ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

Modeling Correlation Based Web Attack Detection Using Deep Learning Approach

Yitayal Kassie Mikru

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

Addis abeba, Ethiopia

October 2020

ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Yitayal Kassie Mikru

Advisor: Mulugeta Libsie (PhD)

This is to certify that the thesis prepared by Yitayal Kassie, titled: *Modeling Correlation Based Web Attack Detection Using Deep Learning Approach* and submitted in partial fulfilment of the requirements for the he Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Name and Signature of members of the Examining Board:

Name	**Signature**	**Date**
1. Advisor: Mulugeta Libsie (PhD)	_____	_____
2. Examiner: Dida Midekso (PhD)	_____	_____
3. Examiner: Ayalew Belay (PhD)	_____	_____

Abstract

World Wide Web is a vastly resource full of knowledge, entertainment and cultural exchange of the planet. In today's world, more than half of world's population use Internet for their day to day activities, their life much more depends on Internet services. Due to this, the Web is the number one targeted versatile attack. To detect the attack many works have been done, but events (HTTP request packet payload) correlation based web attack detection has got less attention, as payload is the key attackers used for attacking application layer.

In order to solve such a problem, we have proposed and implemented an event correlation based web attack detection using deep learning approaches. The aim of our proposed system is based on the correlation of events increasing the detection capability for the current sophisticated web attacks. To do this, our proposed system has integrated components such as convolutional neural network, and bidirectional long short-term memory recurrent neural network are the hearts of our proposed system. Convolutional neural network extracts high level features by correlating low level feature of events, then passes the sequence of extracted high level features to the bidirectional long short-term memory recurrent neural network. It learns the sequence of features by considering the past and the future of the events information and classify the incoming events as attack or benign. This approach helps to minimize false positives and false negatives, make the system adaptive to changes, detecting new attacks and reducing operational costs.

The proposed system is implemented using the CSIC 2010 HTTP dataset which contains the attack and normal raw HTTP request packet. We extract the payload of the request packet and using it the model is trained and tested. We split 65% of the data for training, 20% of the data for testing and 15% of the data for validation. The common performance evaluation techniques accuracy, precision, recall and f1-score were used to measure the effectiveness of the proposed system. The study showed that the result of an event correlation based web attack detection using the combination of convolutional neural network and bidirectional long short-term memory recurrent neural network achieved 98.6% accuracy, 99.2% precision, 97.3% recall, and 98.2% f1-score.

Keywords: Web, Web attack, Payload, Event, Correlation, Deep learning, Convolutional neural network, Bidirectional long short-term memory, Word embedding.

Dedication

To my grandmother: Enatneshi Ayicheh

To my grandfather: Aderaw Belay

To my mom: Mazengia Aderaw

Acknowledgements

First of all, I would like to thank God and Saint Virgin Mary for their valuable support to begin and to bring to an end this study and being with me in all directions of my life.

I would like to thank my advisor, Dr. Mulugeta Libsie, for accepting this thesis work and dedicating his time, patience and also providing comments all the way throughout the various stages of this study.

Lastly, I would like to thank my family for all your supports and helps and it is also my pleasure to express my deep gratitude to my pals Mesele, Mulat, Tiliksew and all my classmates who have helped me in so many ways.

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Networks
B2C	Business to-Consumer
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Conventional Neural Network
CERN	European Organization for Nuclear Research
DAG	Directed Acyclic Graph
DL	Deep Learning
GTI	Global Threat Intelligence
HTTP	Hyper Text Transfer Protocol
HTML	Hypertext Markup Language
IDS	Intrusion Detection System
LSTM	Long Short-Term Memory
ISTR	Symantec Internet Security Threat Report
ML	Machine Learning
MLP	Multilayer Perceptron
OS	Operating System
OWL	Web Ontology Language
RNN	Recurrent Neural Networks
SVM	Support Vector Machine
URL	Uniform Resource Location
W3C	World Wide Web Consortium
WWW	World Wide Web
XSS	Cross-Site Scripting

Chapter One: Introduction

1.1 Background

Early websites were repositories of static documents. People access and view these documents through browsers. The information flow is unidirectional from servers to clients and users do not need to submit any sensitive data. Today's Web has become a dynamic, complicated and powerful application. It provides services such as shopping, social networking, entertainments, blogging, and much more for people and has permeated almost every field of our life. Owing to the Web, people are enjoying unprecedented convenience and happiness, however, at the same time their confidential data like bank account numbers and national identifiers are inevitably exposed on the Web [1]. The data processed on the Web are becoming more and more sensitive and valuable, resulting in that attackers are increasingly focusing on and succeeding with Web based attacks. Hence, protecting web from intrusions is vital [2]**.** There are many reports which show the increasing of web threats from day to day [3].

Symantec Internet Security Threat Report (ISTR) [4] shows that the number of web attacks explosively increased, reaching more than 1 billion web requests analyzed each day up 5% from 2016, 1 in 13 web requests lead to malware up 3% from 2016. In an other report, McAfee Global Threat Intelligence (McAfee GTI) [5] says protections against malicious URLs reported 365,000 (0.5%) of them are risky, out of 73 million tested URLs. Also, Web application assessment includes evaluation of the security level on a scale from "extremely poor" to "acceptable." "Extremely poor" means presence of numerous critical vulnerabilities that, for example, allow execution of OS server commands by any external attacker or disclosure of sensitive information. Depending on the number of critical vulnerabilities and complexity of vulnerability exploitation, the security level may vary from "extremely poor" to "below average." In 2017, almost half of tested applications (48%) were evaluated as "average." Almost half of web applications (48%) were within the range from "below average" to "extremely poor" [6].

From those attacks defined in [7], the top ranked web attacks like, Cross-Site Scripting (XSS), SQL injection, file disclosure, remote file inclusion, code injection attack is the most common. So, to protect the web from those attacks different techniques are used, like event correlation.

which is a technique that relates various events to identifiable patterns [8].

Deep learning algorithms, extract high-level semantic features by calculating the correlations among adjacent regions [9]. The goal of this research is thus to model and implement an events (HTTP request packet payloads) correlation based web attack detection system. It deals with the correlation of events (HTTP request packet payloads) using a deep learning approach. The main significance of this research is to detect new attacks, to minimize false positives and false negatives, reduce operational cost, and make the system adaptive to any changes.

1.2 Motivation

The work in [10] presents the status of Internet users and world population. It shows as of June 30, 2018 4,208,571,287 people use Internet from 7,634,758,428 total number of world population. This means, more than half of world's population use Internet for their day to day activities. Consequently, the work in [11] reports in particular, the 2018 survey showed a dramatic rise since 2016 in monthly assessments for Webs, the top threat experienced by chief information security officers (CISOs). Perhaps most encouragingly, cyber-security is being elevated to state leadership as a key issue on a regular basis. Monthly reporting to business stakeholders has also increased to 25 percent, up from 10 percent in 2016. To solve the problems, different approaches have been suggested [12, 13, 14, 15, 16], but still the problems is not solved.

Therefore, the main motivation of this work is to increase the performance of minimizing false positives and negatives, new attack detections, reducing operational costs and making of adaptive to changes.

1.3 Statement of the Problem

Nowadays, web is not only the largest pool of knowledge, entertainment, and cultural exchange of the planet, but also one of the most versatile attack vectors [17]. The approaches used for web attack detection influences the effectiveness of attack detection. There are a number of web attack detection approaches proposed, but a few of them were considered to detect web attacks using events (HTTP request packet payload) correlation as payload-based attack detection is an effective approach for detecting application layer attacks [15, 16].

Xiaohui *et al* [18] proposed a convolutional neural network and long short term memory recurrent neural network based events correlation approaches. The convolutional neural network (CNN) has an ability to extract high-level features using spatial correlation of events. It helps to compress a wide-field view of the events into high-level feature and feed the extracted features to the LSTM. Whereas, long short-term memory (LSTM) recurrent neural network has a capability to learn sequences of extracted events information. It is important information for the accurate detection of malicious activity as attacks take steps to reach their goal. The LSTM learns the sequence only from the past information in unidirectional way. This led the system to know less about the context and it has been proved that the bidirectional network are substantially better than unidirectional one [19]. Therefore, by making the LSTM capable to learn from the past and the future information in bidirectional way. It needs further works, this helps to increase the performance of new attack detection, minimizing false positives and false negatives, adaptability, and reducing operational costs.

Generally, lots of previous works have been done on web attack detection but less attention is given for events (HTTP request packet payload) correlation based attack detection. The aim of this research is filling the gap of the above mentioned problems by modeling the combination of the two types of deep learning building blocks, convolutional neural network (CNN) and bidirectional long short-term memory recurrent neural networks (BiLSTM-RNN).

1.4 Objectives

General Objective

The general objective of this research is to model a correlation based web attack detection using deep learning approach.

Specific Objectives

To achieve the general objective, the following specific objectives are formulated:

- Study the works and the current practices that have been done in the area of web attacks and their detections.

- Developing an architecture for the proposed web attack detection approach.
- Collect data for training and testing of the proposed model.
- Develop a simulated prototype for proposed web attack detection model.
- Train the proposed web attack detection model.
- Test and evaluate the effectiveness of the proposed web attack model.

1.5 Methods

In order to accomplish the objectives of the research, the following methods and procedures will be used:

Literature Review

Extensive literature review will be made on different related works to gain detail understanding of the state of the art in the area and to identify the actual problem, and to find useful approaches that can efficiently solve the problem.

Data collection

Collecting data from CSIC 2010 HTTP dataset that will be used for training and testing of the proposed model.

Prototype Implementation

The implementation of the proposed model will be done using different tools and data.

Testing and Evaluation

Results of the proposed approach will be tested using the testing data and evaluated in terms of accuracy, precision, recall and f1-score performance measurement metrics.

1.6 Scope and Limitations

The scope of this research is modeling and implementation of events (HTTP request packet payload) correlation based web attack detection using deep learning approach. The system is modeled to increase the detection of new attacks, to make the system adaptable, to minimize false positives and false negatives, and to reduce operational cost. However, the system model only focuses on detection of web server attacks using the event (payload features of HTTP request packet) or more specifically query of users. It doesn't consider detect client side

attacks.

1.7 Application of Results

There are several technologies and methods to keep the webs from malicious activities. A deep learning based approach is one of these methods which protect a webs from a web based attacks by analyzing all incoming HTTP request packets and searches for any suspicious patterns and then when an attacks are discovered it notifies (triggers) the administrator or excluding the HTTP request packets from accessing the web services. Therefore, the result of this study help organizations or individuals to protect their web from attacks.

1.8 Organization of the Rest of the Thesis

The remaining part of the thesis is organized as follows. Chapter Two presents literature review. Chapter Three presents the existing works that are related to this thesis. Chapter Four deals with the design of proposed web attack detection system. Chapter Five presents an implementation and discussion of the experimental results and its evaluation. Finally, in Chapter Six conclusions made on the thesis result, contributions of this thesis and recommendations on possible future works are presented.

Chapter Two: Literature Review

In this chapter, a number of literatures have been reviewed to provide background information about web, evolutions of web, web attacks and their detection trends. About event correlation and its techniques will also be discussed.

2.1 The World Wide Web

The Web is the largest transformable-information which is a system of interlinked hypertext documents accessed via the Internet [20, 21]. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks. On March 12, 1989, Tim Berners- Lee, then a CERN (European Organization for Nuclear Research) employee, wrote a proposal for what would eventually become the World Wide Web [22]. The 1989 proposal was meant for a more effective CERN communication system but Berners-Lee eventually realized the concept could be implemented throughout the world. Berners-Lee and Robert Cailliau proposed in 1990 to use hypertext "to link and access information of various kinds as a web of nodes in which the user can browse at will [23]. However, gradually accessing information also has changed, and more people are relying on the Web as a primary source of information. This information can be obtained from different places such as websites, blogs, online publications, social networks, databases and much more [24]. To make the web like this it passed different evolutions.

Initially, use of Web 1.0, the basic Internet Web, was associated with major companies. Its use was limited to publishing corporate information, developing marketing and sales plans and transactions with customers. This Web ushered in the first online strategy for businesses [25]. Then, Web 2.0, the social Web, a platform for collaboration, offered users a new version of WWW, not so much in terms of updating the Web's technical specifications, but rather in terms of the changes software developers and end users made to the way it was used. Web 2.0 is qualitatively different from previous Web technologies as it has Web applications that facilitate information sharing, interoperability, user-centered design and collaboration in the WWW. Examples of Web 2.0 are Web Communities Web services, Web applications, social network services, video hosting services, wikis, blogs, mashups and folksonomies, among others [26]. The late 1990s saw a change in the role of Internet users as they began to create

content and social value. The symbols of this era include YouTube, Facebook, Linkedln, Wikipedia, among others. Internet then became a cooperative platform in which collective power and networking effects opened up the possibility of generating extraordinary value. These social changes in turn caused changes in business models that attempted to make the most of each individual contribution and prepared to coexist in a definitively virtual environment. Although it remains a challenge, or is still unfinished, the Web 2.0 era is giving way to Web 3.0 or what is known as the semantic Web [27]. Web 3.0 combines human and artificial intelligence to provide more relevant, opportune and accessible information. Web 3.0 has a more powerful language derived from neuronal networks and genetic algorithms, with a particular emphasis on analysis, processing capacity and how to generate new ideas based on user-generated information. Web 3.0 is a neologism used to describe the transformation of the Web into a database, a way of making content more accessible through multiple non-browser applications, artificial intelligence technologies, the semantic Web, the geospatial Web and the 3D Web. The market often uses it to promote improvements in relation to Web 2.0 [28]. The fourth step in the evolutionary process is occupied by Web 4.0 based on wireless communication (mobile devices or computers) connecting people and objects whenever and wherever in the physical or virtual world in real time. For example, GPS that guides cars and now helps drivers to improve the planned route or save fuel will shortly save them from having to handle it [29].

