

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



Attack Step Prediction of Targeted Attacks using Deep Learning

Authors

Imtiaj Ahmed Chowdhury - 160041048
A. H. M. Rezaul Karim - 160041022
Fardin Ahsan Sakib - 160041059

Supervisor

Dr. Muhammad Mahbub Alam, PhD.
Professor, Department of CSE,
Islamic University of Technology (IUT)

*A thesis submitted in partial fulfilment of the requirements for the degree of
B. Sc. Engineering in Computer Science and Engineering (CSE)*

Academic Year: 2019-2020

Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT),
A Subsidiary Organ of the Organization of Islamic Cooperation (OIC)
Gazipur-1704, Dhaka, Bangladesh

Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by *Imtiaj Ahmed Chowdhury, A. H. M. Rezaul Karim* and *Fardin Ahsan Sakib* under the supervision of *Dr. Muhammad Mahbub Alam, PhD*, Professor of the Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh.

We are very grateful to our supervisor *Prof. Dr. Muhammad Mahbub Alam, PhD*, Department of Computer Science and Engineering, Islamic University of Technology (IUT), for his supervision, knowledge and support, which has been invaluable for us.

It is also declared that neither of this thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Authors:

Imtiaj Ahmed Chowdhury



Student ID - 160041048

A. H. M. Rezaul Karim



Student ID - 160041022

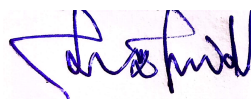
Fardin Ahsan Sakib



Student ID - 160041059

Approved By:

Supervisor:



Prof. Dr. Muhammad Mahbub Alam, PhD
Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT), OIC

Abstract

Defending against targeted attacks is becoming increasingly difficult as attackers are constantly evolving with more complex and intricate strategies. As more entities are falling victim to targeted attacks and the cost associated with such attacks is skyrocketing, the need for proactive defense is rising. A distinguishing feature of targeted attacks from other cyber attacks is they are mounted in multiple steps. Attackers follow a series of steps like recon, infiltration etc. to reach their final objective. Previous research tried to predict attack steps from IDS alerts and none of them specifically focused on targeted attack. Our key insight is that as targeted attackers employ stealthy and sophisticated approach, they often bypass traditional IDS solutions, rendering IDS alerts based attack step prediction ineffective. In this work, we propose a system that can predict future attack steps in a targeted attack from previously observed attack steps and provide cyber defenders an opportunity to preemptively block an attack. To the best of our knowledge, this is the first work to predict attack steps specifically for targeted attacks. We define attack steps based on ATT&CK framework. We leverage encoder-decoder architecture to build the system as it has been proven to be effective in Natural Language Processing (NLP) for sequence modelling. We test our system on APTGen dataset and show that it can predict the next step to be taken by attacker with 86.83% accuracy. We also show that our system is robust against adversarial manipulation by attackers.

Keywords: *Targeted Attack, ATT&CK Framework, Sequence to Sequence Model, Encoder-Decoder architecture*

Contents

1	Introduction	5
1.1	Overview	5
1.2	Research Challenges	6
1.3	Contribution	6
2	Background Study	6
2.1	Targeted Attack	6
2.2	MITRE ATT&CK Framework	7
2.3	Recurrent Neural Network (RNN)	9
2.4	Long-Short Term Memory(LSTM)	11
2.5	Encoder-Decoder Sequence to Sequence Model	12
3	Literature Review	13
4	Problem Formulation	15
5	Methodology	15
5.1	Model Architecture	16
6	Implementation	16
6.1	Dataset	17
6.2	Sequence-to-Vector Approach	17
6.3	Sequence-to-Sequence Approach	17
6.4	Sequence-to-Sequence with Augmented Data Approach	18
7	Experimental Results	19
8	Conclusion and Future Works	19

List of Figures

1	Cyber Kill Chain Model	8
2	Recurrent Neural Network	10
3	Long-Short Term Memory (LSTM) cell	11
4	Encoder-Decoder Model	13
5	Architecture Overview	16
6	An Attack Sequence	16
7	Sequence-to-Vector Model	17
8	Sequence-to-Sequence Model	18
9	Sliding Window Technique	18
10	Stacked Encoder-Decoder Architecture	19

1 Introduction

1.1 Overview

Among the cyber threats that possess the most danger in today's internet-connected landscape are targeted attacks. Due to their intricate nature, it is difficult to predict these attacks on time, and by the time they are detected, the attackers have already accomplished their objective. As targeted threat actors employ unconventional and unorthodox methods to accomplish their objectives, they are very difficult to detect using traditional Intrusion Detection System (IDS) solutions. Targeted attacks, unlike any other cyber-attacks, are deliberate, persistent, and purposeful. The purpose can vary from Industrial Espionage to Information theft to sabotage, and the attacks can range from any small scale organization to even large scale government agencies, including national defense agencies.

Targeted attacks due to their sophisticated nature are difficult to predict. Although some of the works claim to predict attacks with high accuracy but each have their limitations. Fava *et al* (1) used sequence modeling to predict future attack steps by utilizing alerts generated using the IDS. Ramaki *et al* (2) used the IDS generated alerts to construct meta-alerts which finally is converted to Bayesian Attack Graphs to predict future security events. The work of Perry *et al* (3) mostly depends on the dataset and is limited in performance compared to the real life attack scenarios. Tiresias (4) relies mostly on the Intrusion Detection System (IDS) for generating the alerts which are then used to predict the events of the attacks. But as targeted attack adversaries use stealthy techniques to accomplish their goals, it becomes easier for them to bypass the traditional IDS solutions. So these predictions are not quite effective against these targeted attacks. No prior work has been done on predicting the attack sequences of targeted attacks or the tactics and techniques the attackers would use. This was due to the unavailability of data about the attack sequences which we obtained from Takahashi *et al* (5). Using the attack sequences we can finally predict the tactics the attackers will execute in a targeted attack and also attain the techniques related to these attacks.

The tactics and techniques of the adversaries have been studied carefully and compiled to form the MITRE ATT&CK Framework (6). It contains the tactics that the attackers follow to get into the system and extract the information along with various techniques necessary to achieve these tactics. APTGen (5) uses this standard in order to create a sequence containing three tuples consisting of tactics, techniques, and softwares that the attackers might use. Incident reports of 8 different large scale attacks were analysed to create a total of 800 sequences that contains the tactics the adversaries followed, the techniques they used to accomplish the tactics and the software used in these techniques.

Using different deep learning models that can effectively predict the next sequences given the previous sequences, we can obtain the tactics the attacker might use along with the techniques which is going to be used to accomplish the attack. Predicting the next step of the attack can help us understand the motive of the attacker along with implementing proper security measures to defend against the attack and thus thwart the attacker. The prediction can also be used to find out the weakness of a security system by analyzing each step predicted by the model. Increasing the security of these steps can help prevent further attacks on the organizations.

