commerce websites from attacks like SQL injection and cross-site scripting (XSS), keeping customer information safe and maintaining customer trust.

4. **Banks and Financial Services**: Secures online banking and financial services by detecting attacks aimed at these systems, ensuring safe and private financial transactions.

5. **Healthcare Systems**: Protects patient data in electronic health records and telemedicine apps from attacks, ensuring privacy and compliance with health regulations.

6. **Government and Defense**: Enhances security for government and defense applications by detecting and stopping threats, protecting sensitive information and national security.

Project justifications:

This project is important because it helps keep everyone safe in the digital world. Here's how:

1. **Protecting People's Information**: By using advanced technology to watch out for sneaky cyber-attacks, like ones targeting online banking or personal health records, this project keeps your private information safe from bad guys.

2. **Keeping Businesses Running Smoothly**: Businesses need their computers and networks to work smoothly without any interruptions. This project helps make sure that happens by stopping cyber-attacks before they can cause any trouble.

3. **Following the Rules**: There are rules and laws about how companies should protect your information. This project helps companies follow those rules, so they don't get in trouble and so you can trust them with your data.

4. **Saving Money**: When bad things happen online, like hackers stealing information or shutting down websites, it can cost a lot of money to fix. This project helps prevent those bad things from happening, saving companies and customers a ton of money.

5. **Giving Experts More Time**: People who work in cybersecurity are really smart, but they can't watch everything all the time. This project helps them out by doing some of the hard work, so they can focus on the really tricky stuff.

6. **Building Trust**: When you know a company takes your security seriously, you trust them more. This project helps build trust between companies and their customers, making everyone feel safer online.

Limitations & Constraints:

1. **Data Privacy Worries**: This project needs a lot of data about how networks work to teach computers to spot bad stuff. But, some of that data might have personal or secret info. Keeping that info safe while still teaching computers what to look for can be tricky.

2. **Finding the Right Data**: Teaching computers to spot bad things means showing them lots of examples of bad stuff happening. But sometimes, it's hard to find enough of those examples, especially for newer types of attacks that haven't happened much yet.

3. **Keep Up with the Fast Pace**: As networks get bigger and busier, the computer needs to be able to keep up with all the activity and still spot the bad stuff quickly. It's like trying to find a needle in a haystack, but the haystack keeps getting bigger and bigger.

4. **Making Compatible with Other Security Tools**: This project needs to work nicely with other security tools already in use, like antivirus software and firewalls. Sometimes, getting them to all work together smoothly can be a bit tricky.

Assumptions:

1. The data used will be enough to predict the attack in the application layer.

Team Member Details:

Name	**Email ID**	**Github ID**
Nayan Pandey	**077bct049.nayan@pcampus.edu.np**	**NayanPandey371**
Niraj Chaurasia	**niraj.ch450@gmail.com**	**Nir-Roman**
Rohit Joshi	**077bei037.rohit@pcampus.edu.np**	**rohitjoshi21**

Requirements

Expected Inputs: Network Traffic Data: Information about how data moves through networks, like logs of what websites are visited or data sent and received.

1. Application Data: Details about how web applications behave, such as the requests they make or the content they display.
2. Attack Examples: Examples of known cyber-attacks, like viruses or hacking attempts, to help the system learn what to look for.
3. Key Features: Important characteristics extracted from the data, like traffic patterns or suspicious activities.
4. Setup Details: Information about how the network is set up and any security rules in place.

Expected outputs:

1. Threat Alerts: Warnings about potential security threats, telling what kind of threat it is and what actions to take.
2. Attack Types: Sorting detected threats into different categories, like viruses or website attacks, to understand what's happening.
3. Performance Measures: Numbers showing how well the system is doing, like how often it catches real threats versus making mistakes.
4. Visualizations: Pictures or graphs showing things like traffic patterns or trends in attacks to help understand the data better.
5. Model Updates: Periodic improvements to the system, like adding new ways to detect threats or training it with more examples.

End-User & Stakeholders Requirements

What task does end-user do with the product

Requirements	Details	Priority	Success Criteria
As a network administrator, I want to receive real-time alerts about security threats detected within the network so that I can take immediate action to mitigate risks. High Alerts are delivered to administrators within seconds of detection, enabling swift response to potential threats.			
As a business owner, I want to view visualizations of network traffic patterns and attack trends so that I can understand the security posture of my organization's network environment.			