Basically, the Web architecture is the conceptual structure of the World Wide Web. The World Wide Web is a constantly changing medium that enables communication between different users and the technical interaction (interoperability) between different systems and subsystems [30]. The architecture of a Web system affects many of its quality attributes, such as testability, security, and accessibility. The architecture itself is influenced by factors such as system requirements, infrastructure constraints, and interoperability needs [31]. Also Web architecture consists of the requirements, constraints, principles, and design choices that influence the design of the system and the behavior of agents within the system. When followed, the large-scale effect is that of a shared information space [32]. In a nutshell we can say the architecture of a Web system is a building block of Web Browser (Users computer) and Web server that communicates with each other, web browser request for web page to the server and web server responds the requests accordingly [33, 34].

A Web system incorporates a web browser that displays information content by bringing information resources to the user and also it is an application retrieving, presenting and traversing information resources. Whereas web server transfers information to the client. Which means, a computer program that accepts HTTP requests and return HTTP responses with optional data content. Also in between web browser and web server (Client and server) there is a communication medium which helps to define the path by which client and server communicate with each other [35]. The Web system contains different elements that exist in the world wide concept. The client is the user's interface to the Internet. Whatever type of service is requested this interface stays the same. So users do not need to understand the differences between the many different access schemes in common use on the Internet. The user initiates a request by specifying a Uniform Resource Identifier or a "hyperlink". This link can specify any accessible information or resource on the Internet as long as it can be uniquely identified as an object. The word "Web" refers to the combination of accessible objects and the links pointing to them throughout the Internet. The server is responsible for handling the request sent from the client. This can either be a local accessible resource or the server can request the resource from another server in which case the first server temporarily turns into a client. The client sends of the user request to a World Wide Web server using the Hypertext Transfer Protocol (HTTP). This is a typical client-server application based on a stateless connection between the client requesting the URI and the server handling the request. On a successful request, a data object is returned from the server to the client. The object is written in the Hypertext Markup Language (HTML) which is a hypertext language with the possibility of containing hyperlinks that the user can follow [36, 37].

Today, it has been estimated that there are over one billion websites on the world wide web [38], and this number is steadily increasing over time. In 2015, the global business to-consumer (B2C) e-commerce turnover has increased by about 20 percent, attaining a value of 2.2 billion dollars [39] and even governments are increasingly transitioning to web services to enable savings on budget [40]. Unfortunately, this popularity regularly attracts a large number of attackers and according to Symantec three quarters of the websites they scanned in 2015 contained unpatched vulnerabilities [41].

Generally, providing effective cyber-security solutions for web applications is very challenging. This happens due to the fact that the commonly used IDS (Intrusion Detection

System) and IPS (Intrusion Prevention System) systems have problems in recognizing new attacks (0-day exploits), since these systems are based on the signature-based approach. In such a mode, when the system does not have an attack signature in its database, such attack is not detected. Therefore, there is a need to develop more sophisticated methods that are both capable of adapting domain expert knowledge [42, 43], as well as emerging cyber security solutions (e.g., event correlation, data mining [44]), and deep learning is capable of automatically finding correlation in the data, so it is a promising method for the next generation of intrusion detection [45].

Event correlation monitors the various security events to determine which events are significant and relate to a particular attack [46]. So, to analyze vast amount of requests within a sophisticated way, we use correlation of events (HTTP request packet payload) using deep learning approaches for web attack detection.

2.2 Event Correlation System

Anything which happened at some moment in time which is defined as an event could be an action or occurrence identified by a program, such as pressing a key or clicking a mouse button. In computing environment, the term event is also used for that message which conveys what has happened and when it has happened [47], or more specifically an event is the notification that a happening of interest has occurred [48]. However, in our context an example of events is web requests packet payload. It means that HTTP request packet payloads that are requested by clients to the web server is what we call an event. In all our thesis when we say event we are referring to the HTTP request packet payloads of clients to the web server. The payloads are textual data and represent the content of the communications, such as ‘‘op=system info’’, ‘‘mod=logging’’, ‘‘article=270515’’etc. It contains specific combination patterns; certain combination patterns are associated with normal communications, while other combination patterns are associated with anomalous communications [15].

The process of analyzing events to infer a new event from a set of related events is defined as event correlation [48].

In another way of expression correlation is a process of finding relationships among events in order to reconstruct attack scenario from the isolated events. Thus, it gives high level view of

actual attacks. It deals with a number of independent event sources simultaneously [49]. In addition, it is defined as establishing or finding relationship between entities, is a recognized technique in security to improve the effectiveness of threat identification and analysis process by combining information from multiple sources [50]. By looking at collective information on a set of events rather than the individual ones, one can identify more types of attacks with fewer false positives and false negative. In big data era, the single detection techniques have already not met the demand of complex attacks, it deals with massive data transmitted in Internet in time, get rid of missing information for low speed, cause false negatives and false positives, and minimize the losses brought by the intrusion [51]. Due to this, the practice of event correlation is useful and necessary not only to reduce the number of events but also to do some processing of the likely causes to take some of the workload of the security administrators [52]. To show event correlation more clearly by example, we use Figure 2.1 [53].

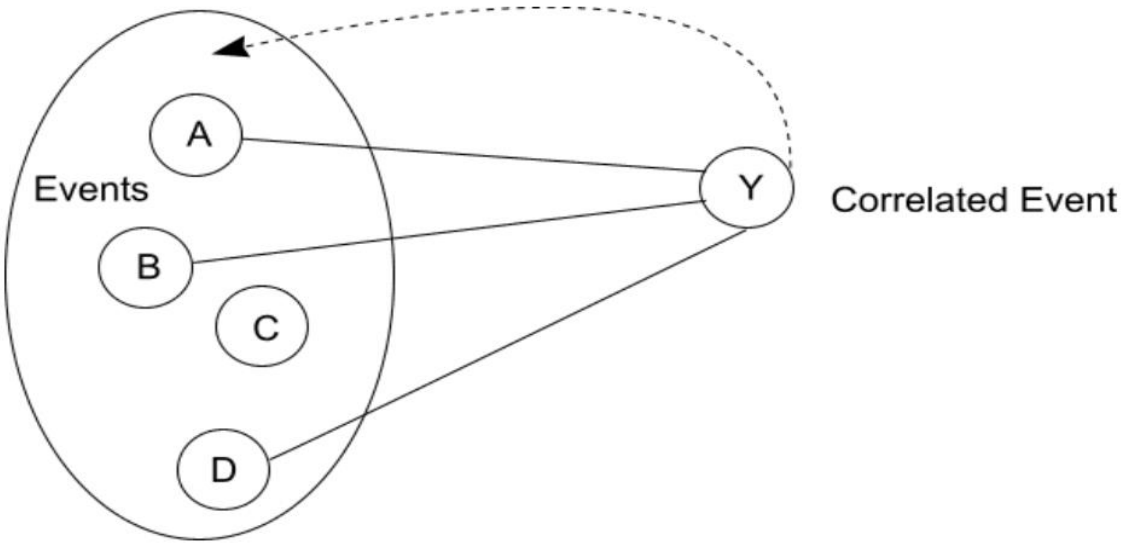

Figure 2. 1: Event correlation

Here in Figure 2.1, as we see different individual events are correlated and showed as a single view (high-level view) and it helps to find pattern from many individual events.

Generally, the chief goal of the correlation process is to identify more significant events in a potentially colossal set of recorded events [54]. If the context of the other events is not considered during analysis, the single events that recorded the malicious action might look benign [55].

2.3 Event Correlation Algorithms

Event correlation algorithms can be divided into three categories: 1) Similarity-based, 2) Knowledge-based, and 3) Statistical-based [56, 57, 58].

2.3.1 Similarity-Based Algorithms

This category of algorithms defines factors to compare the similarity of events. The most important advantage of these algorithms is that there is no need for precise definition of attack types. The following are three main subcategories for these types of algorithms.

a. Simple Rules

The functionality of this idea is based on defining very simple rules to express relations among alert features. In this subcategory, algorithms try to define simple rules in order to compute similarity between attributes of alerts and find the relation.

The major required knowledge for this style of correlation rules are rule structures and required functions for checking similarity [59, 60, 61, 62]. These algorithms can be used in different hierarchical levels and alerts are correlated from various aspects. In the detecting attack sequence capability, these algorithms include limits for defining attack types and can only detect sequences specified based on attack class. If the pattern definition is in a form that partitions conditions of the domain of that alert which alerts can be combined together into separate sets, it can also allocate input data to parallel processors for each pattern based on these conditions. Thus, these algorithms have a very good parallelism capability. These algorithms require maintaining all generated meta-alerts in the current time window for each pattern. Thus, its required memory is linearly proportional to alert input rate multiplied by the time window length.

b. Hierarchical Rules

This subcategory includes algorithms which have formed abstraction levels hierarchically and it makes decisions about security event detections based on these abstraction levels. This algorithm introduces researches that express similarity factors in a hierarchy of concept generalization.

The methods presented in [63, 64] are examples of such algorithm which is designed to detect root cause in network. These algorithms include a method for comparing alert values, with a linear calculation degree proportional to generalization hierarchy tree depths. The required memory for these algorithms is also linear and equivalent to generalization tree sizes. Previous knowledge requirement level in these algorithms is up to defining generalization trees and thus precise and deep network structure and elements knowledge is not necessary, except in case of needing definition for address values and attack classes hierarchy.

c. Artificial Intelligence

This subcategory algorithm in which comparison factors are generated automatically. The event correlation methods using artificial intelligence approaches based on machine learning [65], such as Bayesian networks [66, 67], artificial neural networks [68, 67, 69]. The advantage of this approach is the possibility of independent (unconditional) event correlation minimizing manual configurations.

Artificial intelligence (AI) has existed over many decades, and the field is wide. AI can be viewed as a set that contains machine learning (ML). The ML is a subset of AI, and deep learning (DL), in turn, a subset of ML. The term deep learning which refers to artificial neural networks (ANN) with complex multilayers [70].

The Bayesian networks is a probabilistic graphical model that represents a set of variables and their conditional dependencies via a Directed Acyclic Graph (DAG). The network components and the vulnerabilities are modeled as nodes while the edges represent how they are related to each other. Each node in the graph defines a causal relationship of itself with its parents. The step by step attack scenario shows how vulnerabilities have a causal relation among themselves [71].

Learning a Bayesian network from data consists of the induction of its two different components: (i) the graphical structure of the Bayesian network, that is, the dependence relations between variables, and (ii) the parameters of the network, that is, the strength of these relationships, which are encoded in conditional probability tables associated with each variable. The conditional probability table of a variable contains probabilities of the variable being in a specific state given the states of its parents [72].

In the context of alert correlation, an alert Bayesian network describes the prepare for relations between alert types (attack steps). Children alerts can be viewed as a direct cause of the parent alerts, i.e., consequent attack step. The occurrence probability of the children alerts can be estimated given the state of their parents. So, Bayesian networks can be used as a powerful tool to model attack transition patterns and predict possible upcoming attacks [72].

Artificial neural network (ANN) is an information manager model that is similar to biological nervous systems function of the man brain [73]. We are constantly analyzing the world around us. Without conscious effort, we make predictions about everything we see, and act upon them. When we see something, we label every object based on what we have learned in the past. Neural networks work with the same concept by adjusting the connection exist between neurons. It is composed of a large number of highly interconnected processing elements called neurons, which convert an input vector into some output.

ANN are applicable for many area of problems such as for detection in medical diagnosis, security, image objects, financial irregularity etc., are being enhanced through ANNs application. More specifically it automatically analyzes security events related to true alerts for detecting cyber-threats and execute multiple analysis

The most widely used deep neural network are convolutional model and recurrent model [74].

Convolutional neural network **(**CNN**)** is one of the most used deep learning models, which has shown exemplary performance on several competitions related to computer vision, video processing, natural language processing and cyber security. The powerful learning ability of CNN is primarily due to the use of multiple feature extraction stages that can automatically learn representations from the data. Remarkably, the ideas of exploiting spatial information and multi-path information processing have gained substantial attention [75].

Recurrent neural networks (RNNs) is widely adopted in research areas concerned with sequential data, such as text, audio, and video. However, RNNs consisting of sigma cells or tanh cells are unable to learn the relevant information of input data when the input gap is large. By introducing gate functions into the cell structure, the long short-term memory (LSTM) improved the remembering capacity of the standard recurrent cell [76]. The LSTM learns the sequence information by considering only past information. To solve this problem, a bidirectional long short-term memory recurrent neural network (BiLSTM) is proposed which

considers to learn sequence from past and future information. It has been proved that the bidirectional networks are substantially better than unidirectional ones in many fields [77, 78].

