



UNIVERSITY OF NAIROBI

FACULTY OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

CCI 599: PROJECT

Security Information and Event Management Using Deep Learning

PROJECT DOCUMENTATION

P52/37435/2020

ODONGO BENSON ODHIAMBO

Submitted to:

Dr. Elisha Abade

On

July 2022

Declaration

I hereby declare that everything expressed in this research documentation is based on my knowledge and research carried out with exception to printed or electronic content and has not been submitted to any institution of learning for any academic awards.

Name: **ODONGO BENSON ODHIAMBO**

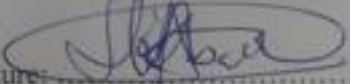
Registration No: **P52/37435/2020**

Signature: 

Date: 16/9/2022

This project has been submitted in partial fulfillment of the requirements for the award of a Master's Degree in Computational Intelligence from the University of Nairobi and has been done with the full support of my supervisor:

Name: **Dr. ELISHA ODIRA ABADE**

Signature: 

Date: 16/09/2022

Abstract

Organizations currently deploy various security solutions to protect their information resources. These tools like firewalls, network gateways, and other intrusion prevention tools have become obsolete as hackers can now break into firewalls, send emails with malicious and infected attachments or even bribe employees to gain access to an organization's firewalls. A new approach to cybersecurity is by using user and entity behavior analytics (UEBA). The focus of this paper is to demonstrate how UEBA and deep learning algorithms can be used to detect suspicious and anomalous behaviors within a system. Based on historically profiled user action sequences in a network, Long Short-Term Memory (LSTM) neural network will be used to predict the next state of user action and flag an action as suspicious when the action sequence deviates from the predicted sequence.

Keywords

Cybersecurity; User and Entity Behavior Analytics (UEBA); Deep learning, Long Short-Term Memory (LSTM)

Table of Contents

1.	INTRODUCTION	5
1.1	Background of the study	5
1.2	Statement of the Problem:	6
1.3	Research Objectives	7
1.4	SIGNIFICANCE OF THE RESEARCH	7
2	LITERATURE REVIEW	8
2.1	Introduction	8
2.2	Security orchestration, Automation and Response	11
2.3	User and Entity Behavior Analytics	11
2.4	Deep Learning Algorithms.....	12
2.4.1	Why Deep Learning for Insider Threat Detection	13
2.4.2	Long Short-Term Memory (LSTM)	14
2.5	Related Work	15
2.6	Research Gap	16
3	METHODOLOGY	17
3.1	Introduction	17
1.	Data Sources and Preparation	17
2.	Feature Extraction.....	17
3.	Behavior Profiling.....	19
4.	Anomaly detection Using Deep Learning.....	19
	LSTM.....	19
3.2	System Development Methodology	23
3.3	Project Timelines.....	25
4	RESULTS, FINDINGS AND DISCUSSION.....	26
4.1	CERT DATASET.....	26
4.2	EVALUATION AND RESULTS	27
4.2.1	Results.....	28
4.3	DISCUSSION.....	31
5	CONCLUSION.....	32
5.1	LIMITATIONS	32
5.2	FUTURE RESEARCH.....	32
6	References.....	34

Figure 1 . Unfolded LSTM network	15
Figure 2 .LSTM Architecture with a single memory block	20
Figure 3.1 General Architecture of UEBA	22
Figure 4 UEBA data sources.....	22
Figure 5. prototyping model phases.....	24
Figure 6 : LSTM Details.....	28
Figure 7:Training with only user Activity data	29
Figure 8 : Training with Mixed Data : User Activities/features and action sequence	29
Figure 9 :WDD loss	30

1. INTRODUCTION

1.1 Background of the study

Individuals, enterprises, governments, and society as a whole have become more technologically reliant than ever before. The majority of digital devices, such as computers, smartphones, and tablets, are now capable of connecting to the internet. As a result, sensitive information, such as the specifics of bank accounts, national identification and voter registration, credit card details, as well as other sensitive details, are now stored on cloud storage servers. The storing of this kind of information in the cloud makes it susceptible to data leaks and breaches, both of which could have negative effects on the individuals involved.

Cybercriminals have recently become more sophisticated, changing what they target and using different techniques to attack different security systems. Until recently, firewalls, web gateways, and intrusion detection tools were enough to provide security. These tools have become obsolete since currently skilled hackers and cybercriminals are now capable of bypassing this kind of perimeter defense. These attackers can now infiltrate into secure firewalls, send emails with malicious and infected attachments or even bribe employees to gain access to the organization's firewalls.

An Accenture Security study (2019) estimated that the average cost of cybercrime had climbed by \$1.4 million over one year, with the average number of data breaches increasing by 11 percent to 145.

Malicious people are no longer only outsiders but also insiders who could be disgruntled employees, contractors, and consultants who have access to your premises, and therefore, these preventive measures are no longer sufficient. Firewalls and traditional intrusion detection tools are not going to be purely foolproof, and hackers and attackers will

get into an organization's system at one point or another. Detection is therefore very important: when these attackers manage to get into the organization's system, the organization should be able to detect quickly to minimize the damage.

1.2 Statement of the Problem:

Malicious people are no longer outsiders but insiders who could be disgruntled employees, contractors, or consultants who have access to the company's premises. The ability to rapidly identify compromised user accounts and insiders with harmful intent is a key challenge in enterprise security. Detecting compromises have become so complicated nowadays that simple rules alone are not enough hence the need to use machine learning to analyze these data.

Workspaces have huge data, almost every system is nowadays logging so much data that nobody can sit down and analyze them. The current platforms that are used to monitor and identify potential security threats cannot automatically analyze and identify threats from insiders.

Therefore, this study suggests analyzing network data produced by users and entities using machine learning methods in order to identify fraudulent or suspicious behavior within an organizational system

1.3 Research Objectives

These study goals were established::

1. To investigate how machine learning can be applied to logs generated by users and network entities to enhance cybersecurity
2. To implement a prototype in demonstrating the application of deep learning in cybersecurity
3. To assess the effectiveness of the proposed prototype at spotting dangers and unusual activity in a network

1.4 SIGNIFICANCE OF THE RESEARCH

Organizations can safeguard their data from the inside out with the help of User and Entity Behavior Analytics by creating a database of acceptable user behavior patterns and then comparing the actions of remote users to those patterns to determine whether or not they are secure and normal.

Furthermore, a tool based on data science and machine learning can help organizations detect malicious activity faster and act according to prevent damage.

2 LITERATURE REVIEW

2.1 Introduction

With the continuous hacking practices, many researchers have generated varied definitions of hackers with no unified meaning or description. For instance, Taylor et al. (2015) argue that hackers are people who access computers illegally and manipulate the system's weaknesses. On the contrary, Pavlick (2017) establishes that hackers can access computers legally to make sure that the firm's software is safeguarded. Furthermore, various researchers like Taylor et al. (2015) accounts that hacking could be legal when the hacker has the authority to log into the system. These legal hackers are identified as white hats. They include computer specialists who examine the methodologies of a firm's computer system to determine any flaw (Pavlick, 2017; Taylor et al., 2015). Alternatively, according to Taylor et al. (2015), illegitimate computer hackers are known as the black hats. They maliciously generate malware and access a company's system and networks without authorization. Therefore, this paper will use the definition of the latter type of hacker for clarity.

Some experts have known since the early 1990s that most security breaches and hacks are the result of inside jobs. Researchers Loch et al (1992), they reported that the security of a firm's data is susceptible to insider threats and hackers and negligent employees. Moreover, Warkentin & Wilson (2009) and Mitnick & Simon (2001) explain that this susceptibility results from the employees being weak connection in data security. For instance, during a study carried by Richardson and Director (2008) from the Computer Security Institute, they observed that their participants lost \$288,618 on average through an insider security breach. Besides, 44% of these subjects confirmed being exploited after hacking practices. Richardson and Director, (2008) concluded that insider hacking is amongst the most common and detrimental computer security-related occurrences.

For example, a study conducted by Rockefeller (2014) on Target's security breach in 2013 elaborated how employees' uncooperative behavior can severely affect a company. The report showed how a 3rd party dealer, Fazio Mechanical Services, hacked this company's payment software. This malicious activity led to the loss of some staff members' network credential data via a phishing attack delivered as a vicious email to Fazio. These hackers applied the stolen data from Fazio to access Target's payment software remotely, stole personal data and payments of around 110 million clients, and deleted this activity to a server in Eastern Europe from Target's system.

According to Wallace (2014), this hacking at Target impacted at least one-third of the American population, revealing the financial statistics of around 34% of Americans. In the study by Abrams (2014), he reported how this information breach directly contributed to a decline in Target. It lost around \$148 million to its shareholders and a 1% decrease in its stock price. Moreover, this company started to give out free services like credit monitoring to its clients, undermining its productivity in the economic market. At the same time, most people avoided using Target in fear of their data being leaked, earning the company a bad reputation. Apart from these predicaments, the company faced several lawsuits from clients damaging its goodwill further.

