**NEIL GOGTE INSTITUTE OF TECHNOLOGY**

REPORT

(INTELLIGENT LOG ANALYSIS FOR ANOMALY DETECTION)

M. Naga Jahnavi – 245322748105- PH.NO: 9704327003

V. Lakshmi Gayatri - 245322733126- PH.NO: 6305492280

Ch. Nishitha Reddy - 245322748012- PH.NO:9391926528

Ch. Jhansi Sai Sree - 245322748011- PH.NO:7702215199

T. Deekshita Reddy - 245322748117- PH.NO:6304146475

Abstract:

With the rapid expansion of digital infrastructures and the ever-increasing volume of log data generated, the task of identifying anomalies amidst the sea of information has become critical for maintaining the security and efficiency of systems. This abstract presents an overview of intelligent log analysis techniques tailored for anomaly detection, focusing on their significance, challenges, and advancements.

Firstly, the significance of intelligent log analysis lies in its ability to sift through vast amounts of log data to identify deviations from normal behaviour, which could signify potential security breaches, system failures, or operational inefficiencies. Traditional rule-based approaches often struggle to keep pace with the dynamic and complex nature of modern systems, necessitating the adoption of intelligent techniques such as machine learning and artificial intelligence.

However, leveraging machine learning for log analysis presents its own set of challenges. These include the need for labelled datasets for supervised learning, the curse of dimensionality in high-dimensional log data, and the interpretability of models. Researchers have addressed these challenges through techniques such as unsupervised learning, feature engineering, and the development of explainable AI models, enhancing the efficacy and interpretability of anomaly detection systems.

Furthermore, recent advancements in intelligent log analysis have seen the integration of deep learning architectures, natural language processing techniques, and ensemble learning methods to enhance the detection accuracy and adaptability of anomaly detection systems.

In conclusion, intelligent log analysis plays a pivotal role in safeguarding digital infrastructures by enabling the timely detection and mitigation of anomalies. Despite challenges, ongoing research and technological advancements continue to enhance the capabilities of anomaly detection systems, paving the way for more resilient and adaptive cybersecurity measures in an increasingly interconnected world.

Top of Form

Problem Statement:

In the contemporary digital landscape, organizations across various sectors face the daunting challenge of managing and securing vast and complex IT infrastructures. One of the critical aspects of this challenge is the analysis of system logs for anomaly detection. System logs, which record events, activities, and errors within IT systems, contain a wealth of information that can be leveraged to identify abnormal behaviour indicative of security breaches, operational inefficiencies, or potential system failures. However, the sheer volume and complexity of log data make manual analysis impractical and inefficient. Therefore, the development of intelligent log analysis techniques for effective anomaly detection is imperative to address this pressing issue.

**Key Challenges:**

1. Volume and Velocity: Modern IT infrastructures generate massive volumes of log data at high velocities, making it challenging to process and analyse in real-time.

2. Variety and Complexity: Log data come in diverse formats and structures, including textual logs, system logs, network logs, and application logs, each requiring specialized analysis techniques. Additionally, logs may contain noise, inconsistencies, or irrelevant information, complicating the detection of meaningful anomalies.

3. Dynamic Environment: IT environments are dynamic, with system configurations, user behaviours, and network traffic patterns constantly evolving. Anomaly detection systems must adapt to these changes to maintain effectiveness.

4. Labelling and Training Data: Supervised machine learning approaches for anomaly detection require labelled training data, which may be scarce or expensive to obtain. Furthermore, labelling anomalies accurately can be subjective and time-consuming.

5. Interpretability: The interpretability of anomaly detection models is crucial for understanding the reasoning behind detected anomalies and gaining stakeholders' trust in the system. Black-box models may hinder interpretability, posing challenges for system administrators and analysts.

6. False Positives and False Negatives: Anomaly detection systems must strike a balance between minimizing false positives (incorrectly flagging normal behaviour as anomalous) and false negatives (failing to detect actual anomalies), as both can have significant consequences for system security and operational efficiency.