The detail of artificial intelligence components used in our proposed model such as Word embedding, CNN, BiLSTM, Dense Layer and Softmax Classifier are described below in details.

Word Embedding

Word embedding is a representation of feature vectors in a coordinated way by capturing the relationship between tokens. The tokens of payloads are represented based on their relational context and semantics (meaning) with other tokens. The embedding layer receives input, only tokens that are represented by integers, which are encoded using Algorithm 4.1. After receiving them in a sequential continuous feature vector space within a specified embedding dimension, which are the size of vocabulary (input_dim), size of the vector space (output_dim), and the length of input sequences (input_length).

The embedding layer is initialized with random weights and then learns the embedding of all the tokens while training of the model with the training dataset. In a nutshell, word embedding is a technique where words are encoded as a real-valued vector representation in high dimensional space, where the similarity between words in terms of semantic translates to closeness in the vector space [79]. Therefore, the semantic relatedness and correlations between words can be directly calculated in the word embedding space [80].

Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a powerful artificial neural network technique. It has different layers like a convolutional layer and a pooling layer [81]. CNN extracts high-level semantic features by calculating the correlation between a neighborhood of input features [15, 82]. Furthermore, CNN has the capability of capturing correlations of special structures between different input data [83, 9, 75]. To perform this, CNN applies a series of convolution and pooling operations on the tokens of payload data. Convolution is one of the main building blocks of a CNN performed on the input data with the use of a filter or kernel to produce feature maps. Whereas, pooling continuously reduces the dimensionality, the number of parameters, and computation in the network by taking the maximum value in each

window to decrease the feature maps size [84].

The convolutional neural network receives input from the embedding layer output. The convolutional layer performs convolution operations by calculating the correlation between adjacent regions of embedded tokens feature vectors. Let $Y = (y_1, y_2, y_3, \dots, y_L)$ denote an embedded sequence feature matrix, over this matrix a set of n kernels or filters $F = \{f_i \in \mathbb{R}^{dxh}\}_{i=1}^{n}$ where d- dimensional tokens feature vector, with a window size of h tokens to produce a new feature map or (to extract high-level features) is performed in the convolutional layer. For each filter or kernel f_i, the produced feature map s^i is generated from a window of sequence tokens for each position t is denoted as:

$$s_t^i = \emptyset(f_i^T Y_{t:t+h-1} + b) \quad (1)$$

Where: $s^i \in \mathbb{R}^{1xL-h+1}$, b is the bias and $\emptyset$ is a nonlinear rectify transformation function In our case, we choose Rectified Linear Unit (ReLU) on each convolution operation to introduce nonlinearity relationships into the model. For all n filters or kernels in F, there is a total n sequence $S = \{s^i\}_{i=1}^{n}$. Therefore, the produced convolutional features map S is a new sequence feature with width n and length $L - h + 1$.

After performed the convolution operations then we apply a max-pooling operation over the sequence of mapped features, which picks the maximum values using Equation 2.

$$p^i = \max_{t\in\{1,\dots,L-h+1\}} \{s_t^i\} \quad (2)$$

The max pooling operations are applied to each s^i mapped features**.** Finally, the new sequence of mapped feature vectors is defined as**:**

$$P = (p_1, p_2, p_3, \dots, p_L) \quad (3)$$

The operation is applied to the length dimension, so the result of max-pooling is a sequence of size n. Capturing the most important representative features from the mapped features is the idea of the max-pooling operation.

Bidirectional Long Short-Term Memory (BiLSTM) Recurrent Neural **Network**

Long Short-Term Memory network is a special kind of RNN (Recurrent Neural Network) capable of learning long-term dependencies. It is explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn [85]. LSTM consists of input gate, output gate, forget gate and candidate memory cell. It avoid the long-term dependence

problem with the memory cell [86]. Also, long short-term memory finds specific sequences by computing the correlation between the current state and the previous state [15]. But these LSTM predict the current status based only on former information or previous information. It is clear that some important information may not be captured properly by the cell if it runs in only one direction [87]. To improve this challenge is using BiLSTM which processes sequence data in both forward and backward directions with two separate hidden layers. BiLSTM join these hidden layers with the same output layer. In other words, the BiLSTM that the current output is not only related to the previous information but it also related to future information. It helps to understand the context better.

The pooling layer passes the maximum feature values representation P to BiLSTM. The BiLSTM learns the sequential dependency relationship between features which is able to correlate previous and future information to present task and remember information for a long period of time. It learns the feature dependency using the gap between two-time steps. At each time step the output of the module is controlled by a set of gates in $\mathbb{R}^d$ as a function of the old hidden state h_{t-1} and the input at the current time step P_t the forget gate f_t the input gate i_t and the output gate o_t. These gates collectively decide how to update the current memory cell state c_t and output of the current hidden state h_t, at each time step t.

The forward LSTM transition functions are defined as follows:

$$\mathrm{i_t} = \sigma(W_i . [h_{t-1},\ P_t] + b_i) \tag{4}$$

$$f_t = \sigma(W_f . [h_{t-1,} P_t] + b_f) \tag{5}$$

$$o_t = \sigma(W_o . [h_{t-1,} P_t] + b_o) \tag{6}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot t\mathrm{anh}(W_c . [h_{t-1,}\ P_t] + b_c) \tag{7}$$

$$h_t = o_t \odot t\mathrm{anh}(c_t) \tag{8}$$

Where: σ is the logistic sigmoid function, and the $t$anh is the hyperbolic tangent function, W is the weight matrix and b is the bias.

The notations $\cdot$ represent the Matmul product, and $\odot$ denotes the elementwise multiplication. we can view f_t as the function to control to what extent the information from the old memory cell is going to be thrown away, i_t to control how much new information is going to be stored

in the current memory cell, and o_t to control what to output based on the memory cell c_t.

The backward LSTM transition functions are the reverse copy of the forward LSTM. As each BiLSTM layer contain a forward LSTM and a backward LSTM, The output of the two hidden layer is determined by combining together the forward output $\overrightarrow{h_t}$ to backward output $\overleftarrow{h_t}$. BiLSTM generates an output vector, y_t, in which each element is calculated as follows:

$$y_t = \sigma(\overrightarrow{h_t}, \overleftarrow{h_t}) \quad (9)$$

Dense (Fully Connected Layer) and Softmax (Output Layer)

After each gates performs operations in the BiLSTM, the dense layer takes the deep representation of the hidden state and transforms it into the fully connected tokens feature vector N, then transmits to the Softmax classifier.

The Softmax classifier function turns it into a two-dimensional probability distribution M = (m0, m1) as follows:

$$\mathrm{M} = \mathrm{Softmax}(\mathrm{N}) \quad (10)$$

The values are scales in to 0 and 1. The training sample of tokens payload feature are classified using as follows:

$$\mathrm{Classification} = \begin{cases} 0, & if\ m0 < m1 \\ 1, & else \end{cases} \quad (11)$$

The classification 0 means the classifier evaluates sample tokens payload feature as benign, while the classification 1 means the classifier evaluates sample tokens payload feature as an attack.

2.3.2 Knowledge-Based Algorithms

This category is based on a knowledge base of attack definitions. Algorithms existing in this category are divided into two main subcategories: 1) Pre-requisites and Consequences and 2) Scenario. The basis of Pre-requisites and Consequences algorithms is on the definition of pre-requisites and possible occurring results. Thus, each incident is chained to other incidents by a network of conjunction and disjunction combinations and generates the possible network of attacks. Hence, this idea is placed in a higher level than correlation based on features

similarities and in a lower level than combining based on pre-defined attack patterns. Although these algorithms do not require precise definition for each attack scenario like scenario-based algorithms, the previous knowledge is necessary for determining pre-requisites and all existing incident results. Scenario algorithms are based on the idea that many intrusions include various steps which run one by one to success the attack. Thus, low level alerts can be compared with pre-defined intrusion steps and correlate a sequence of alerts related to each attack. Thus, a set of different attack scenarios definitions exist in a knowledge base in this type of algorithm. A list of current attack scenarios is maintained when the correlation system is operating, which this list includes all scenarios that at least one step of them are done recently. By the arrival of a new alert, it is compared to the current scenario and if the possibility is more than a certain threshold, it will attach to the scenario. Otherwise, if the alert is compatible with one of the possible scenario definitions inside the knowledge base, a new current scenario is generated using this alert. The main challenge for these algorithms is definition of attack scenarios even with existing automatic attack scenario learning methods. Also, these algorithms are completely deficient against new attack.

a. Prerequisites/Consequences

The algorithms in this subcategory observe and control meanings of alerts and existing concepts in the network and then detect a security event. In addition, makes extracting and forming a relation between different attack stages possible, with the pre assumption of knowing a knowledge base which describes all existing prerequisites/consequences of an alert. In these algorithms, alerts are modeled using first order logic and causal relationships are defined for backgrounds, and consequences of each event. Thus, a graph of possible alerts and relationships between them can be created and provide appropriate tools to reduce the amount of information shown to the user. To continue, some researches expanded the mentioned tools to identify attack scenarios, analyzed the attack procedure [88, 89] and also detected lost components of an attack [90]. In terms of the reliance amount on environmental knowledge, these algorithms have the most requirements and in contrast, generate conclusion outputs without any bias and completely based on real alert meanings. Given that these algorithms do not use any pre-assumed information in addition to default environmental knowledge, they are very flexible and extendable algorithms and the algorithms behavior changes in real time

with any change in the environmental knowledge. In addition, in cases of required processing power, unity, parallelism ability and required previous data, these algorithms are in the heaviest existing algorithms range, because with the arrival of each new alert, any kind of its relationship with all other alerts in the active time window must be checked and this task needs a huge amount of adjustments between alert types, prerequisites/consequences, and their information details like source and destination address.

b. Scenario

The main application of this set of algorithms is detecting multi-step attacks and their reliance is on the existence scenario of these kinds of attacks [91, 92, 93]. Various languages are presented for describing these scenarios but the main idea in all of them is specifying attack steps and prerequisites and its goals. Thus, in terms of required amount of environmental knowledge, this set of algorithms require a higher level of knowledge than prerequisites and results-based algorithms, but this knowledge can have less amount and domain. So, required processing resources in this branch is based on defined rules, can be less than the pre-requisites and results-based algorithms. But due to the very wide range of possible cases, unitizing and paralleling will be difficult. In case of defining a context language for expressing scenarios, these algorithms are completely flexible and extendable, because the system behavior must change in real time according to any change or extension in rules. The required memory for detecting scenarios rises according to the number of defined scenarios and required time window.

2.3.3 Statistical-Based Algorithm

The basic idea of these algorithms is that relevant attacks have similar statistical attributes and a proper categorization can be found by detecting these similarities. These types of algorithms store causal relationships between different events and analyses their occurred frequencies in the system education period using previous data statistical analysis and then attack steps are generated. After learning these relationships and being confirmed by the expert, this knowledge is used for correlating different attack stages. Pure statistical algorithms do not have any prior knowledge on attack scenarios. But scientific results indicate that using these algorithms is possible only in very specific domains in which domain attributes are taken in

account of designing algorithms and otherwise, high error rate exists. This category is also divided into three subcategories: 1) Statistical Traffic Estimation, 2) Causal Relation Estimation, and 3) Reliability Degree Combination. The statistical traffic estimation goal is to detect events which are regularly repeated and finding their repetition pattern. The causal relation estimation is estimating causal relationships between events, predicting next events occurrence, and detecting attacks and the reliability degree combination goal is combining reliability with completely similar events.

a. Statistical Traffic Estimation

In this subcategory patterns of occurred alerts are recognized and the repetition pattern is derived and non-similarity with these patterns will be detected in the future. The goal of algorithms presented in [94] is creating a statistical network traffic model, predicting it, and removing predictable cases. An important category of this kind of alerts contains alerts which occur periodically, due to wrong network or security system adjustments. These algorithms do not need any context knowledge and all of their activities are carried out based on the statistics of each alert. According to this point that each of these filters is defined on a certain alert domain (according to choices made by the system supervisor), parallelism is easily possible in this application and before processing, and the alert processing unit can be easily specified. The processing load of this algorithm completely depends on the used statistical model. The algorithm requirement of dataset is determined based on the model time depth, but due to the use of only statistical information, storing or accessing real alerts is not necessary and only the relatively low and constant memory capacity is necessary. Compatibility with current conditions and flexibility based on new changes, are completely possible.