Colwill (2009) also pinpointed how governmental agencies and organizations can be easily breached through insiders' activities. Thomson (2007) accounted how the Government Revenue and Customers Department in the UK was hacked leading to the loss of private data of twenty-five million residents. Furthermore, in the research conducted by McCue (2008), it was observed that although security measures and monitoring focuses on the external threat, insiders execute about 70% of network hacking, a higher figure compared to external threats.

Even more so, several writers and academics have compiled overviews and performed surveys on insider dangers in various socioeconomic sectors. By way of illustration, Walker-Roberts et al. (2018) conducted a systematic review on identifying instances of insider hacking. This review was based on insider hacking and threats in the healthcare sector. Secondly, Alneyadi et al. (2016) and Ullah et al. (2018) looked at various insider threats, such as data leakage and data exfiltration. However, the latter authors presented comprehensive information and understanding on data exfiltration resulting from the vicious actions of an internal hacker. Additionally, some reviewers and scholars like Crossler et al. (2013), Farhmand and Spafford (2013), and Ho et al. (2018) provided their readers with a deeper understanding of insider threats and the future guidelines for denoting the behavior of local hackers. On the same note, Ho et al.(2018) also presented future directions based on behavioral and technical sociotechnical works. Finally, in the study conducted by Nazir et al. (2017), they offered an inclusive survey on simulation, modeling, and other methods to be applied when assessing the system of SCADA's (supervisory control and data acquisition) susceptibility to hacking.

In summation, insider hacking is dangerous for any firm due to its association with employees, the weakest link in defending a company's network system. Moreover, no technique or data security strategy can be effective without the candid cooperation of the employees (Earnst & Young, 2002). The lack of cooperation from the staff members will cause a challenge for both non-profit and profit organizations for their workers to conform to the data security measure and its implementation. Therefore, there is a need to develop a tool that can automatically detect anomalies by analyzing user and entity behaviors of employees without depending on their compliance.

2.2 Security orchestration, Automation and Response

Information regarding security threats is gathered and automated responses are implemented through a process known as Security Orchestration, Automation, and Response (SOAR) (Shea, 2021).SOAR components:

Security Orchestration – Using a collection of security solutions such as firewalls, IDS/IPS, SIEM, and end-user behavior analytics to monitor activity and send notifications

Security Automation - Automatically analyze data collected from security orchestration and make a recommendation or automate future responses. Methods such as scanning for security holes in the system, analyzing logs, verifying tickets, and conducting audits are all part of this evaluation.

Security Response - offers a single view dashboard to plan, manage, monitor and report incident response

2.3 User and Entity Behavior Analytics

User and entity behavior analytics (UEBA), also known as user behavior analytics (UBA), can be defined as the process of gaining insight into the network events that users contribute daily, as stated by (Rapid7, 2021). Once the data has been collected and analyzed, it can be utilized to spot suspicious activity like lateral movement or the usage of compromised credentials.

It is necessary to not just undertake user but also entity behavior analytics due to the growing threat from external forces other than just individual users. The entities might be routers, servers, applications, or any other network device that could be compromised..

The scenario below as demonstrated by (Shashanka, 2016) gives an example of how UEBA can be applied. An enterprise's critical server, for instance, must be watched for suspicious activity, such as logins from compromised accounts. Each user's server activity will be tracked (Shashanka, 2016).. To detect an anomaly, a baseline is defined against which user patterns will be compared. Two scenarios can be considered:

- Historical baseline: A user's behavior is evaluated against their previous behaviors over the past period
- Peer baseline: The actions of a user are judged in comparison to the patterns of actions taken by their peers.

Identifying a time-granularity for the purpose of analysis and a set of features that may be used to characterize the access patterns of each user over a certain period of time are both required components of a user behavior. The features may include the following::

- A record of the day's first and last access times
- The duration between access times
- The cumulative duration of all daily flows
- Number of flows over the entire day
- The combined amount of bytes downloaded and uploaded

The data collected from the above features over time servers as input for algorithms used for anomaly detection.

2.4 Deep Learning Algorithms

Deep learning algorithms provide us the ability to sift through massive datasets and recognize patterns of behavior, also known as signals in the noise, in a way that would be nearly hard for humans to do on their own. The Long-Short-Term Memory (LSTM) recurrent

neural network and the convolutional LSTM will be the deep learning algorithms that will receive a significant amount of attention throughout this research. (Graves, 2012)

2.4.1 Why Deep Learning for Insider Threat Detection

The potential benefits of deep learning for insider threat detection can be summed up by the following, which are among the many appealing features of deep learning models.

(Shuhan Yuan, 2020)

- **Representation Learning.** Deep learning models' capacity to automatically find the attributes required for detection is their most important advantage. Cyberspace user behavior is complex and non-linear. Information about user behavior is difficult and inefficient to collect when features are manually designed. Learning models with shallow structures, such as HMM and SVM, use only one layer to transform the raw feature data, producing an abstract that may be used for detection.. While these simplistic models can be helpful in many specific scenarios, they have difficulty modeling more complicated user behavior data due to their inherent simplicity. In contrast, deep learning models can make use of deep non-linear modules by employing a generic learning process to learn the representation. To accurately capture complicated user behavior and identify user intentions, particularly those that are harmful, deep learning models make perfect sense.
- **Sequence Modeling.** When it comes to modeling sequential data like video, text, and speech, deep learning models like the Recurrent Neural Network (RNN) and the recently proposed Transformer have shown promising results (Graves, 2013). According to a recent study (Ashish, Noam, Parmar, Jakob Uszkoreit, & Jones, 2017), the opposite is true. Since it is natural to define the user actions collected in audit data as sequential data, using RNN or Transformer to capture the relevant information of

complicated user behavior can significantly boost the efficiency of insider threat detection.

- **Heterogeneous Data Composition.** Picture captioning is only one example of the types of tasks well-suited to deep learning models (Cheng, Fei Huang, Jin, Zhang, & Zhang, 2017). In addition to encoding user activity data as sequences, other information, such as user profiles and user hierarchy in an organization, is vital for insider threat detection. It is projected that using many types of useful data will improve performance compared to using just one type of data for insider threat detection. Deep learning models are more successful than conventional machine learning techniques at combining diverse data for detection.

2.4.2 Long Short-Term Memory (LSTM)

Recurrently connected memory blocks are the building blocks of a recurrent neural network, and LSTMs are no exception. These pieces are analogous to the customizable memory chips seen in digital computers. Input, output, and forget gates make up the three multiplicative units of each block, and together they supply the memory cells with continuous analogs of write, read, and reset operations (Graves, 2012).

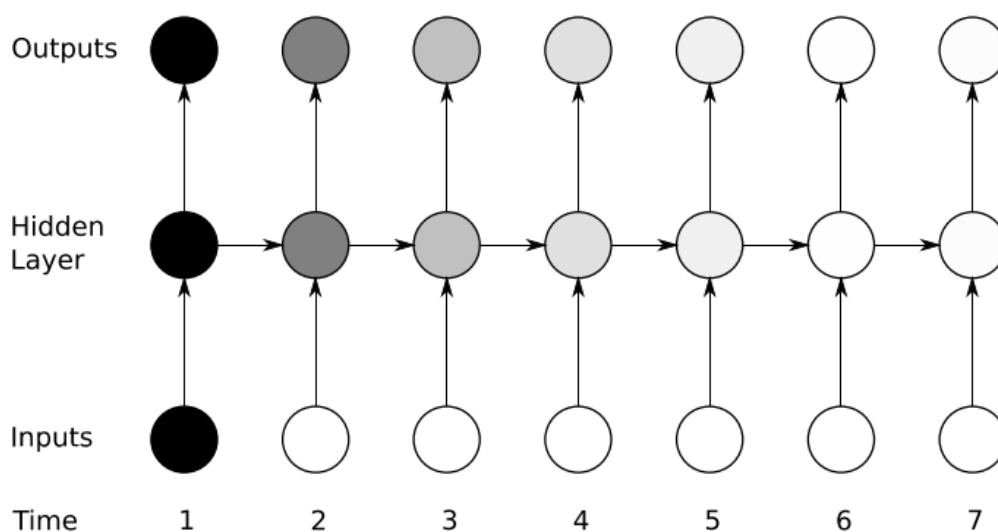


Figure 1 . Unfolded LSTM network

Over the past decade, LSTM has proven effective in a wide variety of settings, including the acquisition of new lexicons in context-free languages (Schmidhuber., 1997), the retention of high-precision real numbers over lengthy noisy sequences, and the performance of other tasks requiring precise timing and counting. (Graves, 2012).

2.5 Related Work

Anomaly detection methods and machine learning approaches have a long history of being used to solve computer security issues. A lot of academics have tried to apply anomaly detection to security issues since then, but not much of it has made it into the market.

The capabilities of machine learning and deep learning have been put to use in many different arenas. Machine learning and deep learning are increasingly being used to monitor company employees for signs of misconduct (ZHIHONG, 2020).

