Spring 2016

Defeating n-gram Scores for HTTP Attack Detection

Samyuktha Sridharan
San Jose State University

Defeating n-gram Scores for HTTP Attack Detection

A Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Samyuktha Sridharan

May 2016

The Designated Project Committee Approves the Project Titled

Defeating n-gram Scores for HTTP Attack Detection

by

Samyuktha Sridharan

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2016

Dr. Mark Stamp	Department of Computer Science
Dr. Thomas Austin	Department of Computer Science
Fabio Di Troia	Department of Computer Science

ABSTRACT

Defeating n-gram Scores for HTTP Attack Detection

by Samyuktha Sridharan

Web applications that generate malicious HTTP requests provide a platform that attackers use to exploit vulnerable machines. Such malicious traffic should be identified by network intrusion detection systems, based on traffic analysis. Previous research has shown that n-gram techniques can be successfully applied to detect HTTP attacks. In this research, we analyze the robustness of these n-gram techniques. We show that n-gram scores are surprisingly robust, but can be defeated using certain obfuscation strategies. We also consider the need for a more costlier HMM-based intrusion detection system.

ACKNOWLEDGMENTS

Several people have been motivating, encouraging and supporting me throughout my graduate school journey. I am thankful to my father, Mr. Sridharan Srinivasan and my mother, Kamala Sridharan for their unconditional love and sacrifices, through every step in my life. My husband, Dr. Venkatesh Arasanipalai Raghavan, has been my motivation and support for the entire duration of my Master's degree program. I would like to thank my sister, Shruthy Sridharan, who has always stood by me. My sincere thanks to my father-in-law, Mr. A.J.Raghavan and my mother-in-law, Rajam Raghavan, for believing in me and encouraging me through the course of my study.

I would like to thank Dr. Mark Stamp for providing me with advise and guidance throughout the project. Dr. Stamp's CS 265 course made me realize my interest towards research topics in Security. I would also like to thank my committee members, Dr. Thomas Austin and Fabio di Troia, for their valuable feedback during the project.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Cybersecurity has become one of the most critical elements in our networked world. Most of the real-world applications, including bank transactions, e-health, finance management, e-commerce, rely on the Internet. With the evolution of cloud-based computing, the use of the Internet is becoming an integral part of everyone's life.

Malicious code writers have constructed web-based applications that generate malicious Hypertext Transfer Protocol (HTTP) requests. These malicious HTTP requests provide opportunities to attack vulnerable machines. Many Intrusion Detection Systems (IDS) are available to alert administrators when there is abnormal or suspicious activity on the system.

There are two different types of intrusion detection techniques namely, signature-based intrusion detection and anomaly-based intrusion detection. According to [13], Signature-based systems are effective in detecting known attacks. However, they are inefficient in detecting carefully morphed variants of the same attacks. To detect variants of known attacks and other intelligent attacks—otherwise known as "zero-day attacks"—anomaly-based detection techniques [18] are efficient. It can be complex to design and implement anomaly based detection techniques. In [12], an efficient, simple, and effective HTTP attack detection technique based on n-gram analysis was implemented and tested. In [2], a more complex and costly HMM-based approach was explained.

Obfuscation of HTTP data is not uncommon. There are several obfuscation techniques that attackers can use to disguise the original data. Typically, obfuscated data is transmitted through the network for the purpose of bypassing the IDS at the receiving end.

This research aims to analyze the robustness of the n-gram scores in [12], by obfuscating the HTTP attack traffic. This research is intended to aid in developing more robust and efficient algorithms to detect HTTP based attacks.

The rest of the chapters are organized as follows. A thorough review of literature is provided in Chapter 2. Chapter 3 gives a detailed explanation of the research methodology adopted in this research. The datasets used in the research, different experiments conducted for each of the research methodology and the results are presented in Chapter 4. The conclusions for this research and future work are discussed in Chapter 5.

CHAPTER 2

Literature Survey

Cybersecurity has been studied extensively in the academia as well as in the industry. In this chapter, background related to the current research is reviewed to identify any gaps in literature and to leverage the obfuscation techniques useful to this research. The background to intrusion detection techniques are explained during the course of this chapter. Several obfuscation techniques to defeat detection scores have been identified from literature and discussed in this chapter. Different n-gram techniques, which were proved to be very efficient in detecting the attacks by researchers in [12], are studied in the subsequent sections.