Proposed Solution:

To address these challenges, a comprehensive approach to intelligent log analysis for anomaly detection is required. This approach should leverage a combination of machine learning, deep learning, natural language processing, and data visualization techniques to effectively analyse and interpret log data. Specifically, unsupervised and semi-supervised learning algorithms can be utilized to detect anomalies without the need for extensive labelled data. Feature engineering techniques can help extract relevant information from diverse log formats. Additionally, ensemble learning methods can combine the strengths of multiple algorithms to enhance detection accuracy and robustness. Moreover, efforts should be made to develop explainable AI models that provide transparent insights into the decision-making process of anomaly detection systems, facilitating effective collaboration between automated systems and human analysts.

By addressing these challenges and leveraging advanced intelligent techniques, organizations can develop robust anomaly detection systems capable of safeguarding their IT infrastructures against emerging threats and operational risks.

Research paper:

The research paper titled "Intelligent Log Analysis for Anomaly Detection" by Steven Yen, provides a comprehensive overview of the application of advanced computational techniques for analysing system logs to detect anomalies. Here is a brief summary based on the paper:

The paper begins by highlighting the importance of intelligent log analysis in modern IT environments, where vast amounts of log data are generated continuously. It emphasizes the significance of anomaly detection in identifying security breaches, operational inefficiencies, and potential system failures.

The authors discuss the challenges associated with traditional rule-based approaches to log analysis, which may struggle to keep pace with the dynamic and complex nature of modern systems. They argue for the adoption of intelligent techniques such as machine learning and artificial intelligence to overcome these challenges.

The paper delves into various machine learning algorithms used for log analysis, including supervised, unsupervised, and semi-supervised learning methods. It discusses the importance of labelled datasets for supervised learning, as well as the advantages of unsupervised learning in scenarios where labelled data is scarce or unavailable.

In this research paper, they discussed the results obtained after experimenting on different datasets such as CICIDS 2017, HDFS.

Furthermore, the authors explore the use of deep learning architectures, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), for capturing intricate patterns and temporal dependencies within log sequences. They also discuss ensemble learning methods that combine multiple algorithms to enhance anomaly detection accuracy and robustness. They also discussed about ‘concept drift’ and through their series of experiments, what they discovered is that there is no one-size-fit-all solution for log anomaly detection. That is, not all anomaly detection techniques can be applied to all types of logs with great result.

Additionally, the paper addresses challenges related to feature engineering, interpretability of models, and the trade-off between false positives and false negatives in anomaly detection systems.

Overall, the research paper provides valuable insights into the state-of-the-art techniques and challenges in intelligent log analysis for anomaly detection, paving the way for more effective and efficient cybersecurity measures in modern IT infrastructures.

But, in our project we didn’t go in-depth to work on CNN and RNN and had worked on the basic AI and ML algorithms to train our machine on our dataset and test it.

About technologies used:

Python, Html and CSS, Flask, MERN, etc.

1)HTML and CSS: We used HTML and CSS for the front-end part which also includes a bit of Java script in it. Here we take input of the important features data which we obtained after the pre-processing and process it to know if the log has anomaly or not.

2)MERN: We used MERN for the login page creation as to ensure the security to some extent.

3)PYTHON: We used python for backend where it includes pre-processing of our dataset using min-max scalar and label encoder. After that we used Random Forest regressor and by some manual data and statistics to obtain the top features of the dataset which play a major role in finding the attack. After that, model training and testing is done by Random Forest classifier. We also used many libraries in this process such as numpy, matplotlib, pandas, etc.

4)FLASK: We used python flask to deploy our code and to obtain the best results.

Networking commands and other information:

To understand the anomalies in a network we have gone through some of the networking terminology and commands. Networking commands play a crucial role in managing and troubleshooting network connections across different operating systems. Here are some common networking commands along with their utilization and working IP addresses for Windows, macOS, and Linux:

1. ipconfig/ifconfig:

• Utilization: Displays the IP configuration information for Windows and Linux/macOS respectively. It shows details such as IP address, subnet mask, and default gateway.

• Working IP Address:

• Windows: 192.168.1.100

• Linux/macOS: 192.168.1.101

2. ping:

• Utilization: Checks the connectivity between two devices by sending ICMP echo request packets and receiving ICMP echo reply packets.

• Working IP Address:

• Destination IP: 192.168.1.1 (router or gateway)

3. traceroute/tracert:

• Utilization: Shows the route taken by packets from the source to the destination, displaying the IP addresses of intermediate routers.