Their study, (Shashanka, 2016) proposed an intelligent platform for cybersecurity threat detection that uses UEBA to track and monitor the behavior of users. Their paper, (Shashanka, 2016) proposed using a Singular Value Decomposition machine learning algorithm to automatically detect potentially malicious activity.

Some further studies were conducted (Raguvir. S, 2020). It was proposed in this study that UEBA be used to help identify trends and abnormalities in user activities and behaviors. The report focused on the usage of user and entity behavior analytics to achieve this goal (Raguvir. S, 2020). The user's identity, IP address, time of use, and date of use were among the many factors they analyzed. In total, three months' worth of data was analyzed. Before creating visual analytics, researchers used a visualization dashboard to discover patterns within the data, then applied mining, scripting, and processing to the raw data (Raguvir. S,

2020). Their research revealed that some irregularities emerged when looking at various patterns.

Multiple ML technologies, including Bayesian-based approaches (Weizhi Meng, 2018), a decision tree (Duc C. Le, 2019), and a self-organizing map (Duc C. Le, 2018), have been evaluated to identify insider threats. Conditions for atypical user behaviors can be identified with the help of real-time learning techniques.

2.6 Research Gap

Detecting compromises have become complicated nowadays and simple rules are not enough. Workspaces nowadays have huge amounts of data, almost every system is logging so much data that it becomes extremely difficult for people to sit down and analyze them. It, therefore, requires the use of artificial intelligence, especially big data analysis to analyze these kinds of data. In this research, deep learning algorithms will be used and especially Users' past activities will be used to train LSTM on how to predict their future actions and spot anomalies. if the sequence pattern deviates.

3 METHODOLOGY

3.1 Introduction

The focus of the project is to perform user-data analysis utilizing deep learning models using users' data collected through their actions on personal computers, server logs, etc. to detect abnormal behavior from the insiders and consultants who access the organization's servers.

The methodology to be used is described below.

1. **Data Sources and Preparation** – this will involve obtaining data from various data sources, this will including data such as server and file access logs, flows, network traffic packets, login and logout files, alerts, and threat feed. Data will be obtained from:

- Active Directory and other authentication systems,
- Real-time Virtual Private Networks and Proxies
- Databases used for configuration management
- information from the human resources department, including information on new and former workers as well as any other information that can shed light on the user's firewall, intrusion detection, and prevention systems
- anti-virus and anti-malware software systems
- Analyses of traffic on a network and
- Threat intelligence feeds

2. **Feature Extraction** – In this step, we will extract users' access habits and represent it so that deep learning algorithms can spot anomalies when they arise. The system is designed with a view that workers in an organization are grouped according to their

roles such as accountants, administrators, engineers, and managers. This means that workers in the same group will have similar duties, making it possible to use their everyday activities, behaviors, and other data to extract role traits and use them to justify detection. For example, workers in accounting are inclined to generate payrolls, raise or read invoices, approve procurement documents and generate accounting reports, throughout office hours, and write and receive emails and phone calls; if one worker in accounting rarely accesses the payrolls or review procurement forms but all of a certain start accessing some files that they haven't accessed before and starts numerous downloads and storage to external drives when there is no major event in the company then the employee's behavior can be flagged as suspicious.

The features extracted will be limited to:

- Users' daily activities for each period (this can be hourly, daily, or weekly) – these activities will be obtained from log files, external drives usage i.e., connecting and removing events and network flows and traffic packets
- The sequence of behavior for a given period. This will involve getting the action sequence of a user, that is, what a user does each time they access the server or computer. For instance, a user may log in, access their email, browse the web, download files then log out. This action sequence is not static but aims to get users' habits as users may do different activities each day.
- Users' social features. This area will involve getting the topics users are interested in based on their social media posts. This feature may help identify any anomalous behavior based on their social media posts. Disgruntled employees may vent their anger on social media to retaliate against their colleagues and may form the basis of insider intrusion.

- Peer features or employee role features. These are features based on activities performed by employees within the same job role. For example, we could expect employees from the same department to perform closely related activities.
3. **Behavior Profiling** – At this point, the extracted features are organized into baselines for each entity, and the behaviors of each individual or group are forecasted using a machine learning model (Shashanka, 2016). Defining the baseline for an anomaly will be based on:
- **Historical Baseline** – This will involve evaluating a user's behavior against their previous behaviors over time in the past to detect a deviation in their routine activities
 - **Peer Baseline** – Will involve evaluating a user's behavior against the ones of the users who performs the same roles as them to detect instances of suspicious activity
4. **Anomaly detection Using Deep Learning** – The assumption is that changes in a user's behavior based on the profile generated either through prediction based on historical baseline or peer baseline is an indication of anomalous behaviors that may cause data breaches. Deep learning algorithms will be used to detect whether data collected from users or entities such as server access logs, logins, logout files, file alerts, threat feeds or packets show suspicious behavior of insiders. In order to identify unusual patterns of behavior exhibited by regular users of a system, convolutional long-short-term memory neural networks will be put into practice.

LSTM – For more precise modeling of temporal sequences and their long-range dependencies than is possible with traditional recurrent neural networks, a specialized

neural network design known as Long Short-Term Memory (LSTM) was developed. The application of LSTM has been successful in numerous sequence prediction and sequence labeling tasks (Hasim Sak, 2014)

As a result of its usefulness in most regression situations as a tool for figuring out what state will come next by looking at the previous ones, LSTM was therefore the right choice of algorithm for the problem we are solving. LSTMs will be trained to learn the normal user day-to-day activity sequence and used to predict the next state sequence using the historical data. If a deviation is detected between the actual activity and the predicted activity sequence, then it would be an indication of anomaly activity.

As illustrated in Figure21 below, an example of the input sequence for the LSTM could be users sequence actions represented as {login, browse the web, check email, download, connect a drive, disconnect drive,, logout} with hidden state layers and cell output that feeds into other layers.

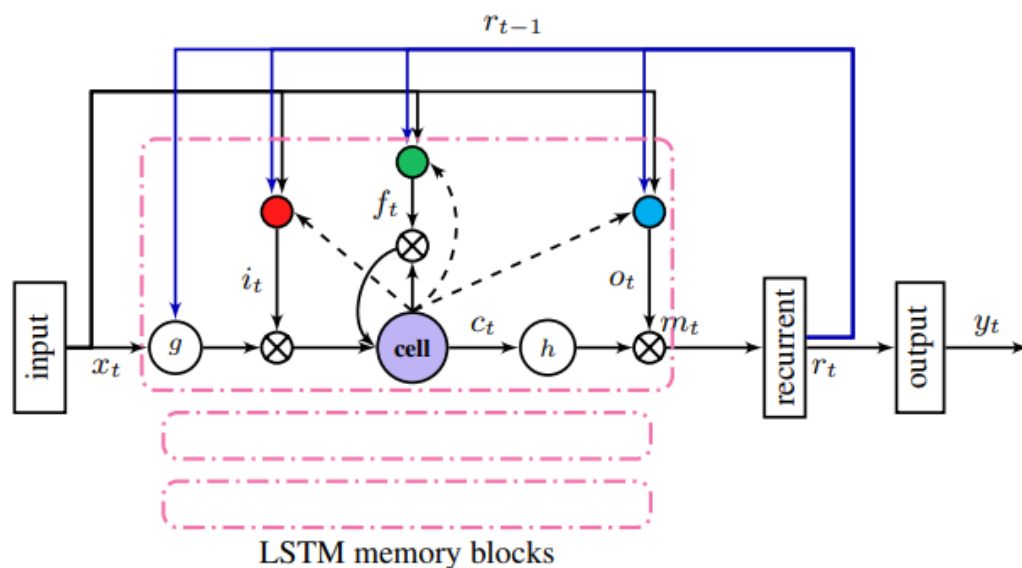


Figure 2 .LSTM Architecture with a single memory block

By iteratively using the below equations from $t = 1$ to $t = T$, a long short-term memory (LSTM) network can compute a mapping from an input sequence $x = (x_1, \dots, x_T)$ to an output sequence $y = (y_1, \dots, y_T)$.