Also, the algorithms described in [95, 96] are expressed based on Association Rules for detecting alerts which normally occur together. An important application of this method is determining alert priorities based on this that which alerts have occurred together and have these accompaniments occurred on a usual procedure or a new pattern is observed, but also this algorithm can be used for creating related meta-alerts. This algorithm does not need environmental knowledge and knowledge base and makes decision completely based on alert statistics. The algorithm requirement of memory is determined based on activity time window

and windows are defined separately. In each window, statistical information about all alerts is calculated and the resulted statistics are compared to previous ones. Based on the determined domain by the user for applying this algorithm, arrived data from different units can be pre-partitioned and thus alerts related to each unit can be processed independently and only in their own processing unit, and processing in each unit is also very much parallelizable according to the need of counting different alert combinations.

b. Causal Relation Estimation

The purpose of this subcategory is finding event sequence or association dominant patterns and using these patterns for detecting false cases, or proper combinations. Algorithms defined in [97] are more proper for learning attack patterns and some for detecting false alerts or lost ones. These algorithms create a possible model for determining correlation relationships between alerts. Using this model, alerts can be correlated without environmental knowledge, but gaining a precise and reliable model requires a huge amount of previous data about the attacks.

c. Reliability Degree Combination

The goal of this subcategory is introducing an algorithm for combining reliability with completely similar alerts [98]. In this type of algorithm, changing the reliability to alerts is proposed based on equivalent alert repetitions. The goal is changing the importance priority of an alert, based on its approval by other resources. This algorithms require a huge amount of labeled previous data for generating probability models. To simplify by removing all possible processing details and only accept the amount of an alert repetition as a factor independent from alert importance and resource history.

2.4 Model Performance Evaluation Metrics

We evaluate the performance of our proposed system model based on the Precision**,** Recall**,** Accuracy, and F_1-score performance evaluation metrics. The metrics are defined based on four related parameters (confusion matrix) true positive (TP), true negative (TN), false positive (FP), false nsegative (FN).

A confusion matrix is a table that is often used to describe the performance of a classification

model on a set of test dataset.

True positives (TP): is when the predicted value is positive and the actual value is also positive.

True negatives (TN): is when the predicted value is negative and the actual value is also negative

False positives (FP): is when the predicted value is positive and the actual value is negative.

False negatives (FN): is when the predicted value is negative and the actual value is positive

Based on those confusion matrix parameters the performance of the model is evaluated using the following metrics:

Precision: is defined as the ratio of the correct true positives (TP) classifications divided by the total number of positive classifications (TP) and (FP).

$$Pression = \frac{TP}{TP+FP} \tag{12}$$

Detection Rate (DR): or recall is defined as the ratio of the number of correct positives (TP) classifications divided by the total number of true positives (TP) and the false negatives (FN).

$$Detection\ Rate\ (DR) = \frac{TP}{TP+FN} \tag{13}$$

Accuracy: is defined as the ratio of all correctly classified samples divided by the total number of samples.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{14}$$

F1-score: is defined as a measure that combines precision and detection rate is the harmonic mean of precision and detection rate and represent the balance between them.

$$F_{1-score} = 2 \times \frac{\text{Precision} \times \text{DR}}{\text{Precision} + \text{DR}} \tag{15}$$

Chapter Three: Related Work

In this Chapter, we discuss different research works that are particularly related to our works. Hence, the various techniques that are related to Web attack detection are reviewed. These works are categorized based on the correlation approaches they used.

3.1 Similarity-Based

Shuying *et al* [13] proposed that events correlate based on their attributes similarity like source IP, destination IP, source port, destination port, protocol type and time attributes. Comparing the degree of similarity between events, approximate degree of the event can be calculated by using the formula of event similarity function. The proximity of the event is measured by the similarity of the characteristic attributes of the event record. The work move with constant detection, since it cannot detect the new (unknown) attacks, easy to bypass, the detection accuracy depends on the domain experts, not adaptive, high false positives and negatives.

Jianyi *et al* [14] proposed an algorithm of reducing the false positives in IDS (Intrusion Detection System) based on correlation analysis. The algorithm analyzes the distinguishing characteristics of false positives and real alarms, and preliminary screen the false positives then making similarity between attributes of the event and based on their similarity correlate the events for reducing false positives. The proposed work move with constant detection model, since it cannot detect the new (unknown) attacks, easy to bypass, the detection accuracy depends on the domain experts, not adaptive.

Hanli et al [95] proposed an online approach of adaptive correlation of intrusion alerts in two stages. In the first stage the authors used Bayesian network to automatically extract information about the constraints and causal relationships among alerts and then based on the extracted information correlates raw alerts and constructs attack scenarios online. The approach is able to dynamically adjust to the current alert behavior and reflect it in the correlation process. The proposed approach evaluated using DARPA 2000 datasets and live Honeynet data. The approach they used is the dataset generated using the signature based Snort IDS and mainly focuses on network attack detection. Whereas our study is on web attack detection. Also signature based approach they applied is not capable of detecting new attacks

and high false positive and negative. In traditional machine learning to extract the learned features needs human intervention due to this the performance of the model depends on domain experts, it needs high cost for extracting the features, high error rate.

Jin *et al* [99] proposed an adaptive analysis framework for correlating different kinds of cyber security related data, such as malicious URL, network traffic, and alert incidents. The raw alarm incidents will correlated and aggregated from time, causality and hierarchy relations. The framework adaptively adjusts the security policy according to the result of detection and analysis. It helps to improve the pertinence of the analysis and to better discover the potential threats. Also in the big data context, by employing big data techniques to improve storage capacity, accelerate the calculation or carry out correlation analysis in a much longer time window. The proposed framework implemented using different open source technologies. The proposed framework lacks of new attack detection, easy to bypass, high false positive and negative.

Kleber et al [100] proposed an approach which collects security alerts from different sources and normalize them according to standardized structure IDMEF (Intrusion Detection Message Exchange Format). Then, the normalized alerts are grouped into meta-alerts (fusion or clustering), which are later classified using machine learning techniques into attacks or false alarms. The proposed approach validated and reported against the *DARPA* challenge and the scan of the month (SotM) from the honeynet project. They implemented the model using Perl programming language and Weka tool. Also they applied three different classification algorithms: SVMs (Support Vector Machine), Bayesian Networks and Decision Trees. The proposed approach is worked using traditional machine learning and it needs human intervention to extract the features to be learned, due to this the performance of the model depends on the knowledge of domain experts, high false posetives and negatives.

Chenn-Jung *et al* [101] proposed an adaptive alert correlation detection and rule tuning algorithm. It helps to reduce alerts for the same kind of attack and alleviate the loading of network equipment in processing the correlative alert. The algorithm automatically tuned Intrusion Detection System rules generation modules based on fuzzy logic technique block the highly correlated alerts. The proposed algorithm evaluated using snort and the VRT certified rules for Snort were employed as the N-IDS sensor, and the datasets used Defcon9,

MIT-LL1999, MIT-LL2000 and Treasure Hun. The work has no capability to detect new attacks or previously unseen attacks, easy to bypass since the proposed algorithm is rule based due to this, the performance of the algorithm depends on the rule creators.

Neelam et al [54] proposed the concept of event correlation that has been introduced for Intrusion Detection System (IDS). They aim to improve security by correlating of events and reduce the workload of the IDS analyst. The correlation has been achieved by getting together similar alerts, thus allowing the system analyst to only look few alerts instead of hundreds or thousands of them. For experiment, they used the result of SNORT Intrusion Detection System with SEC (Simple Event Correlator) by taking the input from the MIT DARPA dataset. The correlation removes duplication of alerts and thus reduces the information overload on security administrator. The work is not adaptive, not detecting new attacks, the performance of the system depends on the knowledge of security experts or rule creators, high false positives and negatives.

Hongyu et al [15] proposed a convolutional neural network and long short term memory recurrent neural network model based payload classification for web attack detection. The CNN model extracts and learns the low level payload feature by correlating the entire payload data stream. The RNN learns the sequence of payload feature representation by computing the correlation between the current state and the previous state. It helps to extract high level semantics feature of the payloads. Both approaches learn feature representation from original payload without feature engineering. The model performed experiments in each separated model of CNN and RNN (LSTM) on CNTC-2017 Webshell, CSIC-2010 HTTP and DARPA1998-all-attacks datasets. The work of CNN has lacks of learning the sequence or dependences of features. The work of RNN (LSTM) has lacks of high level feature extraction.

Xiaohui et al [16] proposed a payload-based web attack detection using AutoEncoder and RNN approaches in separated form and classified web attacks. It used n-gram algorithm to extract basic feature of the payload. The AutoEncoder learns the representation (encoding) for a set of data in unsupervised learning, it helps to reduce the dimensionality of the data. The RNN helps to connect previous information to present tasks. But lack of memorizing the sequences or dependencies of features information.

Ming *et al* [12] proposed a deep learning method to detect Web attacks by using a specially

designed CNN (Conventional Neural Network). The method is based on analyzing the HTTP request packets to classify a request as normal or malicious. They have done experiment on HTTP DATASET CSIC 2010 to evaluate the effectiveness of the method. The proposed work has lacks of learning sequences or dependencies of features.

Ying *et al* [102] investigates an adaptive system that periodically updates the detection model to detect unknown attacks for Web by obtaining malicious queries. They used hybrid support vector machine which is combination of Suspicion Selection (SS) and Exemplar Selection (ES). SS obtains the most important informative queries and features in improving detection performance while ES specializes in harvesting truly malicious queries. Queries obtained by SVM HYBRID are incrementally incorporated into the training pool to update the detection model for incorporating important queries for indications of malicious or normal behaviors. To test the strategy, trained on a ten-day query dataset collected from an academic institute's web server logs. The proposed work has done based on traditional machine learning algorithms due to this, the algorithm needs domain experts to extract features to be learned and the performance of the algorithm depends the experts. It takes high operational cost to extract the features, high false positives and false negatives.

Meera *et al* [103] proposed forensic analysis of the cloud system with specific emphasis to the analysis of cloud services logs. They mentioned that the major issue with service logs is that there is no segregation of events. Hence, group events relevant to specific users or tenants who are of interest to the investigation. It protects the data corresponding to other tenants thereby achieving segregation. They apply rule based event correlation techniques to group events logged by tenants of interests. The system tested and evaluated using Open Stackcloud**.** The work is not adaptive, not detecting new attacks, the performance of the system depends on the knowledge of security experts or rule creators, easy to bypass, not adaptive to changes, high false positive and negative.

Zhang *et al* [104] proposed an approach to malicious payload detection based on Bayesian inference. The approach uses n-gram to extract feature patterns from both attack and benign payloads and used the feature pattern to predict whether the incoming payloads is attack or benign. The proposed approach was evaluated on the CSIC -2010 HTTP Dataset. The proposed work has done based on traditional machine learning algorithms due to this, the

algorithm needs domain experts to extract features to be learned and the performance of the algorithm depends the experts, high false positives and negatives.

Chih-Hung *et al* [67] proposed an approach, which helps to calculate the similarity values among alerts and aggregate the similar alerts to eliminate duplicate low level alerts. The features of alerts include: source IP address, destination IP address, source port, destination port, alert time stamp. After calculating the similarity, the relationship of alerts can be realized. The system calculates the correlation probability between two alerts by using the techniques of multilayer perceptron (MLP) and support vector machine (SVM), and then adds the correlation probabilities into alert correlation matrix. The alert correlation probability is calculated by mapping features of two alerts on the result table built by MLP and SVM. The proposed approach evaluated using the DARPA 2000 dataset. The proposed work has done based on traditional machine learning algorithms due to this, the algorithm needs domain experts to extract features to be learned and the performance of the algorithm depends the experts, high false positives and negatives.

Xiaohui *et al* [18] proposed a combination of CNN-LSTM based web attack detection approach. The CNN exteracts high level features and feeds the exteracted features to the LSTM. It learns the extracted sequence of features by memorizing their sequence and classify the HTTP request as attack or normal. The experiment has done using HTTP DATASET CSIC 2010. In the approach LSTM learns only by considering the past features information in unidirectional way.

3.2 Knowledge-Based

Faeiz *et al* [105] proposed an improved require/provide model which establishes a cooperation between statistical and knowledge based model to achieve higher detection rate with minimal false positives. The knowledge based model with a vulnerability and extensional conditions provide manageable and meaningful attack graph. The proposed model has been implemented in real time and has successfully generated security events on a correlation between attack signatures. It analyzed Botnet traffic as a case study to measure the performance of MARS tool. The experiment result achieved an improvement in relation to identification of attack plans, reduction in graph complexity and false positives. But the approach can’t detect new attacks, lack of adaptability, to define the knowledge base needs domain experts and the

performance of the system dependes on the experts.

Ficco *et al* [106] presented a hybrid and hierarchical event correlation approach for intrution detection in cloud computing. It consists of detecting intrution symptoms by collecting diverse information at several cloud architictural levels using distributed security probes as well as performing complex event analysis based on a complex event processing engine. They stated that the escalation process from intrution sysmptoms to the identified causes and target of intrution is driven by a knowledge base represented by an ontology. However, the correlation approach provides means to group logically and temporarily connected symptomes into attack scenarion. The prototype implementation of the proposed solution consists of a collection of software components that transform raw attack symptoms into high level intrusion reports which can be used to analyze the attack strategy and perform recovery actions. The proposed correlation techniue lack of detecting new attacks, it needs specific knowledge about attack strategies, the performance of the system dependes on knowledge base, and lacks of adaptability.

Tayeb *et al* [107] proposed an ontological representation based Description Logic (DL) which is a powerful tool for knowledge representation. It helps to ensure a decidable reasoning. Event correlation prototype is presented in domain ontology based on several data sources. The ontology is implemented using OWL (Web Ontology Language) which is recommended by W3C (World Wide Web Consortium) in 2004 for the representation of Web semantics. They collected data from different sources using different tools like Nmap to get information about host and network topology. Using Nessus collected information about the vulnerabilities of systems and software. Using Snort they collected information about attacks in real time by Intrusion Detection System, and then applied ontology specification language based on the specification correlate events. The proposed approach lacks of detecting new attack and not adaptive to changes, needs domain expert to define the ontology representation, and the effectiveness of the system depends on the representers or domain experts.