2.1 Network Intrusion Detection Systems

Web-based applications are gaining popularity due to its platform-independence feature, ease of development and maintenance, as well as the low cost, pay-per use model. However, no code is error-free, which includes its vulnerability to attacks, which have posed a risk for several years now. Additionally, the large number of web applications also increase the threat of exploitation. Hence, it has become essential to detect these attacks. There are two different intrusion detection techniques:

- Signature based detection
- Anomaly based detection

2.1.1 Signature Based Detection

Majority of the commercial detection programs are based on signature based techniques. A signature based network intrusion detection [13] looks for 'signatures' in the attack payload. The signatures are then matched with the existing signatures that are stored in the databases. Due to the increasing number of signatures, the databases are getting huge and difficult to maintain. Signature based methods are very efficient in detecting 'known attacks'. Signature based detection has very quick processing time and is very reliable. There are very less false alarms in this detection technique.

2.1.2 Anomaly Based Detection

Anomaly based detection technique [18] allows anti-virus programs to detect 'unknown attacks', alternatively termed as 'zero-day attacks'. These types of attacks are harder to detect. Hence, anomaly based detections are complex and costlier to implement. The authors of [12] have implemented a minimally expensive, simple, robust methodology to detect unknown attacks. The authors have filtered out benign traffic, which is easy to implement, and have applied more costly techniques to the filtered traffic. However, according to [17] the zero-day attacks are increasing every year. Table 1 shows the number of new zero-day attacks found during the years from 2006 - 2015 [17]. In 2015, a zero-day attack was detected once every week.

Due to the increasing number of zero-day attacks, it is necessary that the software is able to detect the sophisticated attacks in a robust and efficient manner.

Table 1: New Zero-Day Attacks Found During Each Year [17]

Year	Number of Zero-Day Attacks
2006	13
2007	15
2008	9
2009	12
2010	14
2011	8
2012	14
2013	23
2014	24
2015	54

2.2 HMM-based detection technique

Hidden Markov Model (HMM) based techniques were implemented by authors of [2] to detect zero-day attacks efficiently. This technique uses a costly multi classifier approach for increased accuracy. Though the results showed that these techniques are very accurate in detecting the attacks, however it is very complex to implement. Figure 1 explains the different stages in the implementation of the HMM-based technique. The important step in the implementation is the feature extraction. The features extracted are n-grams from the payload.

2.3 N-gram techniques

The n-gram based techniques have been applied to different problems over the years. The most common application which uses the n-gram based detection is the plagiarism detection. The other applications which use n-gram based detection are file type classification [1], and attack traffic detection [20]. There are different n-gram techniques. Some of the techniques related to this research are explained in this

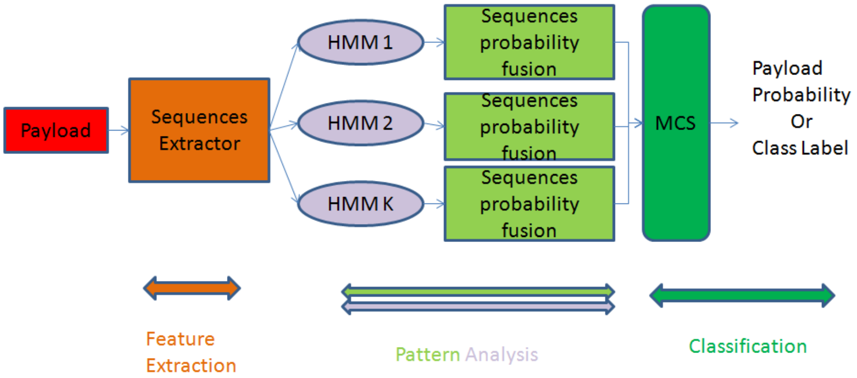

Payload
Sequences Extractor
HMM 1
HMM 2
HMM K
Sequences probability fusion
Sequences probability fusion
Sequences probability fusion
MCS
Payload Probability Or Class Label
Feature Extraction
Pattern Analysis
Classification

Figure 1: HMMPayl [2]