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f) \quad (2)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1} + b_c) \quad (3)$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o) \quad (4)$$

$$m_t = o_t \odot h(c_t) \quad (5)$$

$$y_t = \phi(W_{ym}m_t + b_y) \quad (6)$$

where W terms are weight matrices (for example, W_{ix} is the matrix of weights from the input gate to the input), W_{ic} , W_{fc} , and W_{oc} are diagonal weight matrices for peephole connections, and b terms represent bias vectors (b_i is the input gate bias vector), In this equation, σ is the logistic sigmoid function; i , f , o , and c are the input gate, forget gate, output gate, and cell activation vectors, respectively; m is the element-wise product of the vectors; g and h are the cell input and cell output activation functions; and ϕ is the activation function for the cell output. Hasim Sak (2014) Architectural Design

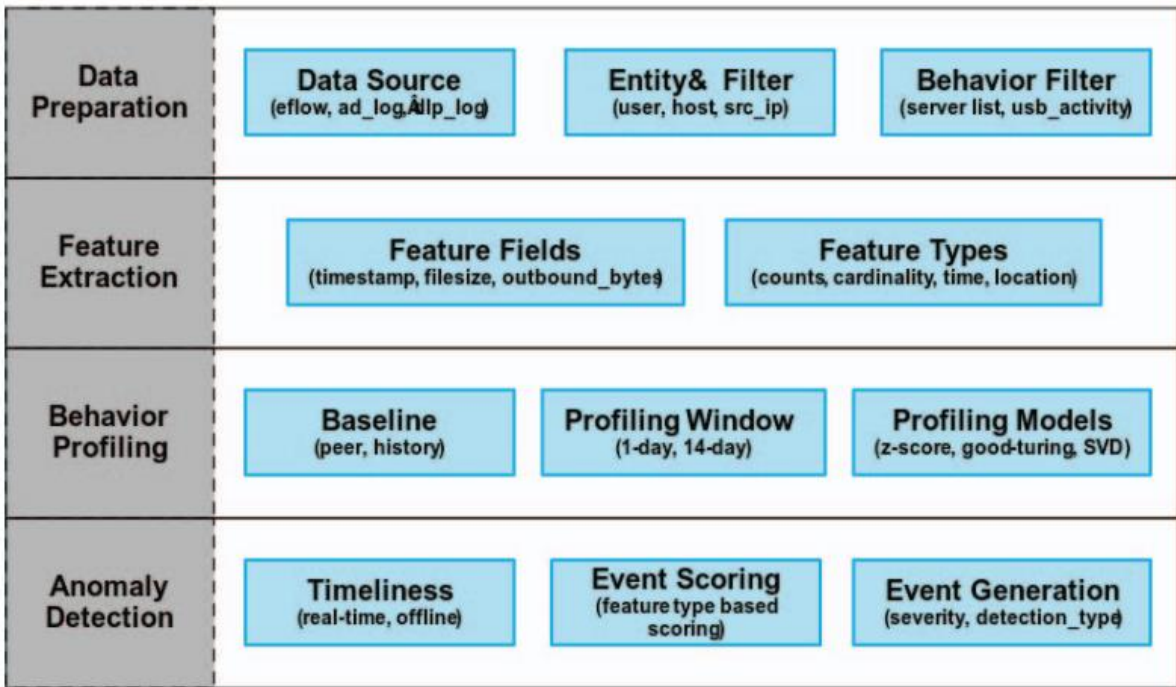


Figure 3.1 General Architecture of UEBA

The above workflow can be described in phases below.

1) **Data Preparation** – This stage involves obtaining data from various data sources.

(Accenture Security, 2019)

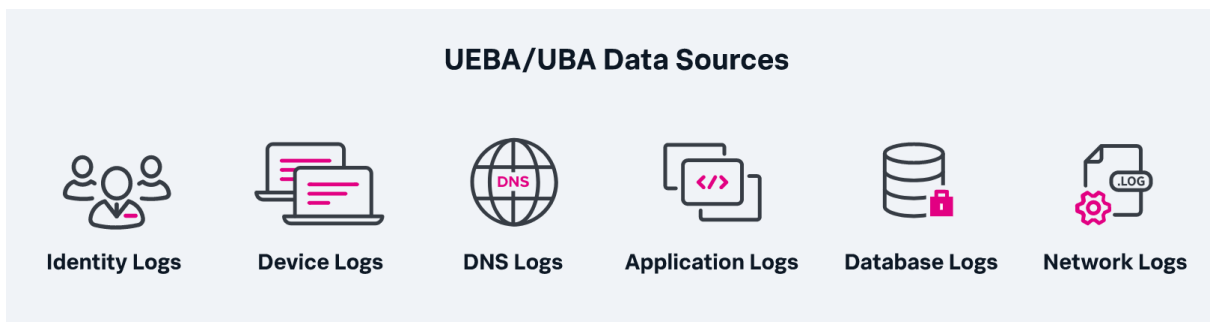


Figure 4 UEBA data sources

(Data Insider, 2021)

- 2) **Feature Extraction** - In order for deep learning algorithms to detect anomalous behavior, it is necessary to extract relevant information from log files (such as login and logout records, network traffic packets, and file access logs) and transform it into a normalized representation. Appropriate attributes are crucial in describing the threat profiles of suspected individuals by modeling their normal behavior and catching aberrations that are suggestive of abnormal behaviors. Therefore, it is helpful to define the many activities that a user can perform (logging in, opening a file, sending an email, etc.) so that their behaviors can be modeled based on how insiders act and used as our features (T. Rashid, 2016)
- 3) **Behavior Profiling** At this point, the extracted features are organized by entity, and a machine learning model is used to create a behavior profile for each entity (Shashanka, 2016)
- 4) **Anomaly Detection** – At this stage, deep learning models are applied to detect deviations in actual user actions from their defined behavioral profiles and alert for suspicious or anomalous activity

3.2 System Development Methodology

As for the software itself, we'll be using a prototyping approach. The Prototyping Model is a method of software development in which a working prototype is created, evaluated, and revised as necessary. In addition, it lays the groundwork for the development of the actual system or program. (Martin, 2021).. The reason for choosing prototyping is because the model that will be built for anomaly detection will need to be tested and evaluated iteratively until a more optimal

model is accepted. That is why there will be a cycle between the implementation and testing stages.

The prototyping model phases

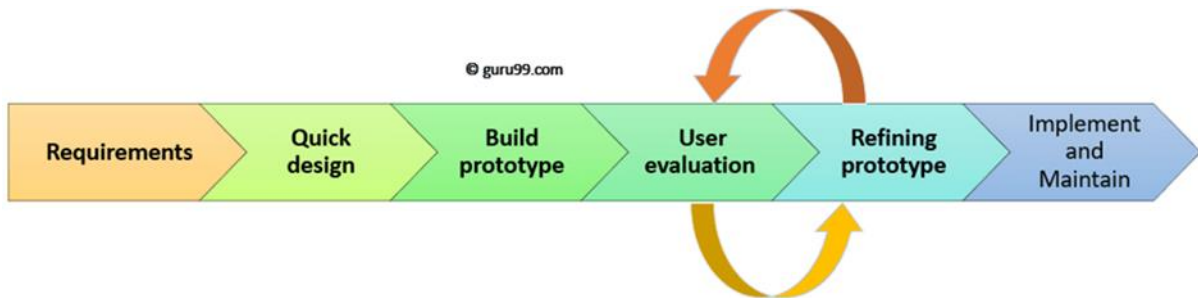


Figure 5. prototyping model phases

Prototyping model has the following six software development life cycles:

Step 1: Requirements Specification and System Analysis

Requirements gathering is the first step in a prototyping framework. During this stage, the system's needs are specified in great detail. Users are surveyed to determine what features they would want to see implemented in the system..

Step 2: System Design

In the second stage, brief prototype will be created. A basic blueprint of the system is drawn up at this point. But it's still missing certain key components to be a whole blueprint. User's get a high-level overview of the system. The rapid design aids in creating a working prototype..

Step 3: Prototype Implementation

Here, the knowledge gained from rapid prototyping is used to create a working version of the product. It's a little replica of the necessary system that works.

Step 4: Initial user evaluation

First, the suggested system is given to the client for their feedback. By doing so, we can better understand the benefits and drawbacks of the current approach. Customers' feedback is compiled and given to the programmers for consideration.

Step 5: Refining prototype

If the user has concerns about the present prototype, you should make adjustments based on their input.

Until all of the user-defined criteria are satisfied, we will continue to iterate through this step. After the prototype has been produced and the user is satisfied with it, a more permanent system is created based on the prototype's specifications.

Step 6: Implement Product and Maintain

After the final prototype has been produced, the system moves on to rigorous testing and eventual deployment in the production environment. Upkeep is performed regularly to reduce system outages and forestall catastrophic crashes.

[3.3 Project Timelines](#)

Appendix 1 shows the project schedule

4 RESULTS, FINDINGS AND DISCUSSION

Many suitable datasets are not available for insider threat recognition. Therefore, the data used for this experiment is a publicly available Computer Emergency Response and Threat (CERT) insider threat dataset. The synthetic data used in the Insider Threat Test Dataset includes both benign and malevolent actors. As of the year 2020 (Lindauer). There are six different types of logs in this data set, including HTTP, email, login, psychometry, device, and file. The dataset includes the work habits of a thousand workers over seventeen months (J. Glasser, 2013).

4.1 CERT DATASET

The Software Engineering Institute's Computer Emergency Response Team (CERT) at Carnegie Mellon University keeps track of more than a thousand actual insider threat case studies and has created a trove of synthetic insider threat datasets based on scenarios with traitor instances and masquerade actions. The CERT dataset consists of five log files that record all employees' computer-based actions in a simulated company, such as login.csv which records all employees' logon and logoff procedures, and so on. The files email.csv and http.csv log all outgoing and incoming emails and online traffic, respectively; file.csv logs all file operations (read, write, copy, delete); and the device logs all hardware operations. statistics.csv detailing the lifetime of a USB flash drive (connect or disconnect).