That being said, a variant of the long-term memory provided by the Recurrent Neural Network (RNN) called the Encoder-Decoder Model is modified in order to perform the prediction of the attack sequences. The stacked LSTM architecture used in this Encoder-Decoder Model produces an attack sequence with the tactics and corresponding techniques with an accuracy of more than 86%.

1.2 Research Challenges

The research work presented a few major challenges that was overcome in the best possible way and a model capable of predicting a future attack step with an accuracy of 86.83% was attained. Preprocessing the collected data in order to fit the encoder-decoder model was the first major issue. Implementation of one-hot vectors and modifying the encoder-decoder model provided the solution in this case. The shortage of data due to unavailability of attack sequences of targeted attack was the most difficult challenge. The reluctance of the victim organizations to share the data regarding the cyber attack is the reason behind the lack of data in case of targeted attacks. Due to the presence of sensitive information, a lack of open source datasets are also observed. However, the Sequence to Sequence model requires a large amount of data to be able to fully capture the relationship between the sequences. Thus, the implementation of sliding window was introduced which augmented the dataset and helped in better capturing the inter-relation between the attack sequences. Finally, the possibility of a wrong step prediction being propagated in prediction of future steps was a complication that was resolved using an enhanced inference mode where in case of any wrong predictions, the actual attack step will be used in order to predict the future attack steps so that error propagation can be prevented.

1.3 Contribution

Predicting the attack steps using IDS generated alerts will not always give the best outcome as not all attack steps are acknowledged by the IDS as a threat. Thus, a different approach is required that can provide a more reliable prediction about the attack steps during a targeted cyber attack. Our contributions, therefore, can be listed as follows:

1. We predict the next step an attacker will take using an Encoder-Decoder Model for Sequence to Sequence prediction and enable the defender to be proactive in taking the countermeasures to the predicted steps.
2. The attack sequence generated contains the tactics with the corresponding techniques that the adversary may follow during the attack. This allows us to take countermeasures to block that particular tactic and thus stop the attack.
3. We provide a novel approach in predicting targeted attacks that have never been predicted before and obtain a high accuracy in generating the attack sequences.
4. Our prediction model does not require the security information of the organization. Thus, the model is universal and can be used in general.

2 Background Study

2.1 Targeted Attack

In Targeted Attacks, the adversaries perform well-planned objectives and employ sophisticated tactics which is different from other type of cyber-attacks. A targeted attack (7) consists of three main criteria:

1. A specific target is chosen and considerable time, resources, and effort is spent in setting up or carrying out the targeted attack.
2. Infiltrating the target's network and stealing information from their servers is the primary objective.
3. Persistence of attack is noticed, along with ensuring that the attack continues beyond the initial network penetration and infiltration of data.

A successful targeted attack causes great loss for the organization or agency that was attacked. Targeted attacks are generally aimed at high profile organizations. Data breach of these organizations would cost a significant blow to their reputation and also generate substantial profit for the attackers. As an example, in the Ashley Madison hack in 2015 (8), the company incurred a financial loss of around 12 million USD along with billions of leaked user information.

Phases of a Targeted Attack:

Targeted attacks are performed in different phases. They can be listed as such:

1. **Intelligence gathering:** The primary phase of targeted attack is to identify and collect publicly available information about the target organization. Information about the target's IT environment and organizational structure along with recent events and work related issues are gathered.
2. **Point of entry:** Customized spearphishing email, zero-day or software exploits, and watering hole techniques are used to infiltrate the infrastructure. A connection is established through any means with the target organization.
3. **Command-and-control (C&C) communication:** Threat actors communicate with the malware they placed inside the infrastructure during the security breach. These communications are hidden as best as possible.
4. **Lateral movement:** Seeking key information or infecting other valuable systems are the main activities at this point.
5. **Asset/Data Discovery:** Adversaries identify assets and data to be exfiltrated and gain access to the valuable and noteworthy assets. Tools such as Remote Access Trojan (RAT) are used to transfer these data.
6. **Data Exfiltration:** In this final stage, attackers transfer the gathered information to a location of their choice. This is done gradually or quickly as per the attacker's wish.

Targeted attacks are called multi-step attacks as they are carried out in a series of steps. In order to maximize the chance of their success, they follow what is commonly known as the *Cyber Kill Chain Model* (9). The model has the following set of steps that combined together constitute the tactics of mounting the targeted attacks. It is a classic cybersecurity model designed by the *Computer Security Incident Response (CSIRT)* whose purpose is to better understand the stages an attack must go through to conduct a targeted attack, and help security teams stop the attack at each stage.

2.2 MITRE ATT&CK Framework

ATT&CK framework is a globally-accessible knowledge base of different tactics and techniques the adversaries use during any cyber-attacks. Adversarial Tactics, Techniques Common Knowledge (ATT&CK) is developed by MITRE corporation which models the attacker's strategies that is observed in real world. It provides a common vocabulary for threat analysis and research and is widely used by security professionals. Two main versions of this framework is available.

- **Enterprise :** Focuses on adversarial behavior in Windows, Mac, Linux, and Cloud environments
- **Mobile :** Focuses on iOS and Android operating systems



Figure 1: Cyber Kill Chain Model

The MITRE ATT&CK framework consists of 14 tactics. These are described in detail:

1. **Reconnaissance:** The active or passive collection of information regarding the targeted organization is considered as the Reconnaissance step. This include details of the victim organization, infrastructure, or staff/personnel. Some of the techniques for this step are Active Scanning, Scanning IP Blocks, etc.
2. **Resource Development:** Adversaries create, purchase or compromise resources to support the targeting of the organization in the Resource Development stage. Among the resources include infrastructure, accounts, or capabilities. For example, Acquiring Infrastructure, Compromising Accounts, etc., is done in this step.
3. **Initial Access:** In this step, attackers use spearphishing or similar techniques to gain foothold. This may result in continued access by validating fake accounts and using remote services. Some of the techniques used in this tactic are Drive-by Compromise, External Remote Services etc.
4. **Execution:** After gaining control, the adversaries run their code on a local or remote system. For example, an adversary might use a remote access tool to run a PowerShell script that does Remote System Discovery.
5. **Persistence:** Using this tactic, the adversaries can keep their access across restarts, changed credentials, and other interruptions that could cut off their access. Techniques used for persistence include any access, action, or configuration changes that let them maintain their foothold on systems.
6. **Privilege Escalation:** Adversaries use this tactic to gain higher-level permissions on a system or network. They perform various techniques to take advantage of system weaknesses, misconfigurations, and vulnerabilities.
7. **Defense Evasion:** In order to avoid the detection mechanisms of the organization's infrastructure, the attackers use Defense Evasion tactics. Techniques used for defense evasion include uninstalling/disabling security software or obfuscating/encrypting data and scripts. Adversaries moreover leverage and abuse trusted processes to cover up and disguise their malware.