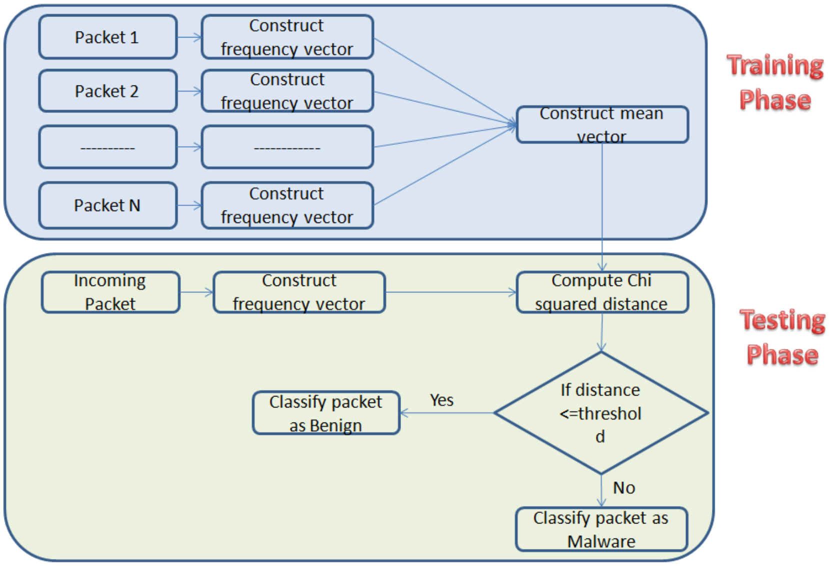
section. In order to defeat the scores in research [12], it is important to understand how the scores are calculated for the techniques.

2.3.1 χ^2 Technique

The χ^2 technique uses frequency vector as the scoring technique. Figure 2 shows the scoring method for this technique. The formula used to calculate the distance is $D^2(X,Y) = \sum_{i=1}^{N} \frac{(X_i - Y_i)^2}{Y_i}$. Since this technique uses frequency vector, when obfuscating the code, the frequency of the obfuscated bytes should be considered.

2.3.2 Ad-hoc N-Gram Technique

The Ad-hoc N-gram technique also uses frequency vector as the scoring technique. This technique is similar to the χ^2 technique, except that the formula to calculate the distance is different. The formula used to calculate the distance is $D^2(X,Y) = \sum_{i=1}^{N}(X_i - Y_i)^2$. Since this technique uses frequency vector, when obfuscating the code, the frequency of the obfuscated bytes should be considered.


Packet 1
Construct frequency vector
Packet 2
Construct frequency vector

Packet N
Construct frequency vector
Construct mean vector
Training Phase
Incoming Packet
Construct frequency vector
Compute Chi squared distance
Testing Phase
If distance <=threshol d
Yes
Classify packet as Benign
No
Classify packet as Malware

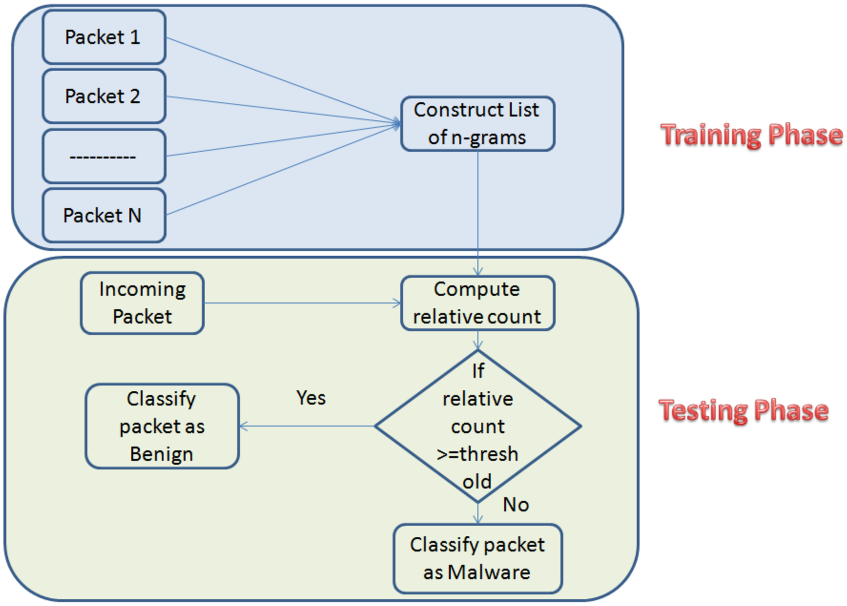
Figure 2: χ^2 Technique [12]

2.3.3 Pattern Counting Technique

Pattern counting technique is the most simple and extremely efficient technique. Pattern counting technique does not use frequency for the scoring technique. This is a binary classification technique. Figure 3 explains the scoring methodology. This technique has scored almost perfect scores for n-gram sizes 2, 3, 4 and 5.