Table 1 :Activity types in CERT log files

Files	Operation Types
logon.csv	Weekday Logon (employee logs on a computer on a weekday at work hours)
	Afterhour Weekday Logon (employee logs on a computer on a weekday after work hours)
	Weekend Logon (employees logs on at weekends)
	Logoff (employee logs off a computer)
email.csv	Send Internal Email (employee sends an internal email)
	Send External Email (employee sends an external email)
	View Internal Email (employee views an internal email)
	View external Email (employee views an external email)
http.csv	WWW Visit (employee visits a website)
	WWW Download (employee downloads files from a website)
	WWW Upload (employee uploads files to a website)
device.csv	Weekday Device Connect (employee connects a device on a weekday at work hours)
	Afterhour Weekday Device Connect (employee connects a device on a weekday after hours)
	Weekend Device Connect (employee connects a device at weekends)
	Disconnect Device (employee disconnects a device)
file.csv	Open doc/jpg/txt/zip File (employee opens a doc/jpg/txt/zip file)
	Copy doc/jpg/txt/zip File (employee copies a doc/jpg/txt/zip file)
	Write doc/jpg/txt/zip File (employee writes a doc/jpg/txt/zip file)
	Delete doc/jpg/txt/zip File (employee deletes a doc/jpg/txt/zip file)

The various log file activity categories are detailed in Table 1. It also includes descriptive material for each exercise. The time, sender, recipients, subject, and body of an email are all recorded under the "Send Internal Email" action type. Each employee's "Big Five personality traits" (as measured by a psychometric score) are included in the CERT dataset alongside computer usage information in the file named psychometric.csv.

4.2 EVALUATION AND RESULTS

Variability and scalability are present in performance evaluations. In this study, we defined Weighted Deviation Degree (WDD) as a metric for evaluating discrepancies between observed and predicted characteristics by applying a linear weighting to the squared error.

$$WDD = \frac{1}{|V|} \sum_{y \in V} w(y - \hat{y})^2,$$

where V is the collection of all features that are present in the actual feature map, y is a single feature that is a part of V , y' is the same feature as y but is a part of the projected feature map,

and w is a value that has been specifically developed in accordance with the feature y .

(ZHIHONG, 2020)

4.2.1 Results

The model used for this experiment consisted of two 160 units and 100 units separate LSTM layers, 37 hidden layers, and an output layer with 'relu' activation. Figure 6 below shows the details of the LSTM model:

```
# ----- (input)-----
x_train=np.loadtxt(files_train,delimiter=',')
y_train=np.loadtxt(labels_train,delimiter=',')

# -----(model structure)-----
main_input=Input(shape=(148,),dtype='float32',name='MainInput')
Inputs=Reshape((4,37))(main_input)
layer=LSTM(100,return_sequences=True,activation='tanh')(Inputs)
layer=LSTM(160,return_sequences=False,activation='tanh')(layer)
layer=Dense(37,activation='relu')(layer)
output=layer
model=Model(inputs=main_input,outputs=output)
model.compile(optimizer='adam',loss='mse',metrics=['binary_accuracy'])

# -----(save the model)-----
```

Figure 6 : LSTM Details

Since each participant had unique traits, they were trained separately in this experiment. The LSTMs were trained using action sequences once every five days due to data and time constraints; the LSTMs were fed information from the previous four days to make predictions about the fifth day's data, and the actual data was compared to the LSTM predictions to determine the degree of error and inform the optimization process.

Accuracy and Loss

```
Epoch 29/30
27/27 [=====] - 0s 6ms/step - loss: 116.6517 - binary_accuracy: 0.5691
Epoch 30/30
27/27 [=====] - 0s 6ms/step - loss: 116.6569 - binary_accuracy: 0.5691
(230, 37)
0.5626322031021118
```

Figure 7: Training with only user Activity data

```
Epoch 80/80
77/77 [=====] - 0s 882us/step - loss: 0.2300 - binary_accuracy: 0.9416
0.8989899158477783
Num_normal: 163 Num_anomalous: 35
Dnn: 159 Dra: 19
0.898989898989899
FPR: [0 0 0 0 0 0
```

Figure 8 : Training with Mixed Data : User Activities/features and action sequence

By using only one user feature i.e. activity data as shown in figure7, to train the LSTM in detecting malicious behavior, even though it showed success in indicating deviation from normal behavior to abnormal behavior it had a lower average accuracy of 0.56.

By combining the various features and training the LSTM on user daily activities, action sequence, and role features, a better model was able to be trained with an average accuracy of 0.899 and a lower loss of 0.23 as shown in figure 8. This is an indication that combing the all the user features in a better indicative way of identifying anomalous user behaviors

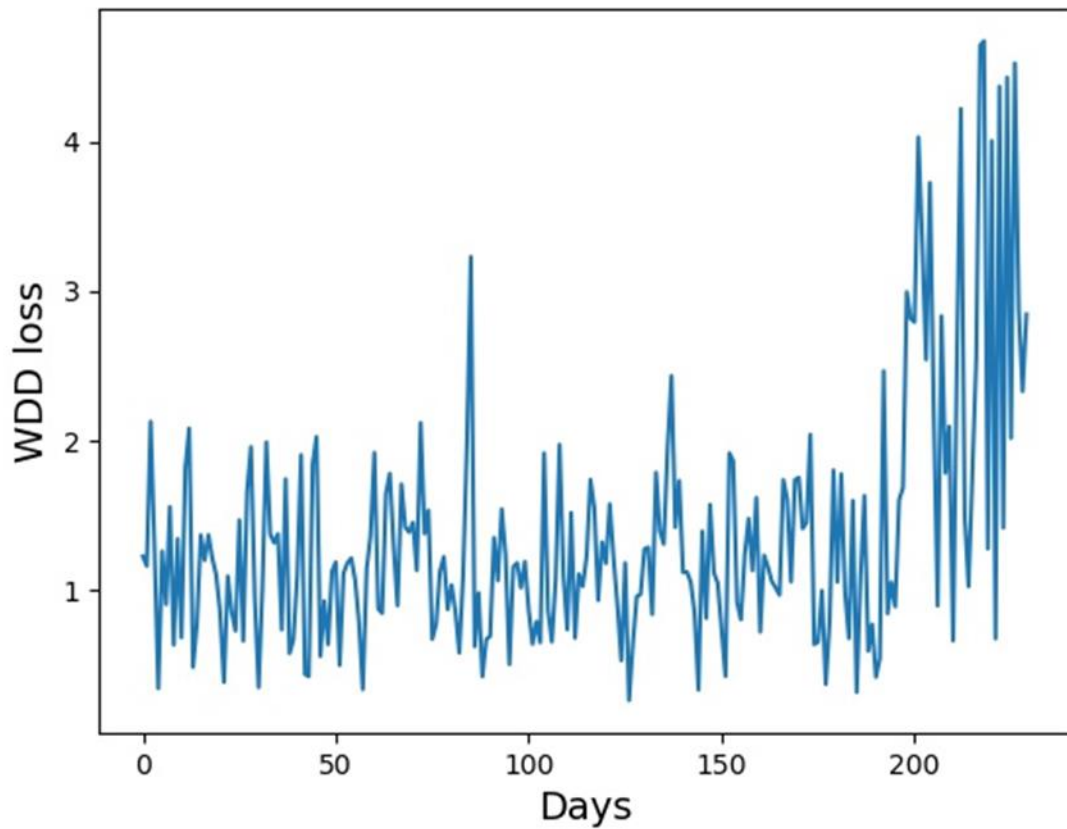


Figure 9 :WDD loss

It was discovered that the first 40 days of the test data have a similar distribution to the first 160 data by computing the WDD loss of the training dataset and the test data.

Due to the user's unusual conduct close to the 200th day, which caused an anomalous discrepancy between the projected behavior and the actual true sequences, there is a notable rapid change in loss on that day.

It can be therefore deduced that the LSTM was able to learn the user's action behaviors and detect if there is a change in the pattern of the user's action sequences.

4.3 DISCUSSION

In this experiment, to achieve better results, some issues need to be taken into consideration:

Size of training set: In this experiment, for the results to be meaningful we had to use a training set of 160. For this experiment, 4 days of data were used for training and the last 5th day for testing data. For this number, we were not able to get an anomaly score until we had 160 training records

Impact of variance in training data: Small deviations can lead to extraordinarily high anomaly scores if values for a given feature are too consistent in the training data. The value of the anomaly score should always be considered in the context of the original training data. If an employee always starts work at the same time and is late by just one minute on the day of scoring, this will be flagged as a significant outlier.

Feature Weighting: When giving certain features distinct weights, extreme caution should be used. All characteristics ought to be given equal weight unless the use case requires otherwise.

5 CONCLUSION

The results presented above showed that using one feature alone is not very accurate in detecting anomalous behaviors but to accurately detect anomalous behavior by checking deviations from all the user features (user's action sequence, user's daily activity data, and user's role features) proved to provide better results in detecting malicious behaviors.