-
8. **Credential Access:** Stealing credentials like account names and passwords are performed in this tactic. Techniques used to get credentials include keylogging or credential dumping.
 9. **Discovery:** An adversary gains knowledge about the internal network and the system during this tactic. Account Discovery, Browser Bookmark Discovery etc. are some of the discovery techniques.
 10. **Lateral Movement:** In order to reach the goal, the attacker needs to move through multiple systems and accounts to gain access to the objective. Adversaries might install their own remote access tools to accomplish Lateral Movement or use legitimate credentials with native network and operating system tools, which may be stealthier.
 11. **Collection:** Adversaries use collection tactic to gather the information and sources that are relevant to their objectives. Common collection methods include capturing screenshots and keyboard input.
 12. **Command and Control:** Adversaries communicate with the systems under their control in the victim network. To do so, adversaries commonly attempt to mimic normal, expected traffic to avoid detection. An adversary can establish command and control with various levels of stealth depending on the victim's network structure and defenses like the Application Layer Protocols, Communication Through Remote Media etc.
 13. **Exfiltration:** Once data collection step is finished, it is packaged to avoid detection while removing it. This packaging may include encrypting and compressing the data and then transferring the data through the command and control channel or any alternate channels.
 14. **Impact:** In order to disrupt availability or compromise integrity by manipulating business and operational processes, the attackers employ this tactic. Techniques used for impact can include destroying or tampering with data. In a few cases, business processes can look fine, but may have been modified to advantage the adversaries' objectives. These techniques might be used by adversaries to follow through on their end goal or to provide cover for a confidentiality breach.

Each tactic contains an array of techniques that have been observed being used in the wild by malware or threat actor groups in compromises. Techniques are thought of as how attackers are escalating privileges or how adversaries are exfiltrating data, etc. The number of techniques are too many to list but can be visualized using the MITRE ATT&CK Navigator which is a web based application.

2.3 Recurrent Neural Network (RNN)

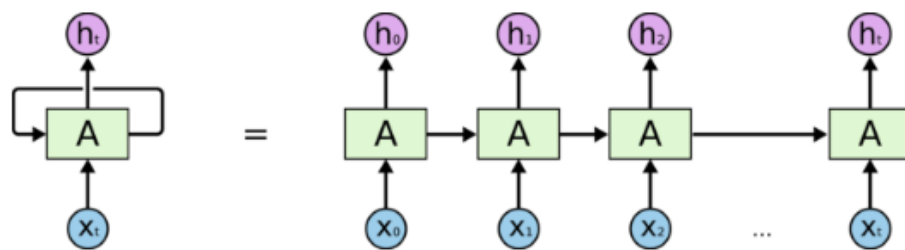
Neural Networks(NN) are a series of algorithms that are programmed to identify patterns and closely imitate the human brain. They use machine perception to perceive sensory data, labeling or clustering raw data. They can identify numerical patterns contained in vectors, into which all real-world data (images, sound, text or time series), must be translated. Artificial Neural Networks(ANN) are made up of several highly interconnected processing elements (neurons) that work together to solve a problem.

An ANN is made up of a large number of processors that operate in parallel and are organized in tiers. The raw input information is received by the first tier, which is similar to the optic nerves in human visual processing. In the same way that neurons further away from the optic nerve receive signals from those closer to it, each successive tier receives the output from the tier

before it rather than the raw input. The last tier produces the output of the system.

Recurrent Neural Network(RNN) (10), (11) learns from the past and then uses that information to predict a future incident. While RNNs learn while training, they remember things learnt from prior inputs while generating outputs. RNNs can take one or more input vectors and produce one or more output vectors and the outputs are influenced not just by weights applied on inputs like a regular NN, but also by a “hidden” state vector representing the context based on prior inputs/outputs.

Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. In other neural networks, all the inputs are independent of each other. But in RNN, all the inputs are related to each other.



An unrolled recurrent neural network.

Figure 2: Recurrent Neural Network

First, it takes the X_0 from the sequence of input and then it outputs h_0 which together with X_1 is the input for the next step. So, the h_0 and X_1 is the input for the next step. Similarly, h_1 from the next is the input with X_2 for the next step and so on. This way, it keeps remembering the context while training.

The formula for the current state is

$$h_t = f(h_{t-1}, x_t)$$

And the activation function is

$$h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t)$$

W is weight, h is the single hidden vector, W_{hh} is the weight at previous hidden state, W_{hx} is the weight at current input state, \tanh is the activation function, that implements a non-linearity that squashes the activations to the range[-1.1] And the output is

$$y_t = W_{hy}h_t$$

Y_t is the output state. W_{hy} is the weight at the output state.

2.4 Long-Short Term Memory(LSTM)

Long-Short Term Memory(LSTM) (12) is a derivative of recurrent neural network (RNN). It has a similar control flow as a recurrent neural network. It processes data passing on information as it propagates forward. The differences are the operations within the LSTM's cells.

The main concept of LSTM is the cell state, and it's various gates. The cell state act as a transport highway that transfers relative information all the way down the sequence chain. It can be called the "memory" of the network. The cell state can carry relevant information throughout the processing of the sequence. So even information from the earlier time steps can make its way to later time steps, reducing the effects of short-term memory. As the cell state goes on its journey, information gets added or removed to the cell state via gates. The gates are different neural networks that decide which information is allowed on the cell state. The gates can learn what information is relevant to keep or forget during training. There are 4 gates in total in a LSTM cell. They can be defined as follow:

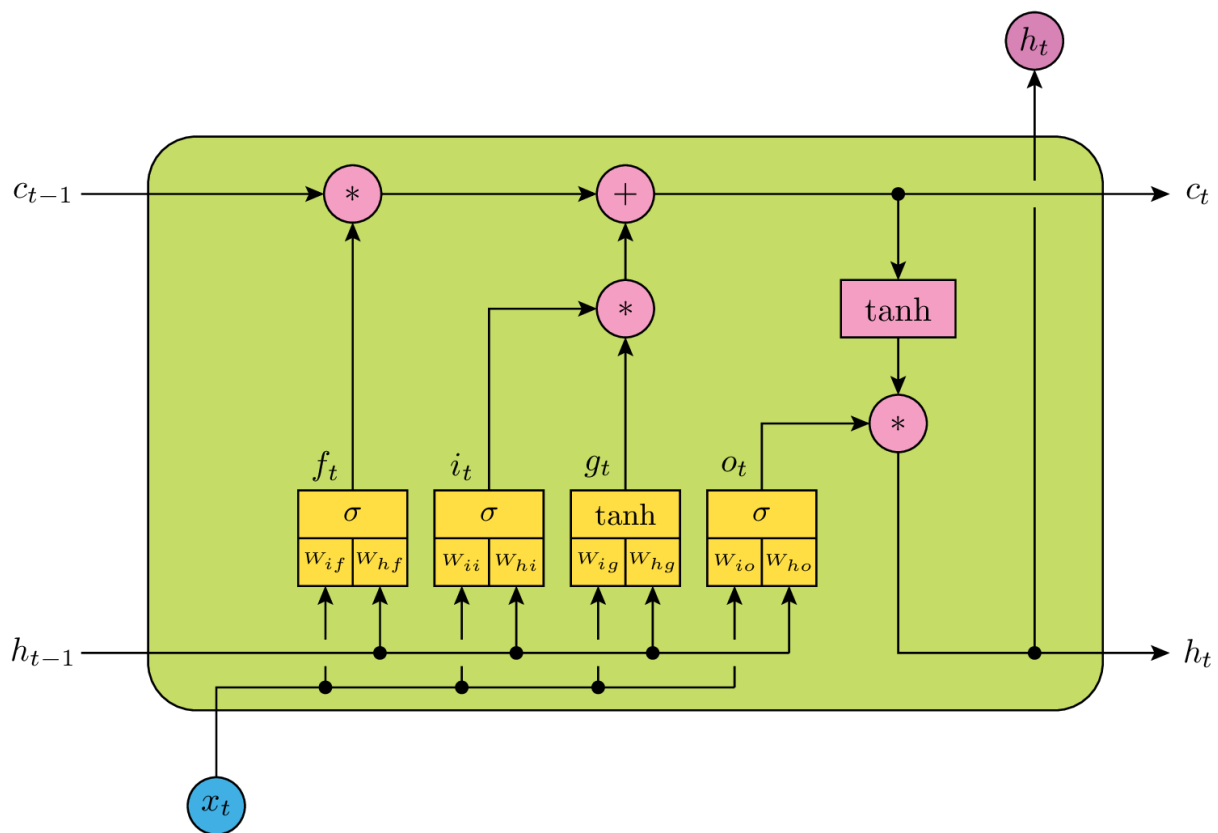


Figure 3: Long-Short Term Memory (LSTM) cell

1. **Forget Gate (f_t):** The Forget gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep. Equation of forget state can be written as:

$$f_t = \sigma(W_f * [h(t-1), x_t] + b_f)$$

2. **Input Gate (i_t) and Candidate Cell State (g_t):** The input gate is used to update the cell state. The previous hidden state and current input are passed into a sigmoid function that decides which values will be updated by transforming the values to be between 0 and 1. 0

means not important, and 1 means important. The hidden state and current input are passed into the tanh function to squish values between -1 and 1 to help regulate the network. Then the product of the tanh output and the sigmoid output are determined. The sigmoid output will decide which information is important to keep from the tanh output. The equations of input gate and candidate cell state can be given as:

$$i_t = \sigma(W_i * [h_{(t-1)}, x_t] + b_i)$$

$$g_t = \tanh(W_g * [h_{(t-1)}, x_t] + b_g)$$

3. **Cell State(C_t):** The cell state gets pointwise multiplied by the forget vector. This has a possibility of dropping values in the cell state if it gets multiplied by values near 0. Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state. The cell state equation is obtained as:

$$C_t = f_t * C_{(t-1)} + i_t * g_t$$

4. **Output Gate (o_t) and Hidden State(h_t):** The output gate decides what the next hidden state should be. The hidden state contains information on previous inputs and is also used for predictions. The previous hidden state and the current input are passed into a sigmoid function. Then the newly modified cell state is passed to the tanh function. The product of the tanh output and the sigmoid output decides what information the hidden state should carry. The output is the hidden state. The new cell state and the new hidden is then carried over to the next time step. The output gate equation and hidden state equations can be written as:

$$o_t = \sigma(W_o * [h_{(t-1)}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

2.5 Encoder-Decoder Sequence to Sequence Model

A sequence to sequence model (13) commonly known as seq2seq model maps a fixed-length input with a fixed-length output where the length of the input and output may differ.

Generally, a sequence to sequence model has two parts, an encoder and a decoder. Both the encoder and decoder are composed of neural networks that are combined to form the encoder-decoder model. They are linked by an encoder vector or context vector which is shown in figure 4.

Encoder

An Encoder is constructed using a stack of several recurrent units where LSTM or GRU cells for used for better performance. Each of the stacks accepts a single element of the input sequence, collects information for that element and propagates it forward. The hidden states h_i are computed using the formula-

$$h_t = f(W^{(hh)}h_{t-1} + W^{(hx)}x_t)$$

This formula represents the result of an ordinary recurrent neural network. The appropriate weights are applied to the previous hidden state $h_{(t-1)}$ and the input vector x_t .

Context Vector/Encoder Vector

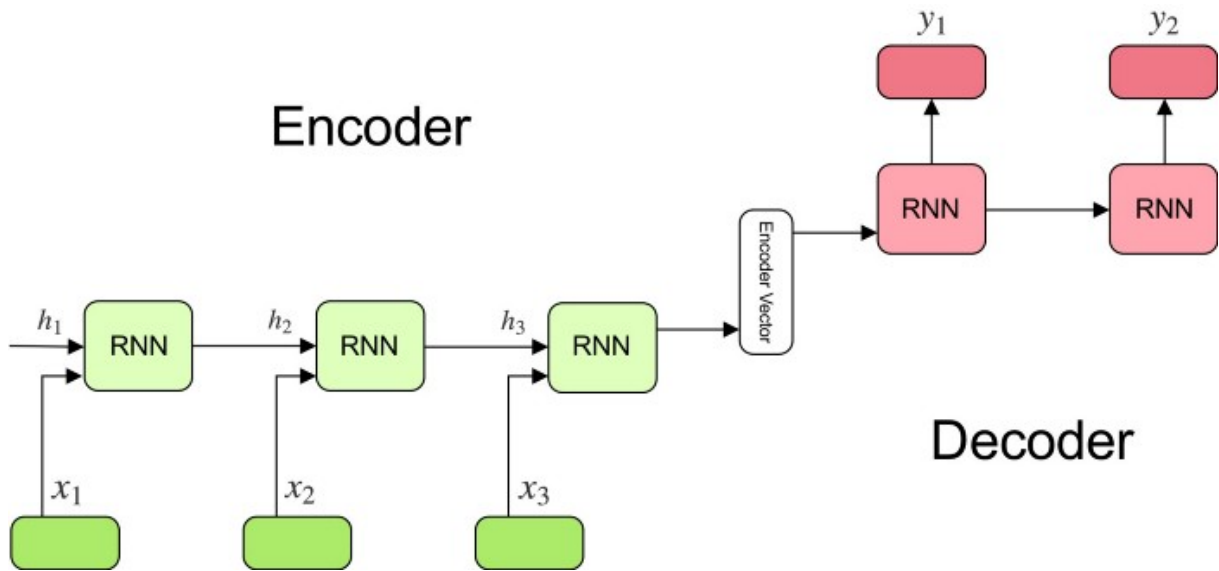


Figure 4: Encoder-Decoder Model

The Context Vector is the final hidden state produced from the encoder part of the model. It is calculated using the formula-

$$h_t = f(W^{(hh)}h_{t-1} + W^{(hx)}x_t)$$

which is similar to finding all other hidden states. This vector aims to encapsulate the information for all input elements in order to help the decoder make accurate predictions. It acts as the initial hidden state of the decoder part of the model.