• Working IP Address:

• Destination IP: 8.8.8.8 (Google DNS server)

4. netstat:

• Utilization: Displays network statistics, including active network connections, listening ports, and routing tables.

• Working IP Address:

• Windows/Linux/macOS: N/A (depends on the active network connections)

5. nslookup/dig:

• Utilization: Performs DNS (Domain Name System) lookup to resolve domain names to IP addresses and vice versa.

• Working IP Address:

• Domain: www.example.com

6. route:

• Utilization: Displays and manipulates the IP routing table, showing the routing information used by the system to determine the next hop for packet forwarding.

• Working IP Address:

• Destination IP: 192.168.2.0 (a remote subnet)

7. ifconfig/ip:

• Utilization: Configures network interfaces, including assigning IP addresses, enabling/disabling interfaces, and configuring network parameters.

• Working IP Address:

• Windows: N/A

• Linux/macOS: 192.168.1.102

8. arp:

• Utilization: Displays and modifies the ARP (Address Resolution Protocol) cache, showing the mapping between IP addresses and MAC addresses.

• Working IP Address:

• Destination IP: 192.168.1.1 (router or gateway)

9. iptables/ufw:

• Utilization: Manages firewall rules, including allowing or blocking traffic based on IP addresses, ports, and protocols.

• Working IP Address:

• Windows/Linux/macOS: N/A

These commands are some of the essential for network administrators and users to diagnose network issues, configure network settings, and ensure smooth communication between devices in a network environment. The working IP addresses provided are examples and can vary based on the specific network configuration.

We also used Wireshark, Ettercap, John-the-ripper, etc to perform attacks such as Password cracking, Network sniffing, etc.

Attacks:

Attacks which are addressed in the dataset we worked on are as follows: DDOS-HOIC, DOS-Golden eye, DOS attack-HULK, DOS attack- slowHTTPtest, DOS attack- Slowloris, FTP- Bruteforce, Infilteration, bot, SSH-Bruteforce.

1. DDoS - HOIC (High Orbit Ion Cannon):

• Description: HOIC is a DDoS tool designed to flood target websites or services with HTTP/HTTPS traffic, overwhelming their servers and causing service disruption.

• Impact: Targeted websites/services may become inaccessible to legitimate users, resulting in downtime and potential financial losses for the affected organization.

2. DoS – Golden Eye:

• Description: Golden Eye is a DoS tool that targets web servers by flooding them with HTTP/HTTPS requests, consuming server resources and causing it to become unresponsive.

• Impact: Similar to HOIC, Golden Eye aims to render targeted websites or services inaccessible, disrupting their normal operation.

3. DoS Attack - HULK:

• Description: HULK (HTTP Unbearable Load King) is a DoS attack tool that generates a large number of HTTP GET or POST requests with invalid headers, exploiting vulnerabilities in web server resource allocation algorithms.

• Impact: The excessive load generated by HULK requests can overload web servers, leading to denial of service for legitimate users attempting to access the targeted website or service.

4. DoS Attack - SlowHTTPTest:

• Description: SlowHTTPTest is a DoS tool that targets web servers by sending HTTP GET or POST requests very slowly, keeping server connections open for an extended period and consuming server resources.

• Impact: SlowHTTPTest can exhaust server resources such as available connections, memory, and processing capacity, leading to a denial of service for legitimate users.

5. DoS Attack - Slowloris:

• Description: Slowloris is a DoS attack tool that targets web servers by establishing multiple incomplete connections and keeping them open for as long as possible, exhausting server resources and preventing new connections from being accepted.

• Impact: Slowloris can effectively prevent legitimate users from accessing the targeted website or service by tying up available server resources and preventing new connections from being established.

6. FTP - Brute Force Attack:

• Description: A brute force attack against an FTP (File Transfer Protocol) server involves systematically attempting to log in to the server by trying different username/password combinations until the correct one is found.

• Impact: Successful brute force attacks can compromise the security of the FTP server, allowing unauthorized access to sensitive files or data stored on the server.

7. Infiltration:

• Description: Infiltration refers to unauthorized access to a network or system, often through exploiting vulnerabilities or using social engineering techniques.

• Impact: Infiltration can lead to data breaches, theft of sensitive information, disruption of services, and compromise of system integrity, posing significant risks to the affected organization.