5.1 LIMITATIONS

In conducting this research, the effectiveness of the approach discussed in this paper may raise issues due to certain limitations below.

Lack of enough data source – There are limited publicly available UEBA data that can be issued to train models in detecting anomalous behavior

Training data assumptions – Assuming the data used to train the model as normal behavior data therefore an initial attack from day one by the user may not be detected as the model will consider it as a normal user behavior

5.2 FUTURE RESEARCH

This research proposed detecting malicious user behaviors using User and Entity behavior analytics and was limited to only three features namely actions sequences, user's role features, and user's daily activities. For further study more user features should be included to get better results. This feature may include:

- Social features – user's social behavior can be added to the research
- Body tracking and gait analysis using depth sensors can be added to the research in the future for purposes of confidentiality.

- In future work also gesture recognition and body language can be added to discover the cases of lying and anxiety.

6 References

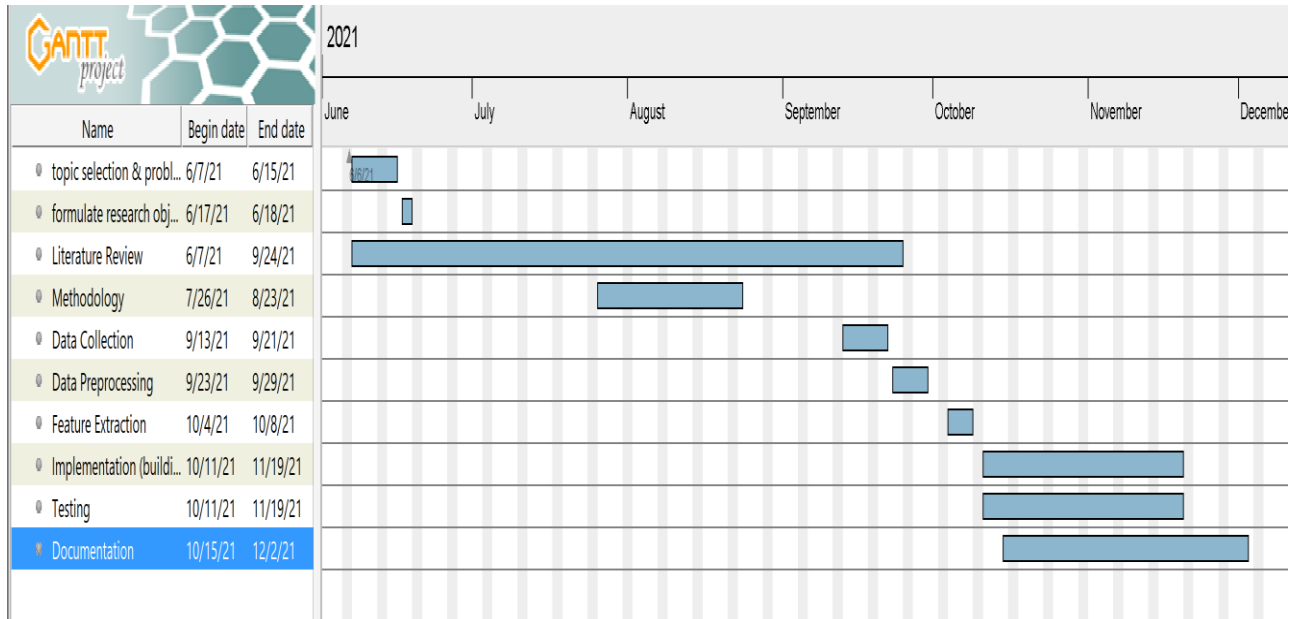
- Accenture Security. (2019). Unlocking the value of improved cybersecurity protection. *NINTH ANNUAL COST OF CYBERCRIME STUDY*, 1-2.
- Ashish, V., Noam, S., Parmar, N., Jakob Uszkoreit, & Jones, L. (2017). Attention Is All. *NIPS*.
- Cheng, Y., Fei Huang, L. Z., Jin, C., Zhang, Y., & Zhang, T. (2017). A Hierarchical Multimodal Attention-Based Neural Network for Image. *40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 889-892). NY, USA: ACM New York.
- Data Insider. (2021, August 2). *What Is User Behavior Analytics (UBA)/User Entity Behavior Analytics (UEBA)?* From Splunk: https://www.splunk.com/en_us/data-insider/user-behavior-analytics-ueba.html
- Duc C. Le, A. N.-H. (2018). Evaluating Insider Threat Detection Workflow Using Supervised and Unsupervised Learning. *IEEE Security and Privacy Workshops (SPW)*.
- Duc C. Le, A. N.-H. (2018). Evaluating Insider Threat Detection Workflow Using Supervised and Unsupervised Learning. *IEEE Security and Privacy Workshops (SPW)*.
- Duc C. Le, A. N.-H. (2019). Machine learning based Insider Threat Modelling and Detection. *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*.
- Graves, A. (2012). Long Short-Term Memory. In A. Graves, *Supervised Sequence Labelling with Recurrent Neural Networks* (pp. 37-45). Springer.
- Graves., A. (2013). . Generating Sequences With Recurrent Neural Networks.
- Hasim Sak, A. S. (2014). Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modelling. *Google, USA*.

- J. Glasser, B. L. (2013). Bridging the gap: a pragmatic approach to generating insider threat data. *Proceedings of 2013 IEEE Security and Privacy Workshops*, (pp. 98-104). San Francisco.
- Lin, D. (n.d.). *APPLYING DATA SCIENCE TO USER ENTITY AND BEHAVIOR ANALYTICS*. EXABEAM.
- Lindauer, B. (2020). *Insider Threat Test Dataset*. From Carnegie Mellon University:
<https://doi.org/10.1184/R1/12841247.v1>
- Martin, M. (2021, August 27). *Prototyping Model in Software Engineering: Methodology, Process, Approach*. From Guru99: <https://www.guru99.com/software-engineering-prototyping-model.html>
- Raguvir, S, P. S. (2020). Detecting Anomalies in Users – An UEBA Approach. *Proceedings of the International Conference on Industrial Engineering and Operations Management, Dubai, UAE, . Dubai,UAE*.
- Rapid7. (2021). *What is User and Entity Behavior Analytics (UEBA)?* From Rapid7:
<https://www.rapid7.com/fundamentals/user-behavior-analytics>
- Robert DiPietro, G. D. (2020). Chapter 21 - Deep learning: RNNs and LSTM. In D. R. Kevin Zhou, *Handbook of Medical Image Computing and Computer Assisted Intervention*, (pp. 503-519). London: Academic Press.
- Schmidhuber., S. H. (1997). Long Short-Term Memory. In S. H. Schmidhuber., *Neural Computation* (pp. 1735-1780).
- Shashanka, M. ,.-Y. (2016). User and Entity Behavior Analytics for Enterprise Security. *2016 IEEE International Conference on Big Data (Big Data)*.

- Shea, S. (2021, March). *SOAR (security orchestration, automation and response)*. From Search Security: <https://searchsecurity.techtarget.com/definition/SOAR>
- Shuhan Yuan, X. W. (2020). Deep Learning for Insider Threat Detection: Review, Challenges.
- T. Rashid, I. A. (2016). A new take on detecting insider threats: exploring the use of hidden markov models. *8th ACM CCS International Workshop on Managing Insider Security Threats*, pp. 47-56.
- Weizhi Meng, K.-K. R. (2018). Towards Bayesian-Based Trust Management for Insider Attacks in Healthcare Software-Defined Networks. *IEEE Transactions on Network and Service Management*.
- ZHIHONG, T. . (2020, September). User and Entity Behavior Analysis under Urban Big Data. *ACM/IMS Transactions on Data Science*,.
- Abrams, R. (2014). Target Puts Data Breach Costs at \$148 Million, and Forecasts Profit Drop. *The New York Times*.
- Alneyadi, S.,Sithirasenan, E., & Muthukkumarasamy, V. (2016)A survey on data leakage prevention systems. *J. Netw. Comput. Appl.*, 62, 137–152.
- Colwill, C. (2009). "Human Factors in Information Security: The Insider Threat–Who Can You Trust These Days?," *Information security technical report 14*(4), pp. 186-196.
- Crossler, R.E., Johnston, A.C., Lowry, P.B., Hu, Q., Warkentin, M., & Baskerville, R. (2013). Future directions for behavioral information security research. *Comput. Secur.*, 32, pp. 90–101.
- Ernst, Y. L., & Young, X. (2002). *Global Information Security Survey*. UK: Presentation Services.

- Farahmand, F., & Spafford, E.H. (2013). Understanding insiders: An analysis of risk-taking behavior. *Inf. Syst. Front.* 15, pp. 5–15.
- Ho, S.M., Kaarst-Brown, M., & Benbasat, I. (2018). Trustworthiness Attribution: Inquiry Into Insider Threat Detection. *J. Assoc. Inf. Sci. Technol.* 69, pp. 271–280.
- Loch, K. D., Carr, H. H., and Warkentin, M. E. (1992). Threats to Information Systems: Today's Reality, Yesterday's Understanding. *MIS Quarterly*, pp. 173-186.
- McCue, A. (2008). Beware the Insider Security Threat. *CIO Jury*.
- Mitnick, K. D., & Simon, W. L. (2001). *The Art of Deception: Controlling the Human Element of Security*. John Wiley & Sons.