2.4 Obfuscation Techniques

Obfuscation was identified as a transformation technique, to modify code to disguise its original form. This technique was used to hide confidential information and to make reverse engineering of application difficult. Obfuscation of applications protects the integrity of the application. Malware writers constantly identify new ways to infect victims without getting detected or at the least, increase the time before getting detected. The longer they go undetected, the higher the damage they can cause to the victims.


Packet 1
Packet 2

Packet N
Construct List of n-grams
Training Phase
Incoming Packet
Compute relative count
If relative count >=threshold
Yes
Classify packet as Benign
No
Classify packet as Malware
Testing Phase

Figure 3: Pattern Counting Technique [12]

Attackers often use obfuscation techniques to disguise the attack traffic and attack the victims [6]. Attack traffic will look similar to benign packets, as a result of obfuscation. This way, when classifying benign and attack traffic, the classification algorithm is defeated, as the attack looks more like benign, that is, there are more false negatives.

Obfuscation techniques defeat both signature based intrusion detection and anomaly based intrusion detection. In the case of signature based intrusion detection, as a result of obfuscation, the signature is altered for the attack traffic and the IDS does not detect the attack as an ‘attack’. Hence, the attack successfully infects the victim. In case of anomaly based intrusion detection, as a result of obfuscation, the statistics and algorithm to classify as benign and attack is defeated. IDS classifies ‘attack’ as ‘benign’. The obfuscation process is presented in figure 4.

Genuine benign HTTP traffic usually consists of ASCII characters. According to [1], HTTP data with non-ASCII characters indicate the presence of suspicious

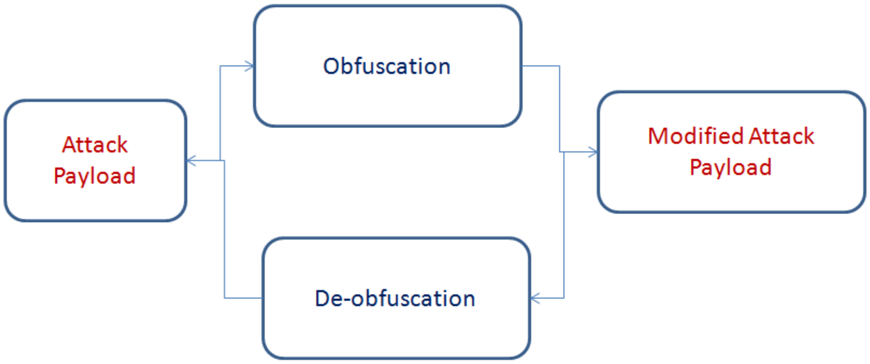


Figure 4: Obfuscation Process

executable code. Anomaly based IDS is very efficient in classifying these types of benign and attack traffic. This is because, the pattern of benign and attack are completely different. Attackers are aware of this idea and hence obfuscate attack data, to disguise attack packets to look more like benign. Obfuscation of data prevents the attack to be easily detected using the existing techniques.

There are different types of obfuscation techniques. Some of the obfuscation techniques are used in this research to analyze the robustness of different n-gram techniques. A background of these obfuscation techniques are explained in this section.

2.4.1 Insertion of dead bytes

Insertion of dead bytes [16] is one of the simple obfuscation techniques. This technique has been used by malicious code writers and application developers, to insert dead code. 'Dead Code' refers to the insertion of code that never gets executed by the application. The dead code is inserted by application developers to make reverse engineering of application difficult. Malware code writers insert dead code to prevent anti-virus programs from detecting them. Anti-virus programs often use statistics / pattern matching to detect 'abnormality' in the code. Inserting dead code gives incorrect statistics, in addition to what it is intended to do.