Decoder

A Decoder is constructed using a stack of several recurrent units where each predicts an output y_t at a time step t . Each recurrent unit accepts a hidden state from the previous unit and produces an output as well as its own hidden state. Any hidden state h_i is computed using the formula:

$$h_t = f(W^{(hh)}h_{t-1})$$

The output y_t at time step t is computed using the formula:

$$y_t = \text{softmax}(W^S h_t)$$

The outputs are calculated using the hidden state at the current time step together with the respective weight W^S . Softmax is used to create a probability vector which will help in determining the final output.

The power of this model lies in the fact that it can map sequences of different lengths to each other. We utilize this fact in order to generate the attack steps of various lengths.

3 Literature Review

Previous network security research has centered on the development of intrusion detection systems (IDSs) to detect malicious network traffic and device use, as well as algorithms to analyze IDS alerts. The increasing number of deployed IDSs produced a number of alerts that

became distressing to human analysts as the complexity and scale of networks increased. As possible solutions to this issue, a number of methods for producing detailed warning reports have been suggested. Threat projection is one of these, and it's a key move in avoiding critical attacks. Previous attack projection research has relied heavily on Apriori knowledge of network and system configurations, and as a result, has been challenged by the diversity and ever-changing nature of system settings and exploitation methods. Alternatively, without knowing the underlying network configuration, the framework presented by Fava *et al* (1) models malicious network activity and extracts relevant information about an attacker's overall behavior and intent. The method presented in his paper borrows from sequence modeling techniques. A variable-length Markov model (VLMM) is developed from previously observed as well as ongoing attack sequences to predict the attacker's likely future attack steps and actions. This research investigated the application of sequence modeling techniques in the context of cyberthreat projection by interpreting cyberattacks as sequences of malicious actions observed through IDS alerts.

The intrusive activities in a network are detected using the Intrusion Detection System (IDS) which generates an alert when a known intrusion activity is detected. Large number of low-level alerts generated by the IDS is the reason why a novel alert correlation system was needed to be introduced. The work of Ramaki *et al* (2) thus proposes a novel alert correlation framework which processes the generated alerts in real time, correlate the alerts, construct the attack scenarios using the concept of Bayesian networks and forecasts the next goal of attackers using the creation of attack prediction rules. The framework has two modes called the off-line mode where a Bayesian Attack Graph (BAG) is constructed using the concept of Bayesian networks and an on-line mode where the most probable next steps of the attacker are predicted. The paper claims that the framework is efficient enough in detecting multi-step attack strategies without using any predefined knowledge and also the algorithm can perfectly forecast multi-step attacks before they can compromise the network. Aggregating the alerts generated by the IDS, meta-alerts are constructed which is used to build the *Bayesian Attack Graph*(BAG). The framework then learns to predict multi-step attack scenarios. Then, for modifying the probabilities, the changes in meta-alerts states are propagated through the BAG. A prediction facility to predict the next steps in a multi-step attack is also available in the framework which has an accuracy greater than 90% in some cases. However, the framework primarily relies on the IDS to generate alerts. But Intrusion Detection Systems cannot detect any Zero-day exploits and also fails to generate alerts in case of rare intrusion techniques. So, it is not possible to predict the attack steps in case of an attack unknown to the IDS.

The research performed by Perry *et al*(3) presents the use of RNN in modeling of penetration behavior by ten teams in the 2017 Collegiate Penetration Testing Competition (CPTC' 17). Models with fair predictive ability can be learned based on observable of early behavior in a cyberattack episode. This hypothesis does not mean that data is homogeneous or stationary in an episode rather real or continuous data is considered. The aim is to see whether and how well earlier attack behavior have distinguishing characteristics that can be used to anticipate and differentiate potential attacks. Analysing the data from CTPC' 17, the use of LSTM on training the model and predicting future security events based on past actions are determined.

Shen *et al* (4) developed Tiresias which is a system that is developed using deep learning techniques which learns from past system events and predicts the event that is most likely to happen next in case of a cyber-attack. In this case, they are concerned with predicting the next security event or the exact action that will be taken by the attacker. The system can provide more precise predictions about the subsequent events compared to previous works. This will allow the companies to deploy countermeasures and thus thwart the actions of the attackers. The dataset used in Tiresias is of 3.4 billion security events collected by commercial intrusion prevention

systems from 740k machines over a period of 27 days. The model uses a Recurrent Neural Network in order to construct a prediction engine from the collected and preprocessed data generated from the intrusion protection programs. Tiresias specifies a probability distribution of next possible security event given the historically observed events. The Long Short Term Memory(LSTM) is used to capture this relation between sequences of the security events. A probability distribution of all the possible security events is obtained from the prediction engine. The event with maximum probabilistic score is chosen. In case of wrong prediction, the contextual information is updated accordingly. The system keeps track of the prediction performance and when it drops below a threshold, the model is retrained. When the adversaries attack the different systems, the IDS records the event data. It is then sent to the pre-processing module. The pre-processing module reconstructs the sequences, which go on to the training and validation step. The training and validation will take place in the Prediction Engine. There is one added module that compares the predicted results with the actual results, which is called the Prediction Performance Monitor. If the performance falls below a certain threshold, the model needs to be trained again with newer data. In this way, the performance of the system is sustained. After evaluation, Tiresias shows significantly better performance than other models. It can be observed that if the model is trained with more data, the performance is also better. And most importantly, the model is stable, meaning that if the model is trained with data from some time back, it still can show reasonable performance similar to the model trained with data from the same time. However, there are some weaknesses associated with the system. First, the model is trained and tuned per dataset, so it will fail to detect any intrusion attempt not belonging to the dataset. Also, inevitably, it may fail to correctly predict the rare intrusion attempt or zero-day exploits. It might not be feasible to deploy in environments with less computational power.