APPENDIX 1



Appendix II

System Screenshots

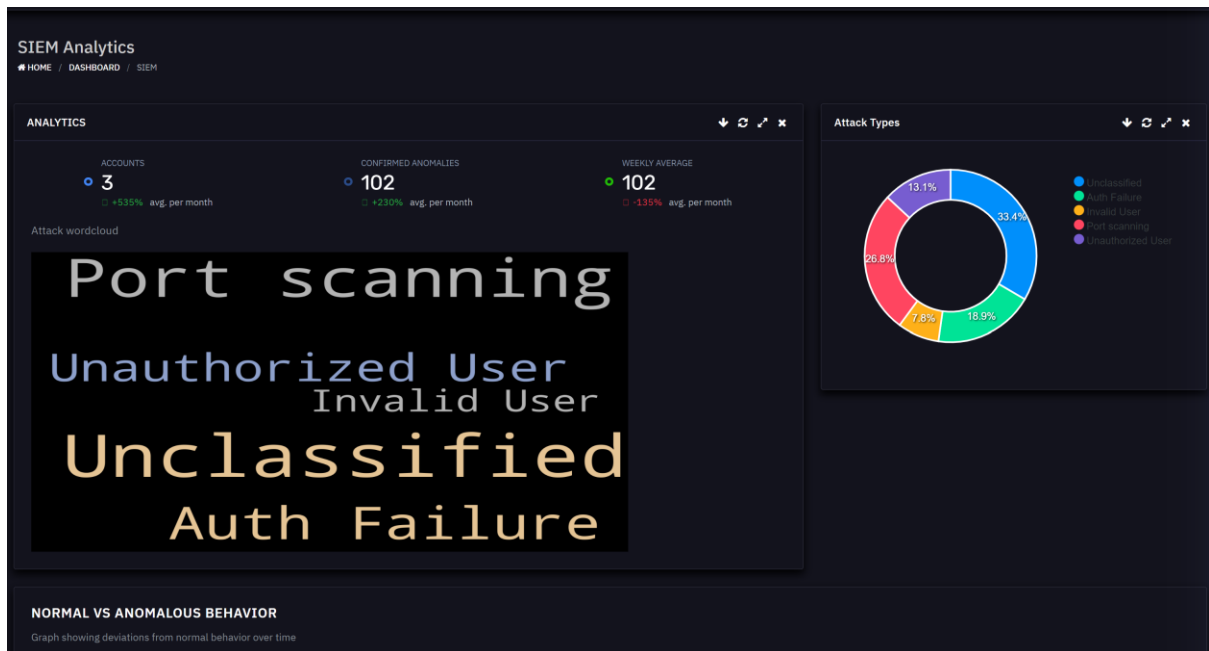


Figure 10 : Dashboard

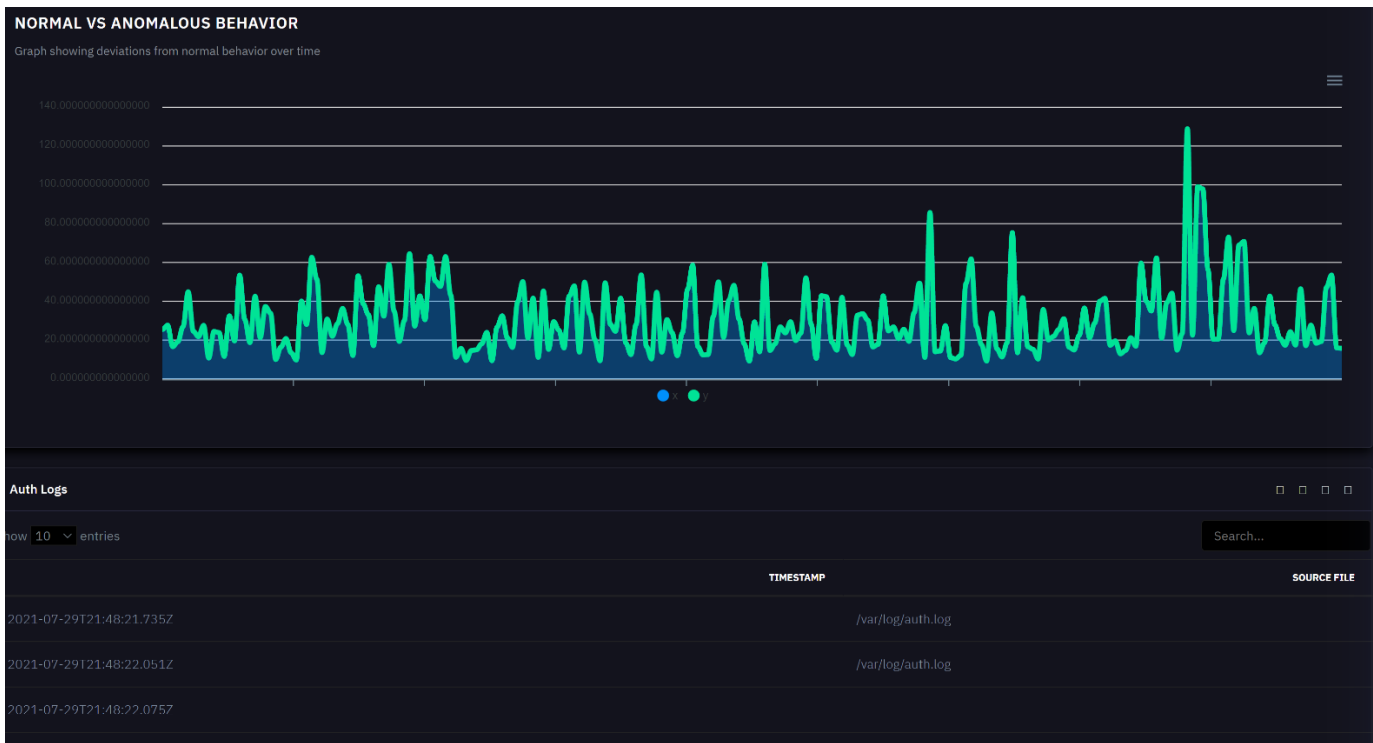


Figure 11 : Dashboard

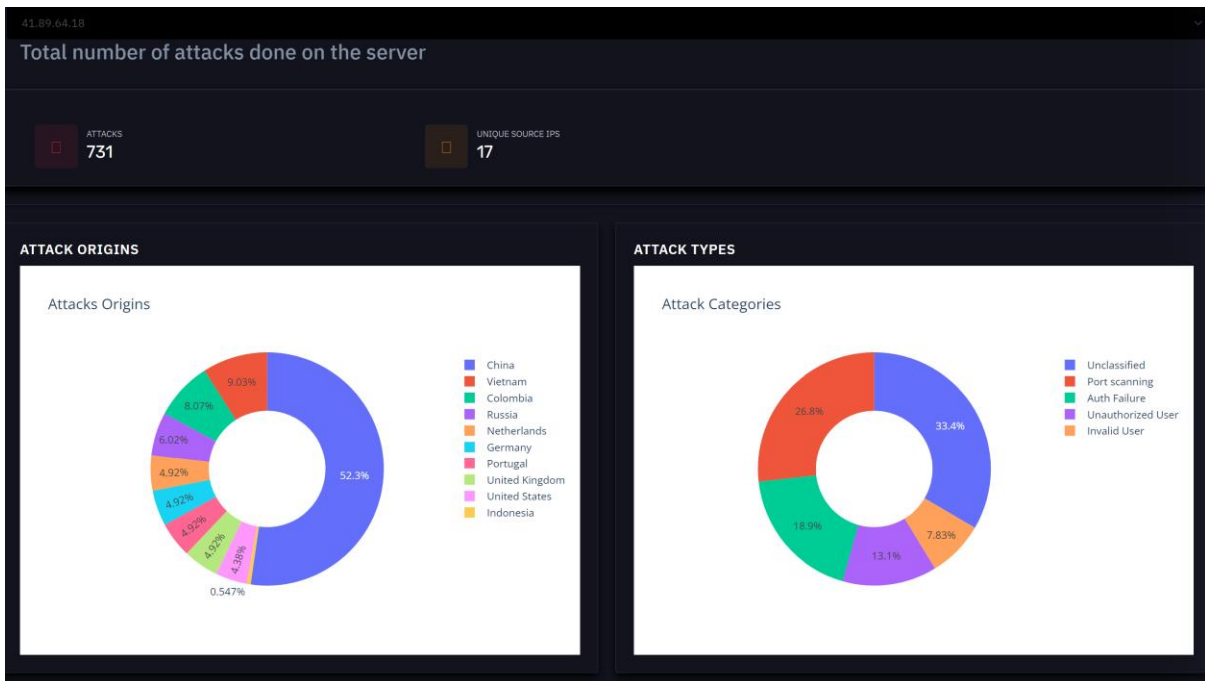
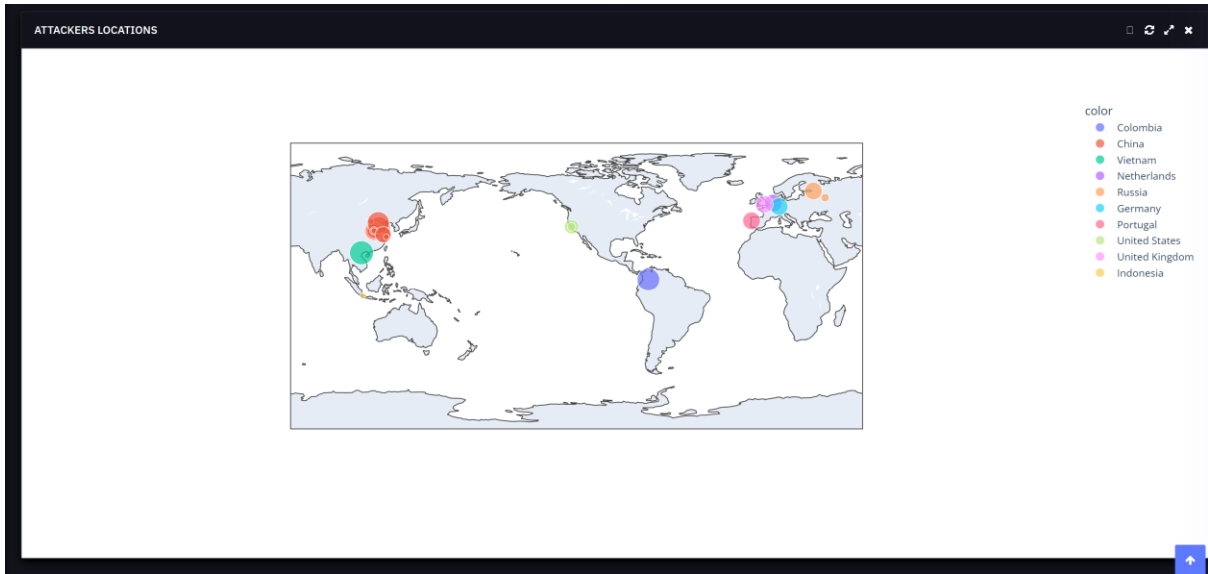



Figure 12: Attack Origins



EDB0714

Activity Profile Setting Contact



Eden Deborah Ball
ProductionLineWorker
Nairobi, Kenya

Alert User Block User

SOCIAL MEDIA

- Github Not provided
- Twitter Not provided
- Facebook Not provided

Full Name	Eden Deborah Ball
Email	Eden.Deborah.Ball@dtaa.com
Functional Unit	3 - Manufacturing
Department	3 - Assembly
Supervisor	Alan Benjamin Holder

User Analysis

Risk score

Online activity level

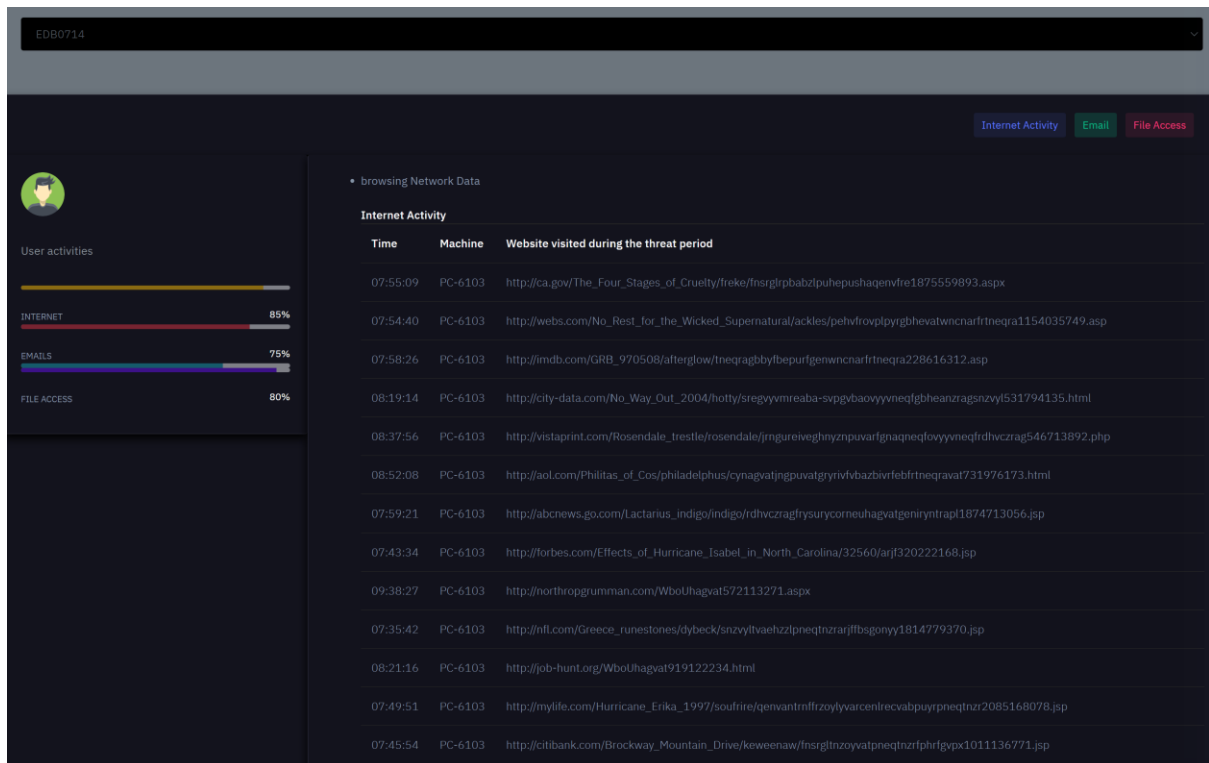


Figure 13: User logs

Code Snippets

```
# ----- (input)-----
x_train=np.loadtxt(files_train,delimiter=',')
y_train=np.loadtxt(labels_train,delimiter=',')

# -----(model structure)-----
main_input=Input(shape=(148,),dtype='float32',name='MainInput')
Inputs=Reshape((4,37))(main_input)
layer=LSTM(100,return_sequences=True,activation='tanh')(Inputs)
layer=LSTM(160,return_sequences=False,activation='tanh')(layer)
layer=Dense(37,activation='relu')(layer)
output=layer
model=Model(inputs=main_input,outputs=output)
model.compile(optimizer='adam',loss='mse',metrics=['binary_accuracy'])

# -----(save the model)-----
```

Figure 14: Model Design

```

# ----- define my loss
def my_loss_forFeatures(label_test,predict_save,myloss_save,figure_save):
    y_true=np.loadtxt(label_test,delimiter=',')