The attackers successfully infect the victim with this obfuscation technique as the IDS do not detect them. This obfuscation technique can be applied to HTTP payload by adding bytes from the benign traffic. The attack traffic reaches the victim system and will be able to cause the harm. We use this obfuscation technique to defeat the scores in [12].

2.4.2 Codebook Cipher

The codebook cipher [15] is a simple, yet efficient encoding/decoding technique used in cryptology. Codebooks were in the ancient times called as Morse Code and Enigma [15]. Enigma was used in World War II. The telegraph operators used the Morse code to map sentence to shorter words. A codebook can be considered as a look-up table.

The book stores the mapping of benign and malware bytes, called the code, and can be transported securely to the receiver's end. At the receiving end, the attack cipher text can be decoded to obtain the actual attack text. The codebook acts as the 'key' to encode and decode the attack packets. The attack cipher text can be decrypted only if we have access to the codebook. If there is no access to the codebook, decoding is not possible. This is a powerful obfuscation technique.

CHAPTER 3

Research Methodology

This chapter explains in detail the research methodology adopted. Different obfuscation techniques explained in Chapter 2 are implemented to the attack data packets. The n-gram detection techniques used by the authors of [12] are implemented for the obfuscated data.

3.1 Terminologies

Intrusion detection techniques often leverage machine learning to distinguish between benign and malware [5]. Machine learning techniques are most effective in detecting zero-day attacks. Using benign data, the models are developed based on different machine learning techniques such as Random Forest and Support Vector Machine (SVM). The attack data is tested against the model to classify it as either benign or attack. The scores are obtained for the attack dataset. The Receiver Operating Characteristics (ROC) curves [8] are plotted based on the true positive rate and the false positive rate.

A False positive refers to a ‘false alarm’. This means that the benign data was marked as ‘attack’. A true positive means that the attack was correctly identified as ‘attack’.

AUC [3] value of 0.9 – 1.0 is termed to be excellent. It denotes that the benign and attack are identified efficiently. In order to defeat the detection score for an existing approach, the AUC value should be less than 0.7. AUC value lesser than 0.7 denotes that the model does not efficiently distinguish between attack and benign [8].

AUC value of 0.7 means that the attack data could look similar to the benign data [8]. In order accomplish the premise, this research adopts different obfuscation techniques.

3.2 Insertion of Dead Bytes

Insertion of dead code has been one of the most common obfuscation techniques. Several researchers have adopted this technique. The dead code never gets executed in the application. It is inserted in the malware to go undetected. This approach has been used in this research. Instead of dead code, 'dead bytes' of data are inserted into the attack packets. The inserted bytes should be similar to the bytes of the benign data. Hence, benign data bytes are inserted as 'dead bytes' in the attack HTTP packets.

3.2.1 Random N-gram

Pattern counting technique is a very simple and efficient technique to implement. This technique models by creating a list of unique n-grams from the benign data packets. Here, n refers to the size of the sliding window. Attack packet data is matched for the patterns in the benign model. The score is calculated per attack packet. In order to defeat the scores, two methodologies are followed.

In the random n-gram approach, the list of random n-gram is obtained from the benign data set. n-gram value is picked randomly from the set and is appended to each of the attack packets. The random n-gram approach is shown in Figure 5. In this approach, the data appended to each of the packets varies for different n-gram sizes.