Functional Requirements

What is the behavior of the system, what should the system do or support

Requirements	Details	Priority	Success Criteria
Intrusion and Attack detection and alert users	Detect any web application attack and alert the administrator/ users	HIGH	Detect known and unknown web attacks and send alert message
Attack classification	Classify the attack	HIGH	Can classify the attack types

Non-Functional requirements

How well the product must perform in terms of Security, Accuracy, Speed, Aesthetics, Usability, Efficiency, Performance, Maintainability, Error-Checking, …

Requirements	Details	Priority	Success Criteria
Performance	Should detect the intrusion quickly	High	Within 2 seconds

Accuracy	High rate of accurate detection and low rate of false positives and negatives	High	Accuracy of over 95%
Availability	Mostly available	High	Available for 99% of the time

Data collection: Required data /source of the data

Data identification: We will be using the data like:

1. Flow Duration: It provides the information of duration of spoof packet flow in the network. Data type will be in time.
2. Total Fwd Packets: No spoof packets. Data type will be int.
3. Total Length of Fwd Packets: Total no of forwarded packets in the network. Data type is int

Similarly other features are: Flow Bytes/s, Flow Packets/s, FIN Flag Count, RST Flag Count,etc.

Data Sources:

1. **Network Traffic Logs**: Logs generated by network devices such as routers, switches, firewalls, and intrusion detection/prevention systems contain valuable information about network activities, including IP addresses, protocols, ports, and timestamps.

2. **Packet Captures (PCAP)**: Packet capture files provide detailed packet-level information, including headers and payloads, allowing for deep analysis of network traffic. Tools like Wireshark or tcpdump can be used to capture and store network packets for analysis

   .

3. **Security Information and Event Management (SIEM) Systems**: SIEM solutions collect and correlate security-related events from various sources within the network, such as logs from network devices, servers, and applications, providing a centralized source of security data.

4. **Open Source Datasets**: There are publicly available datasets specifically curated for cybersecurity research and training, such as the NSL-KDD dataset, CICIDS2017 dataset, UNSW-NB15 dataset, and DARPA datasets.

5. **Honeypots and Honeynets**: Honeypots are intentionally vulnerable systems deployed within a network to attract and monitor malicious activities. Honeypot data can provide valuable insights into attacker behavior and tactics.

6. **Security Incident Reports**: Historical security incident reports and case studies can serve as valuable sources of real-world attack data, helping to train IDS systems to recognize known attack patterns and techniques.

Data Acquisition methods:

1. **Network Traffic Monitoring**: The primary method for data acquisition involves monitoring network traffic using network monitoring tools or packet capture solutions. These tools passively capture and record network packets traversing the network, providing a comprehensive view of communication patterns, protocols, and potential security threats.
2. **Log Collection**: Logs generated by network devices, servers, and security appliances contain valuable information about system activities, user actions, and security events. Log collection mechanisms, such as syslog servers or log management platforms, can be deployed to aggregate and store log data for analysis.
3. **Honeypots and Honeynets**: Deploying honeypots and honeynets within the network infrastructure can attract and capture malicious activities, providing valuable data for analyzing attacker tactics and techniques. Honeypot data can supplement real-world network traffic data for training intrusion detection models.

Data collection challenges:

1. **Data Privacy Concerns**: One of the primary challenges in data collection is ensuring compliance with data privacy regulations and protecting sensitive information. This includes anonymizing personally identifiable information (PII) and implementing access controls to restrict access to sensitive data.
2. **Access Limitations**: Accessing network traffic and log data from different network segments or remote locations may present logistical challenges. Implementing network monitoring solutions or remote logging mechanisms can help overcome access limitations and ensure comprehensive data collection.
3. **Incomplete Data**: Incomplete or inconsistent data may hinder the effectiveness of intrusion detection models. Addressing this challenge involves implementing data validation and preprocessing techniques to clean and normalize the data, ensuring its quality and reliability for analysis.
4. **Data Volume and Scalability**: Dealing with large volumes of network traffic data and logs can strain storage and processing resources. Implementing scalable data storage and processing solutions, such as distributed databases or cloud-based storage, can help manage the scalability of data acquisition systems.

Benchmarking the Model:

Establishing a baseline model:
The baseline model might incorporate the following components:

1. **Rule-Based Detection**: Basic rules or signatures derived from known attack patterns or network anomalies. For example, rules could include detecting specific patterns indicative of common attacks such as SQL injection or denial-of-service (DoS) attacks.

2. **Threshold-Based Anomaly Detection**: Setting thresholds for various network metrics (e.g., packet rates, connection counts) and triggering alerts when observed values exceed predefined thresholds. This approach helps identify deviations from normal behavior but may not capture subtle or previously unseen attack patterns.

3. **Statistical Analysis**: Basic statistical techniques such as mean, median, standard deviation, or frequency analysis applied to network traffic features. Statistical anomalies beyond certain thresholds may indicate potential security threats.

4. **Signature Matching**: Matching network traffic against known attack signatures or patterns stored in a database or signature repository. This approach relies on signature-based detection mechanisms to identify known threats.

5. **Port and Protocol Analysis**: Analyzing network traffic based on port numbers, protocol headers, or payload content to identify suspicious activities or protocol deviations. For example, unexpected protocol usage or non-standard port usage may indicate potential security issues.