    y_pred=np.loadtxt(predict_save,delimiter=',')
    batch_size=np.shape(y_pred)[0]
    paras=np.array([1.2,1,0.2,0,0,0,0,1,0.1,1.2,\
                    2,1,1.5,\
                    0.5,1,1,0.5,0.1,0.7,\
                    0.5,1,1,0.5,0.1,0.7,\
                    0.5,1,1,0.5,0.1,0.7,\
                    0.5,1,1,0.5,0.1,0.7])
    All_loss=[]
    for i in range(batch_size):
        # Times=np.square((y_pred[i]+0.6).astype(np.int32)-y_true[i])
        # ---- Original
        Times=np.square(np.multiply((y_pred[i]-y_true[i]),paras))
        sums=Times.sum()/37
        All_loss.append(sums)
    x_list=range(0,batch_size)
    np.savetxt(myloss_save,All_loss,fmt='%f',delimiter=',')
    # ----- draw pictures
    plt.figure()
    plt.plot(x_list,All_loss)
    # plt.show()
    plt.savefig(figure_save)
    # print(np.shape(loss))

def Calculate_deviations(files_test,label_test,save_path,loss_save,figure_save,action_length):
    x_test=np.loadtxt(files_test,delimiter=',')
    y_test=np.loadtxt(label_test,delimiter=',')
    # print (np.shape(x_test))
    # print(np.shape([x_test[0]]))
    # print(len(x_test))
    # exit (0)
    model=load_model(save_path)
    # exit(0)
    All_loss=[]
    x_list=range(0,len(x_test))
    for i in range (len(x_test)):
        x_small_test=np.reshape(x_test[i],[1,4*action_length])
        y_small_test=np.reshape(y_test[i],[1,action_length])
        loss,acc=model.evaluate(x=x_small_test,y=y_small_test,verbose=0)
        All_loss.append(loss)
    np.savetxt(loss_save,All_loss,fmt='%f',delimiter=',')

```

Figure 15:Model Evaluation