8. SSH - Brute Force Attack:

• Description: A brute force attack against an SSH (Secure Shell) server involves attempting to log in to the server by systematically trying different username/password combinations.

• Impact: Successful SSH brute force attacks can grant unauthorized access to the server, allowing attackers to execute commands, modify configurations, and potentially compromise the security of the entire system, etc.

9. Bot Attack:

A bot attack, also known as a botnet attack, is a type of cyberattack in which a network of compromised computers, known as bots or zombies, is used to carry out malicious activities. These bots are typically infected with malware that allows them to be remotely controlled by an attacker, forming a botnet under their command. Bot attacks can have various objectives, including:

**DDoS Attacks:** Botnets can be used to launch Distributed Denial of Service (DDoS) attacks against target websites or services by flooding them with an overwhelming amount of traffic. This flood of traffic can cause the targeted systems to become inaccessible to legitimate users.

**Spamming:** Botnets can be utilized to send out large volumes of spam emails or messages, spreading phishing scams, malware, or other malicious content to unsuspecting recipients.

**Brute Force Attacks:** Bots can be used to conduct brute force attacks against online accounts, attempting to guess usernames and passwords in order to gain unauthorized access to systems or services.

**Click Fraud:** Botnets may engage in click fraud, where automated bots simulate clicks on online advertisements or pay-per-click links to generate revenue for the attacker.

**Data Theft:** Bots can be used to steal sensitive information, such as login credentials, financial data, or personal information, from compromised systems or networks.

Bot attacks pose significant threats to individuals, organizations, and even entire sectors of the internet. They can be difficult to detect and mitigate due to their distributed nature and the sheer volume of compromised devices involved.

Conclusion:

In this project, we evaluated various techniques used for log anomaly detection. We built upon previous anomaly detection techniques. We have worked on CICIDS 2018 data set. We have used ML algorithms such as Random forest classifier to train and test our machine. Through our experiments, we showed that our model can achieve high accuracy for anomaly detection after training on only a small amount of normal log data. Future work could involve the concept drift problem and applying our proposed system to additional types of log data or the incorporation of a module to help users more easily diagnose identified anomalies.

References:

[1] A. Chuvakin, K. Schmidt, and C. Phillips, Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management, 1st ed. Waltham, MA, USA: Elsevier, 2013.

[2] W. Xu, et al., “Detecting large-scale system problems by mining console logs,” Proc. of the ACM SIGOPS 22nd Symposium on Operating Systems Principles, New York, NY, Oct. 2009.

[3] S. He, et al., “Experience Report: System Log Analysis for Anomaly Detection,” 27th International Symposium of Software Reliability Engineering, Ottawa, ON, Canada, Oct. 2016.

[4] S. Nomm and H. Bahsi, “Unsupervised Anomaly Based Botnet Detection in IoT Networks,” IEEE International Conference on Machine Learning and Applications, Orlando, FL, USA, Dec. 2018.

[5] Lima, M., Zarpelao, B., Sampaio, L., Rodrigues, J., Abrao, T., & Proenca, M. (2010). “Anomaly Detection Using Baseline and K-Means Clustering. International Conference on Software,” Telecommunications and Computer Networks (SoftCOM). Split, Dubrovnik, Croatia.

[6] Yin, C., Zhang, S., Wang, J., & Kim, J. (2015). “An Improved K-Means Using in Anomaly Detection,” First International Conference on Computational Intelligence Theory, Systems and Applications (CCITSA). Yilan, Taiwan.

[7] Xu, W., Huang, L., et al., “Online System Problem Detection by Mining Patterns of Console Logs,” Ninth IEEE International Conference on Data Mining, Miami, FL. Dec. 2009.

[8] Jiawei Han, Micheline Kamber, Jian Pei. Data Mining: Concepts and Techniques. 3rd ed, Morgan Kaufmann, 2012.

[9] Haque, A., DeLucia, A., Baseman, E., “Markov Chain Modeling for Anomaly Detection in High Performance Computing System Logs,” Proceedings of the Fourth International Workshop on HPC User Support Tools, Denver, CO. Nov. 2017.

[10] “Intelligent Log Analysis for Anomaly Detection”, Steven Yen, San Jose State University, Spring 2019.