4 Problem Formulation

Our goal is to predict the next attack steps to be taken by an attacker given a set already observed attack steps. As mentioned in section 2.2, we use ATT&CK framework to formulate our problem definition and have a common vocabulary to avoid ambiguity. We define an *attack step* s_i as a tuple of ATT&CK Tactics and Techniques, $s_i = (Ta_i, Te_i)$ where Ta_i denotes the Tactic followed by attacker at step i and Te_i denotes the Technique to accomplish that tactic at step i . Having defined an attack step, we can now proceed to define an *attack sequence*. Formally, an attack sequence Seq_i is an ordered sequence of attack steps,

$$Seq_i = \{s_1, s_2, \dots, s_n\}$$

Here, n is the total number of attack steps in sequence. The problem is, given previous attack steps $\{s_0, s_1, \dots, s_i\}$ of an attack sequence Seq_i , learn to predict next attack steps $\{s_{i+1}, s_{i+2}, \dots\}$ of sequence Seq_i . Unlike previous works, we have characterized attacker actions from a high-level perspective to better capture attacker strategies. We believe our attack step formulation is more effective in predicting targeted attack steps as adversaries' high-level strategy usually remains the same even though their low-level execution differs.

5 Methodology

After analysing the problem at hand, it can be ascertained that the best way to predict a step in a sequence is through the use of different deep learning models. These can fully grasp the relation between different sequences and then retain the data in the memory so that an efficient output is obtained. Keeping that in mind, we explore the following approaches in order to develop a system that can predict a future attack step given the previous attack steps in a sequence.

The details of the architecture used along with the underlying methodology is explained in this section.

5.1 Model Architecture

As shown in figure 5, our proposed system consists of three modules namely *Input Processing*, *Training* and *Inference* module.

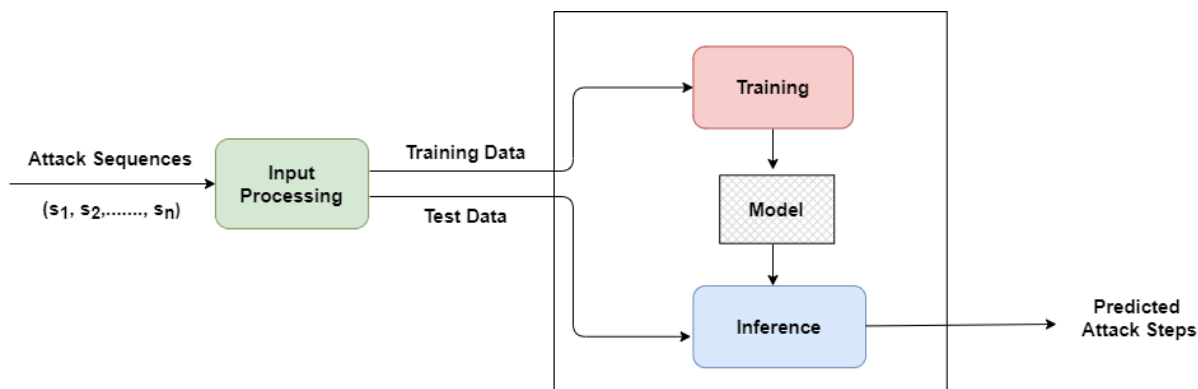


Figure 5: Architecture Overview

Input Processing module takes an attack sequence as input. The attack sequence consists of textual data. An example attack sequence is shown in figure 6.

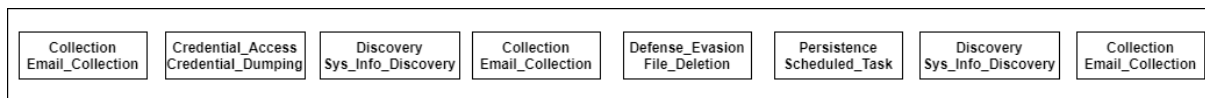


Figure 6: An Attack Sequence

As deep learning frameworks can only work with numbers, we convert the original attack sequence into a sequence of numbers where each attack step is uniquely represented by an integer. In some experiments, we perform additional pre-processing steps which are later described in corresponding experiments in section 6. Finally, we randomly select 80% attack sequence for training and rest 20% for testing.

Training module follows the encoder-decoder architecture as outlined in section 2.5. The set of observed attack steps is fed as input to the encoder. The encoder encodes the given input into a *context vector* as shown in figure 4. The decoder takes the context vector as input and learns to predict the next attack steps of the given attack sequence. Specific implementation details of encoder-decoder architecture in training mode is given in section 6. The trained model is then used in inference module to predict attack steps in real time.

Inference module uses the trained model and takes attack steps in real-time as input. The decoder predicts the next steps to be taken by the attacker. Specific details of inference module is given in corresponding experiments in section 6.

6 Implementation

The methodologies mentioned in section 5.1 about the three modules that needs to be implemented are described in this section. The dataset collected for training and validating the model is also mentioned in detail.

6.1 Dataset

The dataset generated by Takahashi *et al* (5) called the APTGen dataset containing the Targeted Attack Sequences is used in our work. This dataset contains the tactics, techniques and software that the adversaries use during their attack along with the log files generated during the attack period. It was created from the incident reports of 8 different large scale targeted attacks using some sophisticated tools where they simulated the attack environment and generated 800 attack sequences based on the reports.

We collected the dataset and removed the software used in each tactic as the software may vary and will lead to an unstable prediction result. Thus our dataset contains the tactics the adversaries use during the attack along with the techniques to accomplish those tactics.