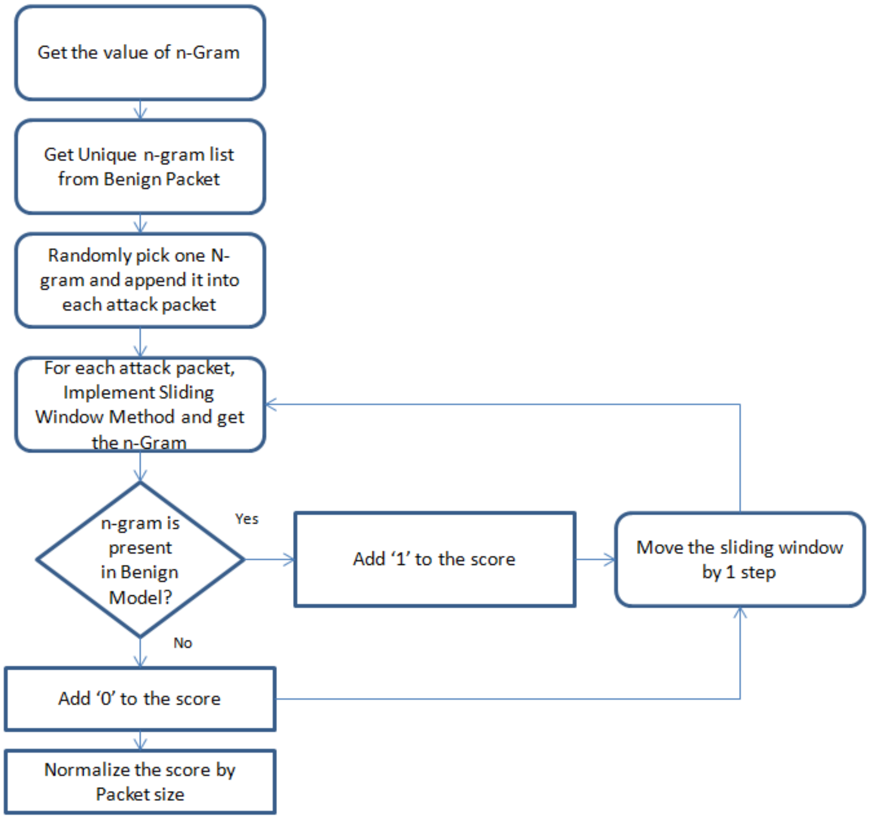

Get the value of n-Gram
Get Unique n-gram list from Benign Packet
Randomly pick one N-gram and append it into each attack packet
For each attack packet, Implement Sliding Window Method and get the n-Gram
n-gram is present in Benign Model?
Yes
Add '1' to the score
Move the sliding window by 1 step
No
Add '0' to the score
Normalize the score by Packet size

Figure 5: Random N-gram

3.2.2 Maximum Occurring N-gram

Literature shows that the detection techniques should be robust enough to detect well crafted 'intelligent' attacks. This section explains another method of obfuscating the attack data by inserting dead bytes 'intelligently'. In contrast to the random n-gram approach, a n-gram value is picked with some intelligence. Once the n-gram list is obtained in the same manner as the random n-gram approach, the different possible combinations for each n-gram are obtained. The n-gram with maximum occurrence, that is, occurrence of n-gram and the combination of n-gram, is chosen. This would increase the raw scores efficiently. The methodology is explained in Figure 6. Similar to random n-gram approach, the data appended to each of the packets varies for different n-gram sizes.