3.3 Statistical-Based

Andrey et al [108] proposed an approach to correlate events based on the structure of security event types. The approach used to automate analysis of security events as input data with dynamic content means performing functional and behavioral analysis by computing the

frequency-time characteristics of events, their ranking and building of patterns behavior. Additionally, for automated analysis of events suggested to build a graph of type of event with direct and indirect link among themselves. The proposed approach experimented using MS Windows 8 security log. Used rank correlation methods to know the relation between the type of events and between specific instance of events. The proposed approach has high false positives and negatives, high error rate, lack of detecting multi-stage attack, and easy to bypass.

Igor *et al* [65] proposed an approach to parallel data processing for solving security event correlation problems based on big data technologies. The work mainly on identifying the link between security event types and assessing the dependency of the link strength on event distribution in time based on Pearson correlation coefficient. They implemented the correlation on Spark platform by configuring Spark's application then, a SparkContext entry point is created to provide a link between different event types, and the result of the experiment showed that the data processing time during security event correlation is inversely proportional to the number of parallel threads set in Spark. The proposed approach has high false positives and negatives, high error rate, lack of detecting multi-stage attack, and easy to bypass.

3.4 Summary

All the reviewed works used different approaches towards event correlation for attack detection. In knowledge-based correlation, the system has a lack of finding new attacks it must be updated frequently to detect new attacks, not adaptive to changes. It needs professional expert to develop the knowledge base and the system performance depends on the experts.

In statistical based correlation, the system suffers from high rate of false positive and false negative, lack of detecting multi-stage attack, high error rate, not adaptive and easy to bypass. Where as in similarity based correlation, simple rules and hierarchical rules, there is lack of detecting new attack, the system performance depends on the rule creators or the experts, high false positives and false negatives, not adaptive and easy to bypass. However, with the highly dynamic nature of the attacks today, this becomes exceedingly difficult, if not impossible to solve. Therefore, to solve the problems many works are suggested using artificial intelligence based correlation, from the proposed solutions some of them have worked using traditional

machine learning approaches.

Machine learning approaches learn depending on the feature engineering that have been done using manually by domain experts and their learning capability is limited as compared to deep learning. Additionaly, some of the proposed deep learning approaches such as RNN has lacks of memorizing sequences of features information. CNN has lack of learning seuences of features information. LSTM has lacks extracting high level features information. The CNN-LSTM is a combination of CNN and LSTM but the LSTM learns only by considering the past features information. However, this shows that there is a need for further research work to detect sophisticated web attacks based on events correlation using a convolutional neural network and feeds it to the bidirectional long short-term memory recurrent neural network.

Chapter 4: The Proposed Web Attack Detection Model

4.1 Overview

In this chapter, we will discuss the design of a correlation based web attack detection using deep learning approaches. Also, the design considerations of proposed model and general architecture of it will be discussed. Additionaly, way of training and testing the proposed model will be discussed.

4.2 Design Considerations

While we design and implement the proposed system, we consider the following considerations**:**

- New attack detection,
- Minimizing false positives, false negatives
- Adaptability, and
- Reducing operational cost.

4.2.1 New Attack Detection

The correlating events (payload of the HTTP request packet) from different sources helps to detect previously unknown behaviors and these behaviors are indicative of a new attack.

Thus, our system extracts an abstract representation of payloads which gives a clue in a holistic view to discover new trends of attack rather than viewing individually. In this way, we can detect the new attack. Furthermore, the system will be able to memorize and correlate a bidirectional long term dependencies of the payload information to present tasks as attackers takes steps to reach their goal.

4.2.2 Minimizing False Positives and False Negatives

False positives are attack data that the system falsely detects as normal. Where as false negative is normal activity mistakenly identified as an attack.

To alleviate this problem, the proposed system correlates event (payload of the HTTP request packet) and represents them in some abstract (high-level) features to make the system focused

on high-level payload data rather than analyzing individually and this helps to minimize false positive and false negatives. Also, the system memorizes payload information for a long period of time including situations before, during, and after events happened by analyzing and learning their behavioral dependency relationship that helps to minimize false positives and false negatives. Additionally, training the proposed model using both normal and attack payload data to learn their pattern and isolate them helps to minimize false positives and false negatives.

4.2.3 Adaptability

Adaptability is defined as the ease with which attack behaviors and methods are sophisticated so that the system or part of the system will be changed according to their behavioral sophistication.

Therefore, the proposed system is capable of learning the relationship between payload data information by remembering for a long period of time. Relating previous and future payload data information to present tasks helps the system to be self-adjusted accordingly as the attack behaviors changed.

4.2.4 Reducing Operational Cost

The proposed system reduces the operational coast using convolutional and max-pooling layers of the convolutional neural network by producing space invariant discriminative of the payload data through dimensionality reduction of important features. The computational power required to process the data is reduced due to the consequence of the dimensionality of data. We use word embedding to compress the input payload feature space into a smaller one. This helps to reduce the operational cost of the system.

4.3 System Architecture

We propose a generalized architecture of the proposed web attack detection system. The system contains three modules with their subcomponents as shown in Figure 4.1. These are Data Preprocessing Module, Training Module, and Attack Detection Module. The details of each will be described in the next sections.

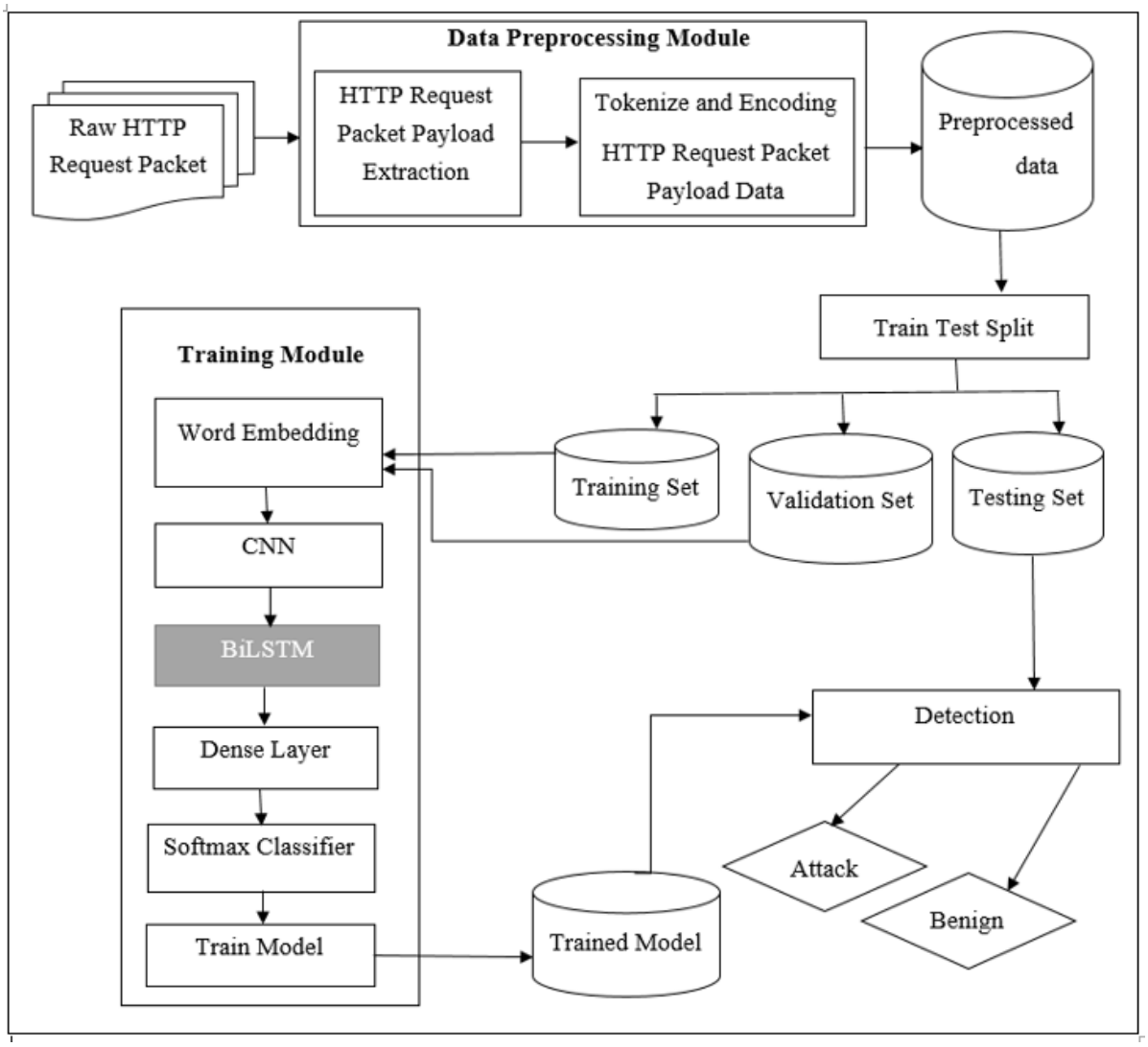

Figure 4. 1: General Architecture of the Proposed System (Some of the F*eatures are Adopted from [18])*

4.3.1 Data Pre-processing Module

The data preprocessing module is responsible for the extraction of events (HTTP request packet payload) and tokenizing and encoding of the extracted payload data.

a. Raw HTTP Request Packet

The raw HTTP request packet is an individual stream of HTTP that is requested by a user or client to get services of a Web server. We use it for further processing (extracting) of HTTP request packet payload.

b. HTTP Request Packet Payload Extraction

We extract payloads by parsing the raw HTTP request packet, which is the actual representation of the resource. Since an HTTP request packet consists of many components, for our proposed Web attack detection system, we only need the payload part of the request packet. Therefore, we remove all the other features of the request packet and keep the payload part.

c. Tokenize and Encode HTTP Request Packet Payload Data

Tokenization is applied on the extracted HTTP request packet payload data. It will tokenize into a sequence of tokens or pieces using the "&" symbol which separates a series of parameters with their values. The tokens of payload data are represented by a unique number or mapping of tokens to integers. We encode to process (training and testing) the proposed system model since machine learning models take arrays of numbers as input. As a result, the sequence of L tokens is represented as a sequence of L encoded integer vectors $(X_1, X_2, \ldots, X_L)$ .Algorithm 4.1 is an algorithm for the tokenization and encoding of payload data.

```
Input: HTTP request packet payload data
Begin:
 Do
  Read  HTTP request packet payload data
  For every HTTP request packet payload data
  Perform Tokenization based on the "&" symbol
  For each Tokens of packet payload
   Encode to unique numbers
  While (end of packet Payload data)
   Return Encoded tokens of packet payload
 End
 Output: Encoded tokens of packet payload data
```

Algorithm 4. 1: Algorithm for Tokenizing and Encoding of a Request Packet Payload

4.3.2 Train Test Split

The preprocessed data are splited into training, testing and validation sets. The purpose of

splitting is in any model building process to develop a generalizable model on the available data which will perform well on unseen data. But to estimate a model's performance on unseen data, we need to have a separate unseen data. This is achieved by splitting our available data into training, validation, and testing sets.

Training set: is a set of sample data used for learning, that is used to fit the hyperparameters of the model. Out of the preprocessed dataset, 65% of the data are used for training the model.
Validation set: a set of sample data used to tune the hyperparameters of a model, which provides an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. Out of the preprocessed dataset, 15% of the data are used for validating the model.
Testing set: a set of sample data used to assess the performance of a final model that are fitted on the training dataset. The remaining of the data are used for testing the performance of the trained model.

4.3.3 Training Module

The Training Module is responsible for training the proposed model using the training payload dataset. The model is a combination of different components like Word Embedding, Convolution Neural Network, Bidirectional Long Short-Term Memory Recurrent Neural Network, Dense Layer, and Softmax Classifier. The details of each component are already discussed in Chapter 2.

Training Model

Training the model starts by specifying the parameters of optimization, cost (loss) function, and metric. We train the model using the Adaptive Moment Estimation (Adam) gradient descent optimization algorithm. It is computationally efficient and has little memory requirement.

The general procedure for training the built model is as follows, (some of the ideas are taken from [109]).

- First initialize some random weights for the embedding layer.

- Then propagate feedforward of those initialized weights to the next layer, from the input layer to the next layer feedforward until it reaches to the output layer, and then finally compute the loss of input weights to the desired output.
- After that, the computed result is not matched to the desired output (there is an error or loss in the output when compared to input weights).
- Then propagate the loss or error to feedbackward from the output layer entirely to the input layer. It helps to update the weights of randomly initialized values (the error is reduced through feedbackward propagation of error). As a result, the initialized weights move to the right direction.
- Repeat feedforward and feedbackward propagation processes until training stop criteria (no of epochs) is reached. Epochs are the number of times the model will cycle through the data.