6.2 Sequence-to-Vector Approach

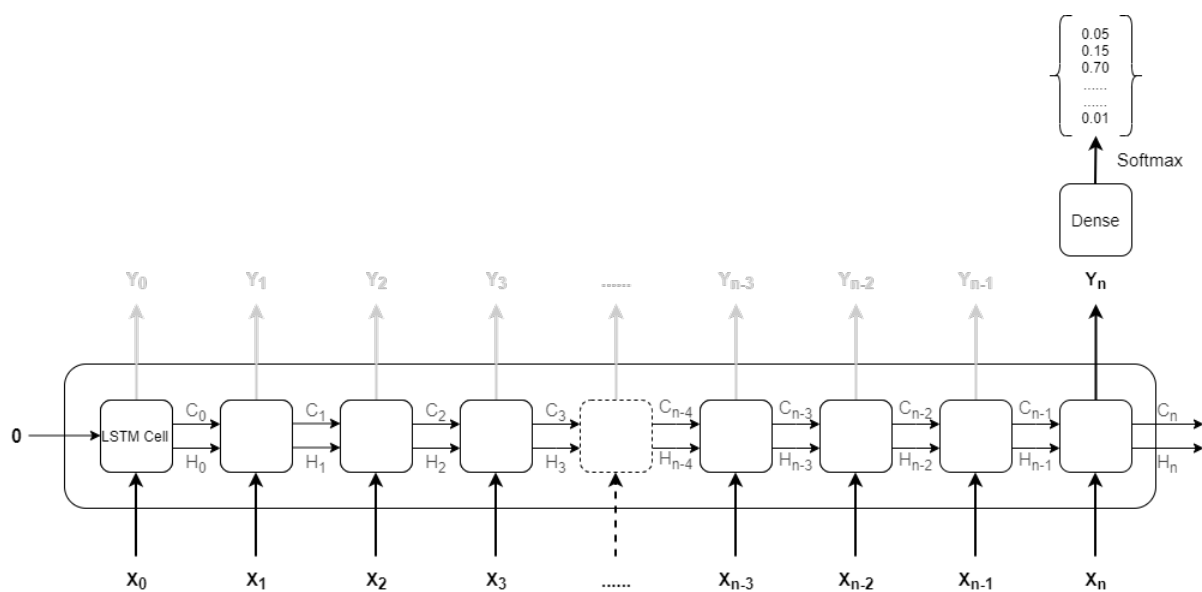


Figure 7: Sequence-to-Vector Model

In our initial approach, sequence-to-vector model was utilized. The goal is to predict the last step in an attack sequence given all the previous steps. We want to evaluate how the information from earlier attack steps can contribute in predicting the last step. Figure 7 illustrates the sequence-to-vector architecture we used. We only consider the last LSTM cell output and pass it through a Dense layer to get a probability distribution. The attack step with highest probability score is chosen as the predicted step. Table 1 shows that the performance of this model is not satisfactory with around 20% accuracy. Further investigation reveals that lack of enough data is the primary reason behind this.

6.3 Sequence-to-Sequence Approach

In this experiment, we use sequence-to-sequence model that leverages encoder-decoder architecture we have seen earlier. The goal is to learn inter-relationship and context between attack steps in the input sequence. The learned context is then utilized to predict the next attack steps in the attack sequence. This type of model has been proven to be very effective in Natural Language Processing (NLP) for sequence modelling. Figure 8 shows an encoder-decoder model being used in Neural Machine Translation (NMT). Table 1 shows that this model can predict future

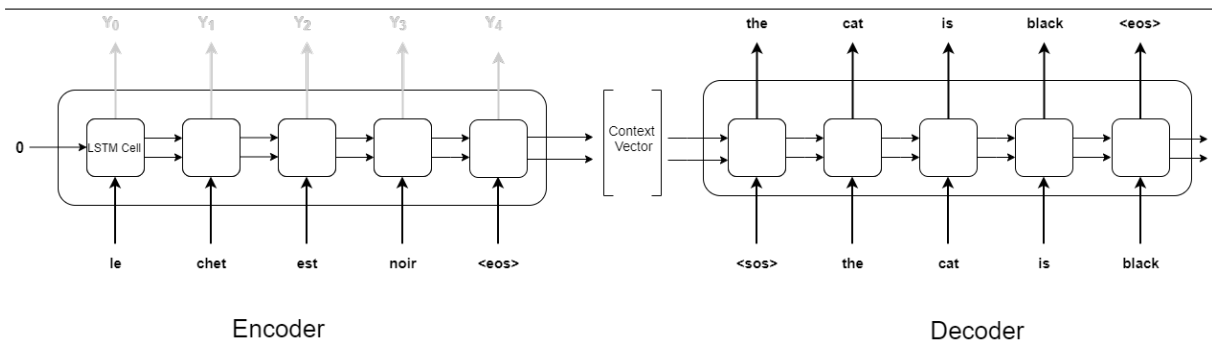


Figure 8: Sequence-to-Sequence Model

attack steps with almost 70% accuracy. Thus, it can be stated that the model can grasp the inter-relationship between the attack steps in the sequences.

6.4 Sequence-to-Sequence with Augmented Data Approach

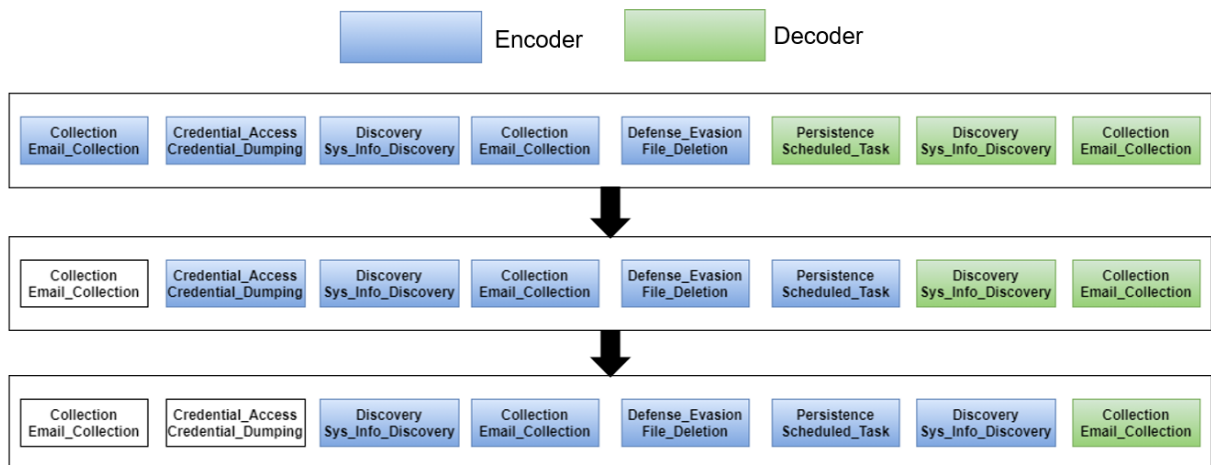


Figure 9: Sliding Window Technique

In this experiment, we take a slightly different approach with the aim of getting more data. A sliding window technique is used that shifts the encoder input by one position to the right. Rest of the input sequence is fed as decoder input. A representation of this technique is shown in figure 9. This sliding window technique is used for the augmentation of data and helps the model to better capture the relationship between different sequences.

This technique resulted in an increased number of samples which is required for getting good performance with deep learning models. Using stacked encoder-decoder can better capture the inter-relationship between sequences. Figure 10 shows an overview of the stacked architecture. In this case, the output of one stack of the encoder-decoder model was introduced as the input of the next stack. However, the context vector obtained at each step was propagated accordingly and thus a more complex model was obtained that helped in capturing the relationship between the attack steps in the sequences more precisely.