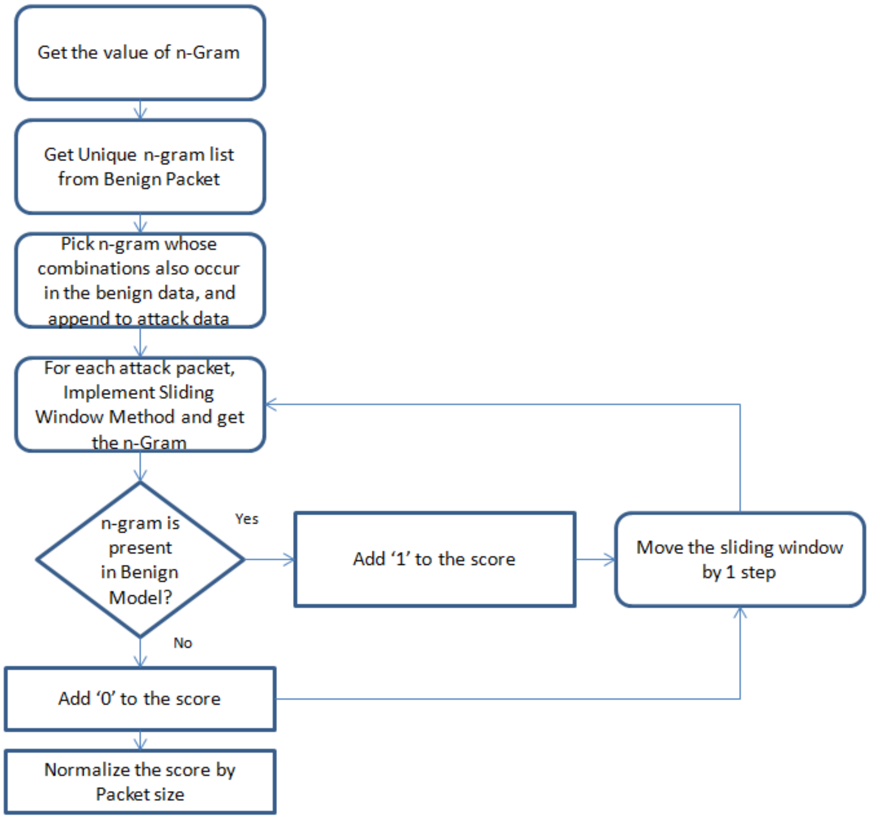

Get the value of n-Gram
Get Unique n-gram list from Benign Packet
Pick n-gram whose combinations also occur in the benign data, and append to attack data
For each attack packet, Implement Sliding Window Method and get the n-Gram
n-gram is present in Benign Model?
Yes
Add '1' to the score
Move the sliding window by 1 step
No
Add '0' to the score
Normalize the score by Packet size

Figure 6: Maximum Occurring N-Gram

The maximum occurring n-gram approach is adopted for HMMPayl method. The maximum occurring n-gram is appended to the attack set and HMMpayl is implemented. Hidden Markov Model (HMM) is developed based on benign payload. The model is tested based on the attack payload.

3.2.3 Random Benign Bytes

Ad-hoc n-gram technique and χ^2 technique consider frequency of occurrence of each bytes in the sliding window, whereas the pattern counting technique considers patterns. Hence, in order to defeat the score for the frequency techniques in [12], the methodology in Figure 7 is adopted. First, the data bytes from benign packets are obtained. Then, these bytes are appended to each of the attack packets with no change in the order of the bytes in benign data. In this approach, the data to be appended remains the same for all n-gram sizes.

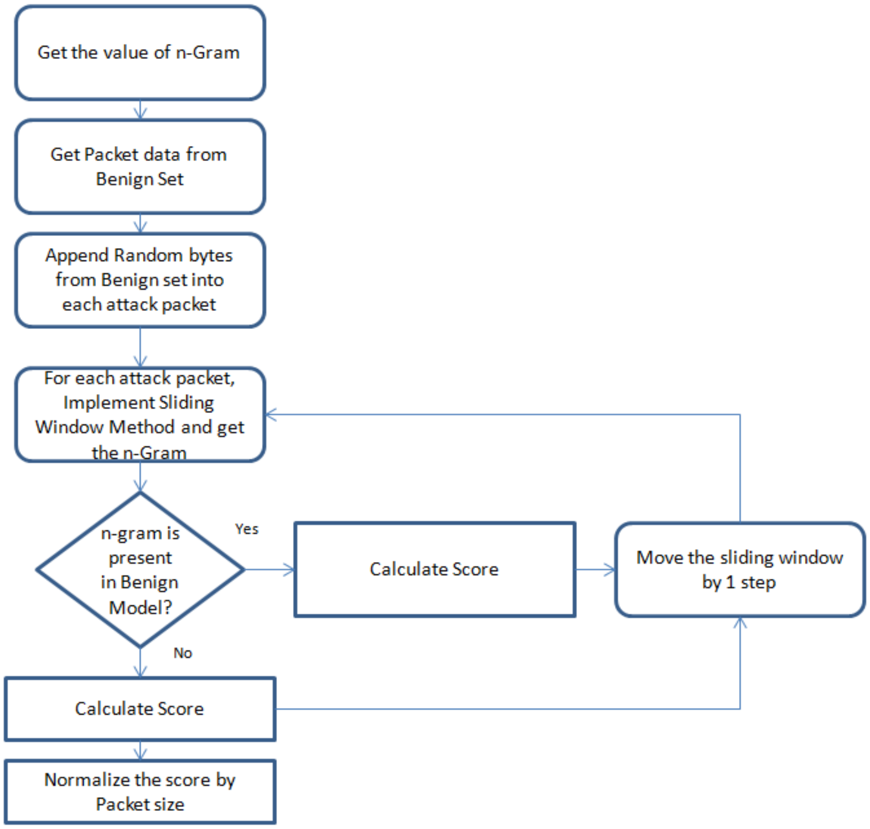

Get the value of n-Gram
Get Packet data from Benign Set
Append Random bytes from Benign set into each attack packet
For each attack packet, Implement Sliding Window Method and get the n-Gram
n-gram is present in Benign Model?
Yes
Calculate Score
Move the sliding window by 1 step
No
Calculate Score
Normalize the score by Packet size

Figure 7: Random Benign Bytes

3.3 Codebook Cipher

In Section 3.2, the length of the packet increases exponentially as the dead bytes are inserted. The n-gram techniques implemented by the authors in [12] use a normalized score based on the packet length. Hence, a more efficient way to defeat the scores is to substitute the data bytes in the payload. In this section, the research methodology explains the substitution methodology for the three n-gram techniques explained in the literature section.