To Initialize weights for embedding using random initializer is shown in algorithms 4.2.

```
Input: input_dim, output_dim
Begin
 mean = 0, stddev= 0.1
 For input_dim I
  For output_dim J
   Initializer[I, J]=np.float32(random.uniform (mean,stddev)
 Return Initializer
End
Output: Initializer
```

Algorithm 4. 2: Algorithm for Embedding Weight Initialization

Training of the model is performed based on the initialized random weight then moving forward (feedforward propagation) and returns back (feedbackward propagation) through the layers by iterating on the training dataset until it reaches the specified epoch number.

The embedding layer feedforward randomly initialized weights to the convolutional layer and then the convolutional layer again feedforward to the next layer relaying this way until the final layer (output) is reached. Algorithm 4.3 shows how feedforward propagation is performed during training of the model.

Input: Training dataset, Initializer

Began

Do

For every tokens payload data in the training dataset

For all embedding dimensional space

Create embedding for every tokens payload in the dataset

For all embedding tokens in convolutional layer

For every nodes in the layer

Calculate the convolution operation

$s_t^i = \emptyset(f_i^T Y_{t:t+h-1} + b)$, $S = \{s^i\}_{i=1}^n$.

End // for embedding space

For all convolved features in polling layer

For every nodes in the layer

Calculate max pooling operation

$p^i = \max_{t\in\{1,\ldots,L-h+1\}} \{s_t^i\}$

End // for convolutional layer nodes

End // for convolutional layer

For each max features in BiLSTM layers

For every nodes in the layer

Calculate the output of hidden states

$y_t = \sigma(\overrightarrow{h_t}, \overleftarrow{h_t})$

End // for pooling layer nodes

End // for pooling layer

For each y_t output

Make fully connected (N)in dense layer

End // for BiLSTM node

End // for BiLSTM layer

For N fully connected layers

Calculate the classifier function

M = Softmax(N)

End // for dense layer

```
    While (Epochs limit is reached)
  End
  Return function result
Output: M function result
```

Algorithm 4. 3: Algorithm for Feedforward Propagation

After the feedforward propagation is performed, the next is computing loss function for our classification model by measuring the probability distribution difference for input data (expected classification) to output result (predicted), and perform backward propagation for updating of the node weight from the output to the input back. The loss function is computed as follows [110]:

$$\text{EntropyLoss} = -(\mathcal{Y}\log(\mathcal{P}) + (1-\mathcal{Y})\log(1-\mathcal{P})) \quad (16)$$

Where: $\mathcal{Y}$ is the expected probability or the known probability of the classification label and $\mathcal{P}$ is the predicted probability by the model of classification label .

The calculated loss information is propagated back from the output classifier function to the input embedding layer by receiving the loss signal and updating of the node's weight entirely through the layers. Algorithm 4.4 shows how backward propagation works during training the model.

```
Input: input and output of the layers
 Begin
  Do
   Propagate losses to backward through the layers
    For all layers
    For every nodes in the layer
      Calculate the nodes signal loss
      Updates each nodes weight in the layer
  While (end of input)
  End //for nodes
  End // for layers
  Return nodes weight update parameter
  Save the Model
End
 Output: nodes weight update parameters
```

Algorithm 4. 4: Algorithm for Feedbackward Propagation

4.3.4 Trained Model

The trained model is the final result, which the tokens payload feature dataset is trained with the learning algorithms. In other words, it is a representation of the learned attack and benign behavior of HTTP request packet payload data.

4.3.5 Detection

The detection is responsible for testing the trained model by isolating new (untrained) tokens payload test data and classify them as attack or benign based on their learned behavioral patterns. To test the trained model, untrained tokens payload data will be fed to the trained model. Using the training dataset, the model learns our problem domain and makes predictions on the new test data which helps to assess the performance of the proposed web attack detection model. Algorithm 4.5 shows how the trained model classifies the fed test data.

```
Input: Test data, Trained Model
Begin:
 Feed Test data to the Trained Model
 For all test data
   Analyze and compare to the Trained Model pattern
   Return classification
 End
End
Output: classification
```

Algorithm 4. 5: Algorithm for Classification

4.4 Summary

In this chapter, we discussed the design of the proposed system with the design considerations we take while designing the proposed system such as new attack detection, minimizing false positives and false negatives, adaptability, and reducing operational cost. The general architecture of the proposed system is presented. It is composed of different modules such as Data Preprocessing, Training, and Detection. The Data Preprocessing module is responsible for preparing HTTP requested packet payload data for training and testing of the proposed model. The Training module is responsible for training the defined model to learn the problem domain of web attack using HTTP requested packet payload data. Then the trained model is tested using the test data and classifies them as benign or attack.

Chapter 5: Implementation and Evaluation

5.1 Overview

In this chapter, we discuss the various experiments to evaluate the performance of the proposed system model. The dataset used for experimentation, list of the development environment, the performance evaluation techniques of the proposed system model and its result will provide in the next sections.

5.2 Dataset Preparation

The dataset we used for conducting experiment on CSIC 2010 HTTP dataset [111], it was developed at the Information Security Institute of Spanis Research National Council. The requests are labeled as benign and attack, which contain thousands of automatically generated HTTP request packets and the dataset includes various type of web attacks such as SQL injection, buffer overflow, information gathering, files disclosure, CRLF injection, XSS, server side include, parameter tampering. In our experiment, we used only the payloads of HTTP request packet.

1. GEThttp://localhost:8080/tienda1/publico/autenticar.jsp?modo=entrar&login=frayda &pwd=422RdO&remember=off&B1=Entrar HTTP/1.1
2. User-Agent: Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)
3. Pragma: no-cache
4. Cache-control: no-cache
5. Accept:text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q= 0.8,image/png,*/*;q=0.5
6. Accept-Encoding: x-gzip, x-deflate, gzip, deflate
7. Accept-Charset: utf-8, utf-8;q=0.5, *;q=0.5
8. Accept-Language: en
9. Host: localhost:8080
10. Cookie: JSESSIONID=567E89E8E503C715129A202B4986FDC8
11. Connection: close

Figure 5. 1: Example of HTTP Request Packet

After parsed and extracted the Figure 5.1 raw HTTP request packet, the payload of the request packet which are used for our proposed system is seem as below.

```
modo=entrar&login=frayda&pwd=422RdO&remember=off&B1=Entrar
```

Table 5. 1: Details of the Dataset

Class	Train	Validation	Test
Normal	16621	3835	5114
Attack	10966	2531	3374

5.3 Implementation Environment

We use different tools for implementing the proposed system model. For parsing and extracting payloads from raw HTTP request packet we used Microsoft excel. For tokenizing and encoding extracted payloads, and training of them is done using anaconda Keras framework with TensorFlow backend Python library. We run experiments under the environment of Microsoft Windows 10 Professional (64-bit), 8 GB of memory, Intel(R) Core(TM) i5-7200U CPU.

5.4 Implementation prototype

For implementing the proposed system model, there are different components of the model hyper-parameters needed to be configured. Such as, the embedding, CNN, BiLSTM hidden states and dense layer. We configured the necessary hyper-parameters and their corresponding values as shown in table 5.2:

Table 5. 2: Hyper-parameter Configuration

	Name of Hyper-parameters	**values**
Embedding	Embedding_dim	64
	Input_dim	100
CNN Convolution layer	Filter_size	32, 64
	Kernel_size	4x4, 4x4
	Pooling size	2x2, 2x2
Dropout	Dropout	0.5
BiLSTM	Hidden units	128
Dense	units	128, 2
Training	batch_size	64
	epochs	40
	validation_split	15%
	Training_split	65%

5.5 Evaluation

In this section, we explain the evaluation techniques of implemented system model, its results, and comparing the results with other related works.

5.5.1 Results

The trained system model is evaluated using the testing dataset and its result shows how is the trained model perform classification on the new (untrained) data.

We generalize the overall performance of the model from the following confusion matrix. The confusion matrix shows the number of actual and predicted classes based on the testing datasets. Each row represents the instances of an actual class and each column represents the instances of a predicted class. Figure 5.2 shows how many of the request packet payloads are

classified correctly or incorrectly from testing dataset.

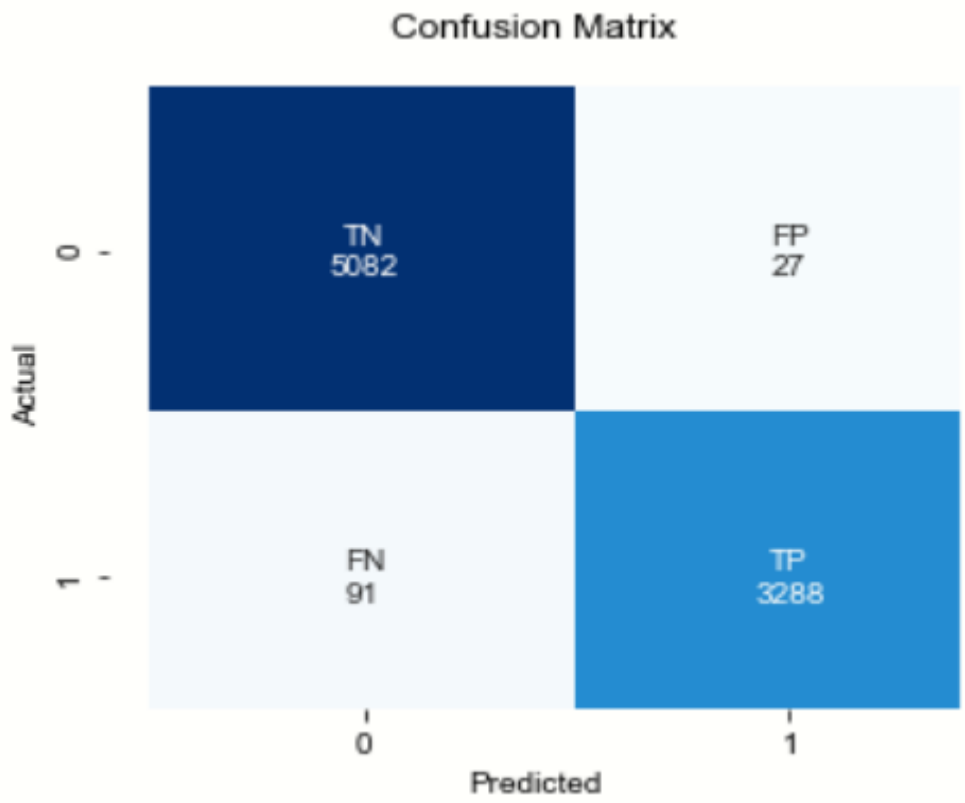

Figure 5. 2:Confusion Matrix of the Model

The classifier shows that among the total number of dataset, 5082 are classified as true negative, which means from the total number of negative test data 5082 are classified correctly as true negative. From the total number of positive data, the model classified correctly 3288 as true positive. From the total number of test data, the model classified incorrectly 27 as false positives, which means the actual negative test data are predicted wrongly as positive. From the total number of test data, the model classified incorrectly 91 as false negatives, which means the actual positive test data are predicted wrongly as negative. The model achieves the large number of true positives and true negatives with minimized false positives and false negatives. From this the overall performance of the model achieved accuracy of 98.6%, precision of 99.2%, recall of 97.3%, and f1_score of 98.2%.

5.6 Comparison

We have tried to compare our work with other related works, as shown in Table 5.3. The CNN and LSTM based separated works are compared with the work done in [15], and the combined CNN-LSTM model are compared with the work done in [18]. The remaining models are conducted using our prepared dataset to show the effectiveness of our proposed CNN-BiLSTM model.

Table 5. 3: Comparison

Models	**Recall**	**Precision**	**F1-score**	**Accuracy**
CNN	91.14%	90.56%	90.85%	94.11%
LSTM	97.79%	94.41%	96.07%	96.13%
BiLSTM	96.6%	99%	97.8%	98.3%
CNN-LSTM	94.3%	94.6%	94.4%	95.5%
CNN-BiLSTM	97.3%	99.2%	98.2%	98.6%

5.7 Summary

The study was performed on a total of 25,569 benign and 16,871 attack payload data extracted from CSIC 2010 HTTP dataset. These payload datasets are divided into 65% for training, 20% for testing and 15% for validation. The experiment was done by using 40 epochs. The experiment result was evaluated using the performance evaluation metrics accuracy, precision, f1-score and recall. Our work, CNN-BiLSTM, achieved better result with accuracy 98.6%, precision 99.2%, f1-score 98.2% and recall 97.3%.

Chapter 6: Conclusion and Future Works

6.1 Conclusion

The web is the world largest diversified pooling source of information. Today, many things like of businesses, education, health systems etc., are executed through web. Due to this, the growth of attacks is radically increased day by day with their sophisticated attacking mechanisms. To solve this problem, this research work had attempted to look into the methods of web attack detection based on the correlation of events (payloads of HTTP request packet) using a deep learning approaches. Since payload is an important issue to be considered for web attack detection, which is when users ask requests to get services from the server it responds to the request based on their payloads. To conduct this, first we articulated the general web attack problems more specifically with related to the payload based attacks. Next, a number of literatures have been reviewed to provide background information about the web, web attacks, current trends of web attack detection mechanisms, event correlation, correlation algorithms, and different research works related to web attack detection using different correlation based approaches.