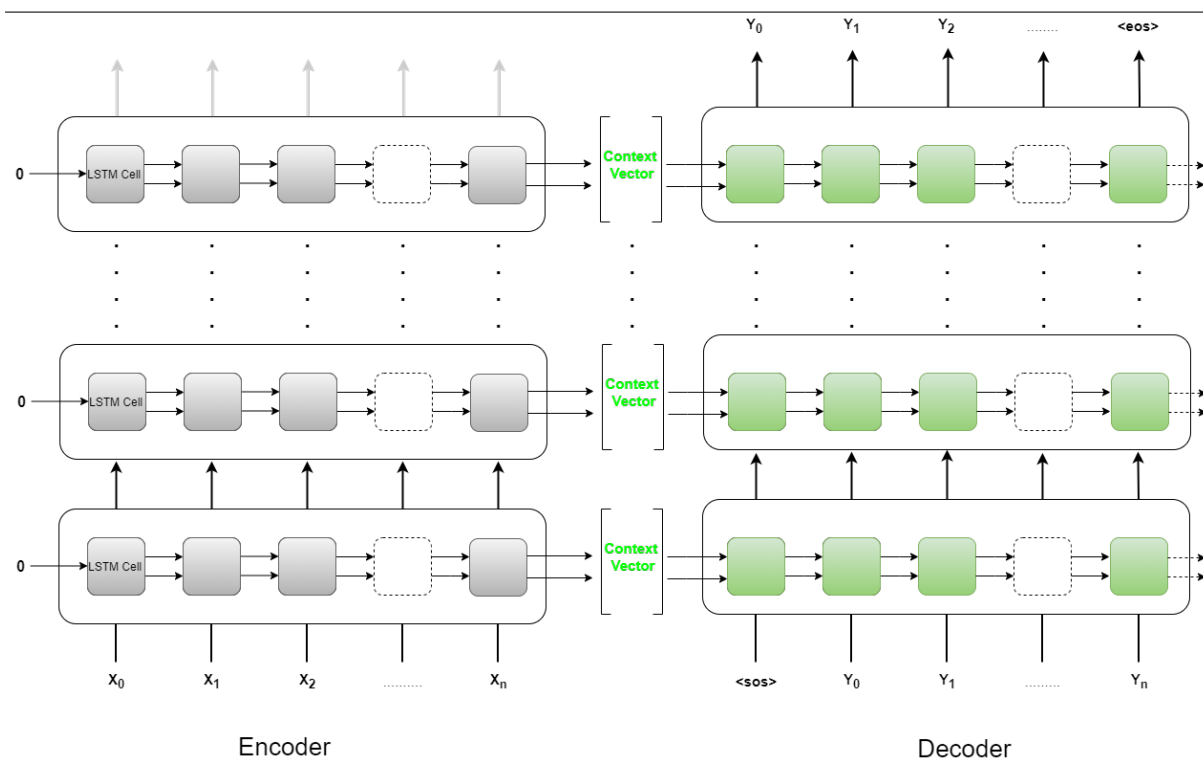


Figure 10: Stacked Encoder-Decoder Architecture

7 Experimental Results

Comparing the three approaches taken, it can be observed that the Sequence-to-Sequence with augmented data can predict the next attack step with a higher accuracy than the other approaches. As table 1 shows this model can predict next steps of an attacker with a success rate of 86.83%. The predicted attack step contains the tactic and technique the adversary is going to deploy next in the attack scenario. With this accuracy, we believe our system can be deployed in real-world scenario for early-blocking an attack and proactive resource allocation.

Approach	Accuracy
Sequence-to-Vector	~20%
Sequence-to-Sequence	~70%
Sequence-to-Sequence with Augmented Data	86.83%

Table 1: Accuracy comparison of different approach

8 Conclusion and Future Works

Unlike the different security event prediction systems and other IDS based systems, we do not rely on the IDS generated alerts to identify the attack events and rather gain an overall idea about the techniques that the attacker is going to execute in the next step of the attack. The successful prediction of this future step can help the organizations to stop the attacks and thus minimize the losses incurred from these targeted attacks. A proper allocation of resources to increase the firewalls at certain steps, for example, can help obtain a better protection against these attacks. Thus the prediction of the attack sequence can be used to thwart the adversaries and prevent both monetary losses and protect confidential information from falling in the hands of the adversaries.

Our prediction model can successfully predict the future attack steps of a targeted attack sequence with an accuracy of 86.83%. With a greater amount of data, this prediction accuracy can be further increased and thus the tactics and techniques of the attacker can be predicted with more precision.

The prediction performed here can provide us with the attack steps that the adversary is going to perform next. But in order to prevent these attacks, the necessary steps of putting up firewalls and blocking different access cannot yet be performed automatically. However, using the log file activities generated during these attack steps, we can automate the process of thwarting these cyber attacks. So, our future research interests can be enumerated as below:

1. To automate the full process of attack sequence generation. If the process can be automated, it will be possible to detect and predict the future steps. This will enable the defender to rapidly respond to the attacker increasing the chances of thwarting the attack.
2. To build a system for proactive resource allocation based on predicted steps. Even after automating the attack sequence generation, we would still require human intervention unless we can automate the resource allocation based on the predicted steps. So, it is one of our future goal to automate the resource allocation so that the whole process can be automated resulting in faster and error free response.

References

- [1] D. S. Fava, S. R. Byers, and S. J. Yang, "Projecting cyberattacks through variable-length markov models," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 359–369, 2008.
- [2] A. A. Ramaki, M. Khosravi-Farmad, and A. G. Bafghi, "Real time alert correlation and prediction using bayesian networks," in *2015 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC)*, pp. 98–103, IEEE, 2015.
- [3] I. Perry, L. Li, C. Sweet, S.-H. Su, F.-Y. Cheng, S. J. Yang, and A. Okutan, "Differentiating and predicting cyberattack behaviors using lstm," in *2018 IEEE Conference on Dependable and Secure Computing (DSC)*, pp. 1–8, IEEE, 2018.
- [4] Y. Shen, E. Mariconti, P. A. Vervier, and G. Stringhini, "Tiresias: Predicting security events through deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 592–605, 2018.
- [5] Y. Takahashi, S. Shima, R. Tanabe, and K. Yoshioka, "Aptgen: An approach towards generating practical dataset labelled with targeted attack sequences," in *13th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET} 20)*, 2020.
- [6] "Mitre att&ck framework." <https://attack.mitre.org/>.
- [7] T. Micro, "Targeted attack." <https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/understanding-targeted-attacks-what-is-a-targeted-attack>.
- [8] Wikipedia, "Ashley madison hack in 2015." https://en.wikipedia.org/wiki/Ashley_Madison_data_breach.
- [9] Netsurion, "Cyber kill chain model." <https://www.netsurion.com/articles/eventtracker-enterprise-and-the-cyber-kill-chain>.
- [10] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
- [11] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

-
- [12] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [13] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in neural information processing systems*, pp. 3104–3112, 2014.