Then, the design of the proposed web attack detection solution was presented. The proposed solution consists of the preprocessing (HTTP request packet payload extraction, tokenization and encoding the extracted payload data) module, that is essential for extraction of the HTTP request packet payload which is used for our system development, the training module. The model contains Word embedding, CNN, BiLSTM, Dense layer and Softmax classifier. The word embedding is a vector representation of payload tokens. The contextual terms of related payload concept features are extracted. CNN receives input from the embedding and extract the high level payload representation features by correlating the local payload feature and pass to the next layer. The BiLSTM receives from the CNN and learn the high level sequential payload features by providing information from the past and the future, it helps to understand the context in a better way and pass it to the next layer. The Dense layer receives from BiLSTM and make its fully connected and pass to the Softmax classifier classifies the payload as attack or benign. After all the model is built, the next is training it (fit or learning the pattern) using the training dataset. Lastly, the detection is presented. This is the classification part of

the model which identifies pattern of the problem definition of the proposed solution and classified the incoming request packet payloads as attack or benign.

The experiment is performed using the CSIC 2010 HTTP dataset. From the dataset we applied 65% of the data for training of the model, 20% for testing, and 15% for validation of the model. The result demonstrate that our model achieved 98.6% accuracy, 99.2% precision, 98.2% f1-score and 97.3% recall.

6.2 Contribution

In general, the contribution of the study is listed below:

- This study showed that an event correlation based web attack detection using CNN and BiLSTM.
- A generic model is designed for event correlation based web attack detection that takes the advantage of word embedding, CNN and BiLSTM models.
- In addition, this study contributes to the growth event correlation based web attack detection using the integration of different deep learning models.

6.3 Future Works

In this research, we have proposed an event correlation based web attack detection using the integration of different deep learning models and achieved encouraged result. However, there is still a room for further improvement and we will suggest the following points to be taken as future works.

- Incorporate more dataset to enhance the performance
- Include other types of attack datasets to increase their detection functionality
- Integrating other types of deep learning models by considering their advantages, the gap of one will be filled by the other ones.
- Combining other attack detection systems like of considering client side attacks, IOT and others.

References

[1] M. Zhang, S. Lu and B. Xu, "An Anomaly Detection Method based on Multi-models to Detect Web Attacks," *10th International Symposium on Computational Intelligence and Design (ISCID),* pp. 404-409, 2017.

[2] J. Liang, W. Zhao and W. Ye, "Anomaly-Based Web Attack Detection : A Deep Learning Approach," *Proceedings of the 2017 VI International Conference on Network, Communication and Computing,* pp. 80-85, 2017.

[3] B. Kelly, L. Ryan and D. Paolo, "The Cost of Cybercrime," *Ninth Annual Cost of Cybercrime Study,* pp. 1-20, 06 Mar 2019.

[4] S. Aimoto, P. Bange, A. Cooley, S. Doherty, B. Duckering and J. Duff, "Internet Security Threat Report," Symantec, 2018.

[5] C. Beek, C. Castillo, C. Cochin, A. Dolezal, S. Grobman and C. McFarland, "McAfee Labs Threats Report," McAfee Global Threat Intelligence (McAfee GTI), 2018.

[6] Technologies Positive, "Web Application Vulnerabilities," *Statistics For,* pp. 1-14, 2018.

[7] Foundation, OWASP, "The Ten Most Critical Web Application Security Risks," *OWASP Top 10 - 2017,* pp. 1-24, 2017.

[8] E. Zhang, "Blog: what-event-correlation-examples-benefits-and-more," https://digitalguardian.com, [Last accessed on 20 Apr 2019].

[9] R. N. Akash, T. H. Tram, M. M. Purnima and G. Mohan, "Crossfire Attack Detection using Deep Learning in Software Defined ITS Networks," *IEEE 89th Vehicular Technology Conference (VTC2019-Spring),* pp. 1-6, 2019.

[10] A. Miniwatts Marketing Group, "World Internet Usage and Population Statistics," Internet World Stats, 2018.

[11] J. Kaji, A. Rao, P. Devan and A. Khan, "The 2018 Deloitte-NASCIO Cybersecurity Study," 2018.

[12] Z. Ming, X. Boyi, B. Shuai, L. Shuaibing and L. Zhechao, "A Deep Learning Method

to Detect Web Attacks Using a Specially Designed CNN," *Neural Information Processing ICONIP 2017, Springer, Cham,* vol. 10638, pp. 828-836, 2017.

[13] Z. Shuying, G. Yue, Z. Mengqun, G. Jianmei and W. Shuangli, "The Study of Network Security Event Correlation Analysis Based on Similar Degree of the Attributes," *2013 Fourth International Conference on Digital Manufacturing & Automation,* pp. 1565-1569, 2013.

[14] L. Jianyi, L. Sida and Z. Ru, "Algorithm of reducing the false positives in IDS based on correlation Analysis," *IOP Conference Series: Materials Science and Engineering,* vol. 322, pp. 1-5, 2018.

[15] L. Hongyu, L. Bo, L. Ming and Y. Hanbing, "CNN and RNN based payload classifcation methods for attack detection," *Knowledge-Based Systems,* vol. 163, p. 332–341, 2019.

[16] J. Xiaohui, C. Baojiang, Y. Jun and C. Zishuai, "Payload-Based Web Attack Detection Using Deep Neural Network," *Springer, Advances on Broad-Band Wireless Computing, Communication and Applications,* pp. 482-488, 2018.

[17] J. Vierthaler, R. Kruszelnicki and J. Sch, "WebEye – Automated Collection of Malicious HTTP Traffic," *Fraunhofer Research Institute for Applied and Integrated Security,* pp. 1-11, 2018.

[18] K. Xiaohui, Z. Ming, L. Hu, Z. Gang, C. Huayang, W. Zhendong and W. Xianmin, "DeepWAF: Detecting Web Attacks Based on CNN and LSTM Models," *International Symposium on Cyberspace Safety and Security, Springer,* pp. 121-136, 2019.

[19] C. Zhiyong, K. Ruimin, P. Ziyuan and W. Yinhai, "Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Forecasting Network-wide Traffic State with Missing Values," *arXiv preprint arXiv:2005.11627 ,* 2020.

[20] A. W3C, "World Wide Web Consortium," in *http://www.w3.org*.

[21] G. Brian, "Basic Definitions: Web 1.0, Web. 2.0, Web 3.0," *http://www.practicalecommerce.com/,* 2007.

[22] B.-L. Tim, "The World Wide Web: A very short personal history," World Wide Web

Consortium, 1998.

[23] C. Robert and B.-L. Tim, "Proposal for a HyperText Project," World Wide Web Consortium, 1990.

[24] A. A. Abdulelah, A. Saleh, F. K. Samer and M. Austin, "Web Evolution - The Shift From Information Publishing to Reasoning," *International Journal of Artificial Intelligence and Applications (IJAIA),* vol. 8, no. 6, pp. 11-28, 2017.

[25] B. Tim, C. Robert and F. G. Jean, "The world-wide web," *Computer Networks and ISDN Systems,* vol. 25, no. 4, pp. 454-459, 1992.

[26] Tim O'Reilly, "What is Web 2.0—Design patterns and business models for the next generation of software," *O'Reilly Media Inc,* no. http://www.oreillynet.com/, 2005.

[27] B. Diana, P. Marta, R. A. Carlos and C. Alberto, "Web 5.0: the future of emotional competences in higher education," *Global Business Perspectives,* vol. 1, no. 3, pp. 274-287, 2013.

[28] Z. Jeffrey, "Web 3.0. A list apart," *Issue № 210,* 17 January 2006.

[29] A. Kambil, "What is your Web 5.0 strategy?," *Journal of Business Strategy,* vol. 29, no. 6, pp. 56-58, 2008.

[30] "Web_Architecture," https://en.ryte.com/wiki/, [Last accessed on 01 May 2019].

[31] T. Scott, "Research directions in web systems evolution V: Architecture," *15th IEEE International Symposium on Web Systems Evolution (WSE),* pp. 103-103, 2013.

[32] B.-L. Tim, B. Tim, C. Dan, C. Paul, F. Roy, L. Chris, O. David, W. Norman and W. Stuart, "Architecture of the World Wide Web," *World Wide Web Consortium,* 2003.

[33] C. Chhorn, "Introduction to web architicture," http://author.uthm.edu.my/, 2011.

[34] A. Nuzzolese, "The World Wide Web Architectures," www.cs.unibo.it/.

[35] S. Anubha, K. Manoj and A. Sonali, "A Complete Survey on Software Architectural Styles and Patterns," *4th International Conference on Eco-friendly Computing and Communication Systems,* vol. 70, pp. 16-28, 2015.

[36] F. Henrik, "Thesis on the World-Wide Web," *World Wide Web Consortium,* 1994.

[37] M. Lindner, "Components of the World Wide Web," *Institute of Computer Technology*

- *Vienna University of Technology,* pp. 45-63, 2005.

[38] Stats Internet Live, "Total number of websites," Internet Live Stats, 2017, http://www.internetlivestats.com/.

[39] W. Zijian, "Global b2c e-commerce report," 2016, www.ecommercewiki.org.

[40] Service, UK Government Digital, "How digital and technology transformation saved 1.7bn last year," 2015, https://gds.blog.gov.uk/.

[41] Symantec, "Internet security threat report," https://www.symantec.com/, 2016.

[42] C. Michał, K. Rafał, F. Adam and H. Witold, "Ontology Applied in Decision Support System for Critical Infrastructures Protection," *Trends in Applied Intelligent Systems, IEA/AIE 2010,* vol. 1, pp. 671-680, 2010.

[43] C. Michał, K. Rafał, P. Rafał, B. Juliusz and H. Witold, "Network Events Correlation for Federated Networks Protection System," *Towards a Service-Based Internet, Springer, Berlin, Heidelberg,* pp. 100-111, 2011.

[44] C. Michał and K. Rafał, "Network Event Correlation and Semantic Reasoning for Federated Networks Protection System," *Springer,* vol. 245, pp. 48-54, 2011.

[45] A. T. Tuan, M. Lotfi, M. Des, A. R. Z. Syed and G. Mounir, "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking," *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM),* pp. 258-263, 2016.

[46] K. Mclntyre, "Event correlation System – The new threat frontline," *SaNS GSEC practecal,* 2003.

[47] A. Amruta and S. Narendra, "Insider threat Detection using Log analysis and Event Correlation," *International Conference on Advanced Computing Technologies and Applications (ICACTA-2015),* vol. 45, p. 436 – 445, 2015.

[48] V.-S. Genoveva, A.-O. Javier and L. Z.-M. José, "Big continuous data: dealing with velocity by composing event streams," *Big Data Concepts, Theories and Applications, Springer Verlag,* pp. 1-27, 2016.

[49] P. Gmv and U. Tuda, "Proactive Risk Management through Improved Cyber Situational Awareness," European Union's Horizon 2020 Research and Innovation

Programme, 2017.

[50] Y. Xie, "A Spatiotemporal Event Correlation Approach to Computer Security," *School of Computer Science Carnegie Mellon University,* pp. 17-30, 2005.

[51] Z. Hongliang, L. Wenhan, S. Maohua and X. Yang, "A Universal High-Performance Correlation Analysis Detection Model and Algorithm for Network Intrusion Detection System," *Hindawi,* pp. 1-8, 2017.

[52] M. Tiffany, "A Survey of Event Correlation Techniques and Related Topics," *ResearchGate ,* 2002.

[53] J. Skinner, "An Introduction to Event Modeling and Correlation," https://slideplayer.com.

[54] D. Neelam and T. Aprna, "Event Correlation for Intrusion Detection Systems," *2015 IEEE International Conference on Computational Intelligence & Communication Technology,* pp. 134-139, 2015.

[55] U. Martin, C. Feng and M. Christoph, "Event Attribute Tainting: A New Approach for Attack Tracing and Event Correlation," *2016 IEEE/IFIP Network Operations and Management Symposium (NOMS 2016),* pp. 509-515, 2016.

[56] O. A.-M. H. L. Z. Safaa, "A survey on IDS alerts processing techniques," *Proceedings of the 6th WSEAS international conference on Information security and privacy,* pp. 69-78, 2007.

[57] Iansresearch.com, "Anchor your security with a well-honed SIEM strategy," https://www.iansresearch.com/insights/guides/.

[58] A. M. Seyed, A. Sajjad and J. Rasool, "Alert correlation algorithms: a survey and taxonomy," *Proceedings of the 5th International Symposium on Cyberspace Safety and Security (CSS). Zhangjiajie, China,* p. 183–197, 2013.

[59] V. Fredrik, V. Giovanni, K. Christopher and A. K. Richard, "A Comprehensive Approach to Intrusion Detection Alert Correlation," *IEEE Transactions on Dependable and Secure Computing,* vol. 1, no. 3, pp. 146-169, 2004.

[60] T. E. Huwaida and M. O. Izzeldin, "Intrusion Alert Correlation Framework: An Innovative Approach," *IAENG Transactions on Engineering Technologies. Lecture*

Notes in Electrical Engineering, Springer, Dordrecht, vol. 229, pp. 405-420, 2013.

[61] V. Alfonso and S. Keith, "Probabilistic Alert Correlation," *4th International Symposium on Recent Advances in Intrusion Detection,* pp. 54-68, 2001.

[62] F. Cuppens and O. New, "Managing alerts in a multi-intrusion detection environment," *Seventeenth Annual Computer Security Applications Conference,* pp. 22-31, 2001.

[63] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Transactions on Information and System Security (TISSEC),* vol. 6, no. 4, pp. 443-471, 2003.

[64] O. A.-M. Safaa and Z. Hongli, "IDS alerts correlation using grammar-based approach," *Journal of Computer Virology, Springer-Verlag,* vol. V, no. 4, pp. 271-282, 2009.

[65] K. Igor, V. F. Andrey, S. Igor and K. Alexey, "Parallelization of Security Event Correlation Based on Accounting of Event Type Links," *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP),* pp. 462-469, 2018.

[66] D. LlU and Y.-q. WEI, "Study on Event Correlation Analysis in Evidence Chain Structure," *IEEE 2012 International symposium on information technology in medicine and education,* pp. 1056-1060, 2012.

[67] W. Chih-Hung and C. Ye-Chen, "Alert Correlation System with Automatic Extraction of Attack Strategies by Using Dynamic Feature Weights," *International Journal of Computer and Communication Engineering,* vol. 5, 2016.

[68] T. E. Huwaida and M. ,. Izzeldin, "Alert correlation in collaborative intelligent intrusion detection systems A survey," *Elsevier: Applied Soft Computing,* pp. 4349-4365, 2011.

[69] L. Yansong and Z. Li, "A new intrusion detection and alarm correlation technology based on neural network," *EURASIP Journal on Wireless Communications and Networking,* pp. 1-10, 2019.

[70] A. Saad, A. M. Tareq and A.-Z. Saad, "Understanding of a convolutional neural network," *International Conference on Engineering and Technology (ICET),* pp. 1-6, 2017.

[71] L. M. Remish, P. N. Sanjana, R. Ramesh and I. Yoshiaki, "Cyber Security Using Bayesian Attack Path Analysis," *CYBER 2018 : The Third International Conference on Cyber-Technologies and Cyber-Systems,* pp. 15-22, 2018.

[72] K. Fatemeh and A. Behzad, "A Bayesian network-based approach for learning attack strategies from intrusion alerts," *Security and Communication Networks,* p. 833–853, 2013.

[73] W. Ding, H. Haibo and L. Derong, "Adaptive Critic Nonlinear Robust Control: A Survey," *IEEE Transactions on Cybernetics,* vol. 47, no. 10, pp. 3429-3451, 2017.

[74] L. Jonghoon, K. Jonghyun, K. Ikkyun and H. Kijun, "Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles," *IEEE Access,* vol. 7, pp. 165607-165626, 2019.

[75] K. Asifullah, S. Anabia, Z. Umme and S. Q. Aqsa, "A Survey of the Recent Architectures of Deep Convolutional Neural Networks," *Artificial Intelligence Review* , pp. 1-62, 2020.

[76] Y. Yong, S. Xiaosheng, H. Changhua and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Massachusetts Institute of Technology Neural Computation,* vol. 31, no. 7, pp. 1235-1270, 2019.

[77] A. Graves and S. Jürgen, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural networks,* vol. 18, no. 5-6, pp. 602-610, 2005.

[78] G. Alex, J. Navdeep and M. Abdel-rahman, "Hybrid speech recognition with Deep Bidirectional LSTM," *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 273-278, 2013.

[79] B. Jason, "What Are Word Embeddings for Text," https://machinelearningmastery.com/, [Last accessed on 15 July 2020].

[80] X. Guangxu, L. Yaliang, X. Z. Wayne, G. Jing and Z. Aidong, "A Correlated Topic Model Using Word Embeddings," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, Beijing, China, 2017.

[81] B. Jason, "Crash Course in Convolutional Neural Networks for Machine Learning,"

https://machinelearningmastery.com/, [Last accessed on 5 March 2020].

[82] N. Milad, B. Alireza and H. Amir, "DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning," *ACM SIGSAC Conference on Computer and Communications Security*, no. 18, pp. 1962-1976, 2018.

[83] Z. Chunting, S. Chonglin, L. Zhiyuan and C. M. L. Francis, "A C-LSTM Neural Network for Text Classification," *arXiv preprint arXiv:1511.08630*, 2015.

[84] C. Daphne, "An intuitive guide to Convolutional Neural Networks," https://www.freecodecamp.org/, [Last accessed on 12 March 2020].

[85] H. Oinkina, "Understanding LSTM Networks," http://colah.github.io/, [Last accessed on 13 March 2020].

[86] Z. Lipton, B. John and E. Charles, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, pp. 283-287, 2015.

[87] Z. Yu, Y. Rennong, C. Guillaume, X. Ximeng and Z. Zhenxing, "Deep Residual Bidir-LSTM for Human Activity Recognition Using Wearable Sensors," *Hindawi Mathematical Problems in Engineering*, pp. 1-13, 2018.

[88] N. Peng and X. Dingbang, "Learning attack strategies from intrusion alerts," *Proceedings of the 10th ACM conference on Computer and communications security*, pp. 200-209, 2003.

[89] N. Peng, C. Yun and S. R. Douglas, "Analyzing intensive intrusion alerts via correlation," *Recent Advances in Intrusion Detection (RAID), Springer, Berlin*, pp. 74-94, 2002.

[90] N. Peng and X. Dingbang, "Hypothesizing and reasoning about attacks missed by intrusion detection systems," *ACM Transactions on Information and System Security (TISSEC)*, vol. 7, no. 4, pp. 591-627, 2004.

[91] S. Cheung, L. Ulf and F. Martin, "Modeling multistep cyber attacks for scenario recognition," *Proceedings DARPA Information Survivability Conference and Exposition*, vol. I, pp. 284-292, 2003.

[92] M. Benjamin, M. Ludovic, D. Hervé and D. Mireille, "A logic-based model to support alert correlation in intrusion detection," *Information Fusion*, vol. 10, no. 4, pp. 285-

299, 2009.

[93] K. Peyman and G. Ali A, "A rule-based temporal alert correlation system," *International Journal of Network Security,* vol. V, no. 1, pp. 66-72, 2007.

[94] V. Jouni, D. Hervé, M. Ludovic, L. Anssi and T. Mika, "Processing intrusion detection alert aggregates with time series modeling," *Elsevier Science Publishers* , vol. 10, no. 4, pp. 312-324, 2009.

[95] R. Hanli, S. Natalia and A. G. Ali, "An Online Adaptive Approach to Alert Correlation," *Detection of Intrusions and Malware, and Vulnerability Assessment DIMVA 2010, Springer-Verlag Berlin,* pp. 153-172, 2010.

[96] M. Stefanos, C. Marvin, Z. Dan and H. Keith, "A data mining Analysis of RTID alarms," *Elsevier North-Holland,* vol. 34, no. 4, pp. 571-577, 2000.

[97] Q. Xinzhou and L. Wenke, "Attack Plan Recognition and Prediction Using Causal Networks," *20th Annual Computer Security Applications Conference, IEEE Computer Society Washington,* pp. 370-379, 2004.

[98] G. Guofei, C. Alvaro A and L. Wenke, "Principled reasoning and practical applications of alert fusion in intrusion detection systems," *2008 ACM symposium on Information, computer and communications security, ACM New York,* pp. 136-147, 2008.

[99] X. Jin, B. Cui, J. Yang and Z. Cheng, "An Adaptive Analysis Framework for Correlating Cyber- Security-Related Data," *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA),* pp. 915-919, 2018.

[100] S. Kleber, R. M. M. Edmundo and K. G. Siome, "An approach to the correlation of security events based on machine learning techniques," *Springer London Journal of Internet Services and Applications,* vol. 4, no. 7, pp. 1-16, 2013.

[101] H. Chenn-Jung, L. Ching-Yu, W. Yu-Wu, L. Chin-Fa, L. Jia-Jian and H. Kai-Wen, "An Adaptive Rule-Based Intrusion Alert Correlation Detection Method," *2010 First International Conference on Networking and Distributed Computing,* pp. 222-226, 2010.

[102] D. Ying and Z. Yuqing, "Adaptively Detecting Malicious Queries in Web Attacks," *Networking and Internet architicture,* vol. 2, pp. 1-17, 2017.

[103] M. G and G. Geethakumari, "Event Correlation for Log Analysis in the Cloud," *2016 IEEE 6th International Conference on Advanced Computing*, pp. 158-162, 2016.

[104] Z. Z, G. R and S. K, "An Approach to Malicious Payload Detection," *2018 World Automation Congress (WAC)*, pp. 1-5, 2018.

[105] A. Faeiz, A. Monis, U. A. Irfan and J. C. Andrea, "Detection of Coordinated Attacks Using Alert Correlation Model," *IEEE , Informatics Research Institute University of Bradford*, pp. 542-546, 2010.

[106] M. Ficco, "Security event correlation approach for cloud computing," *High Performance Computing and Networking*, vol. 3, no. 7, pp. 173-185, 2013.

[107] K. Tayeb and A. Mahdi, "Toward an Efficient Ontology-based Event Correlation in SIEM," *The 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016)*, p. 139 – 146, 2016.

[108] F. Andrey, K. Igor and E. B. Didier, "Correlation of security events based on the analysis of structures of event types," *The 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, 2017.

[109] "backpropagation-neural-networks-process-examples-code-minus-math/," MissingLink, https://missinglink.ai/. [Last accessed on 23 September 2020].

[110] "loss_functions," Machine Learning Glossary, https://ml-cheatsheet.readthedocs.io/. [Last accessed on 25 October 2020].

[111] "Csic 2010 http dataset," Information Security Institute of Spanis Research National Council, https://www.isi.csic.es/dataset/.

Annex A: Payload Data Loading

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
df = pd.DataFrame()
df = pd.read_csv('C:/Users/Y/For Training/csic_totaldataset.csv', encoding='utf-8')
df.head
```

Out[1]:

	payloads	classification
0	id=1&nombre=Jam%F3n+Ib%E9rico&precio=39&cantid...	0
1	modo=entrar&login=caria&pwd=egipciaca&remember...	0
2	id=2	0
3	errorMsg=Credenciales+incorrectas	0
4	modo=registro&login=minthorn&password=ahorquil...	0

In [3]:
```
df['classification'].value_counts()
```

Out[3]:
```
0    25569
1    16871
Name: classification, dtype: int64
```

In [4]:
```
X=df['payloads'].values
Y=df['classification'].values
```

Annex B: Train-Test Splitting and Tokenization of the payload

In [5]:
```
Xtrain,Xtest,Ytrain,Ytest=train_test_split(X,Y,test_size=0.20 , random_state=1)
# describes info about train and test set
print("Number of xtrain dataset: ", Xtrain.shape)
print("Number of ytrain dataset: ", Ytrain.shape)
print("Number of xtest dataset: ", Xtest.shape)
print("Number of ytest dataset: ", Ytest.shape)
```

```
Number of xtrain dataset:  (33952,)
Number of ytrain dataset:  (33952,)
Number of xtest dataset:  (8488,)
Number of ytest dataset:  (8488,)
```

In [6]:
```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
tokenizer = Tokenizer()
tokenizer.fit_on_texts(Xtrain)
xtrain = tokenizer.texts_to_sequences(Xtrain)
xtest = tokenizer.texts_to_sequences(Xtest)
#max_length = max([len(s.split('&')) for s in total_reviews])
max_length=100
vocab_size = len(tokenizer.word_index) + 1 #Index 0 is often reserved for padding s
xtrain_pad = pad_sequences(xtrain, maxlen=max_length, padding='post')
xtest_pad = pad_sequences(xtest, maxlen=max_length, padding='post')
```

Annex C: Detect and Classify using The Model

In [21]:
```
from keras.models import load_model
model.save("best_model2.h5")
print("Saved model to disk")
loaded_model = load_model("best_model2.h5")
print("Loaded model from disk")
```

```
Saved model to disk
Loaded model from disk
```

In [22]:
```
feed_test_data_pos1= "modo=registro&login=neddie&password=laNC00DA&nombre=Sami&apellidos=Buxedas+Murcia
feed_test_data_pos2=  "modo=entrar&login=grimshaw&pwd=G%2F%2FlAc%2CIAr&remember=on&B1=Entrar"
feed_test_data_neg1= "id=3%2F&nombre=Jam%F3n+Ib%E9rico&precio=85&cantidad=1&B1=A%F1adir+al+carriid=2&nc
feed_test_data_neg2= "id=1&nombre=Vino+Riojaany%253F%250D%250ASet-cookie%253A%2BTamper%253D104126401102
test_samples = [feed_test_data_pos1, feed_test_data_pos2,feed_test_data_neg1,feed_test_data_neg2]
test_samples_tokens=tokenizer.texts_to_sequences(test_samples)
test_samples_tokens_pad = pad_sequences(test_samples_tokens, maxlen=max_length)
Classification=loaded_model.predict_classes(test_samples_tokens_pad)
print(Classification)
```

```
[0 0 1 1]
```

Declaration

I, the undersigned, declare that this research is my original work and has not been presented for degree in any other university, and that all sources of materials used for the research have been acknowledged.

Declared by:

Name: Yitayal Kassie Mikru

Signature: _____

Date: _____

Confirmed by advisor:

Name: Mulugeta Libsie (PhD)

Signature: _____

Date: _____

Place and date of submission: Addis Ababa University, October 2020