As in traditional codebook cipher, the ‘mapping book’ of the substitutions are obtained. The ‘mapping book’ consists of the mapping of attack data bytes with benign data bytes. The approach is explained in Figure 8. During encoding, the codebook cipher is implemented and the attack bytes are obfuscated. While decoding, the codebook cipher is implemented with reverse lookup. The decoded packets are the attack packets. This approach is feasible due to the fact that the codebook cipher is one of the strong cryptanalysis techniques. Decoding codebook cipher is difficult and is possible only if the codebook is obtained. This is like a one-time pad [7]. Similar to random n-gram approach, for each of the n-gram sizes, a different codebook is maintained.

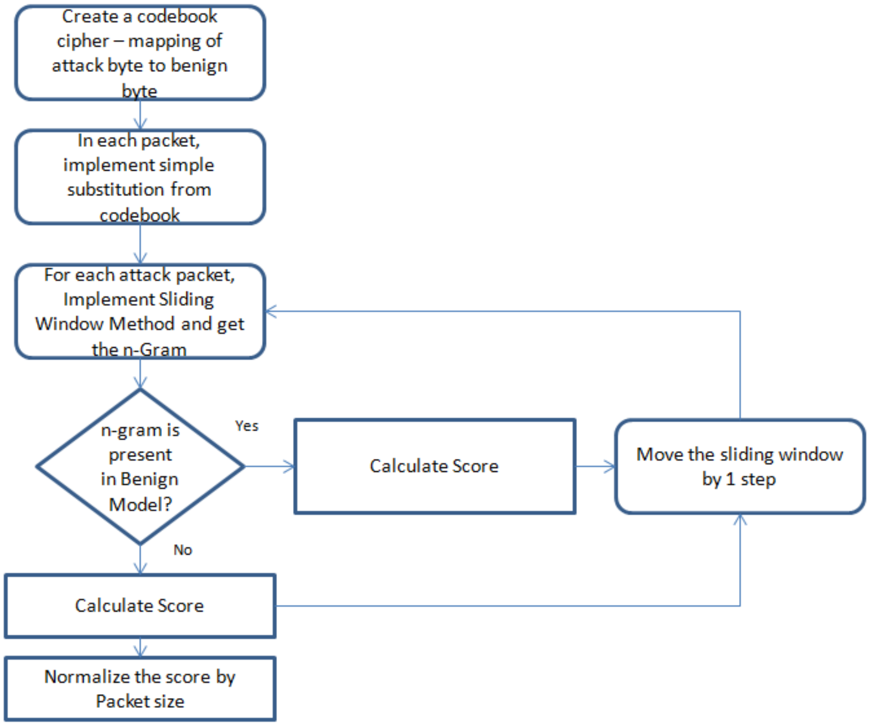

Create a codebook cipher – mapping of attack byte to benign byte
In each packet, implement simple substitution from codebook
For each attack packet, Implement Sliding Window Method and get the n-Gram
n-gram is present in Benign Model?
Yes
Calculate Score
Move the sliding window by 1 step
No
Calculate Score
Normalize the score by Packet size

Figure 8: Codebook Cipher

CHAPTER 4

Experiments and Results

The n-gram techniques described in literature were simple to implement and were able to detect attack packets with high accuracy. The techniques were based on a ‘single benign model’. In this research, we exploit the ‘single benign model’, obfuscate the attack data packets and score each attack packet using the benign model. The methodology is explained in Chapter 3. The Section 4.1 describes the benign and attack datasets used in the research to conduct experiments. The results are presented in Section 4.2.

4.1 Datasets

A variety of datasets have been used in this research to conduct the different experiments. This section provides details on the benign and attack datasets used in this research. The experiments were conducted with DARPA datasets and also real world simulated dataset. The DARPA datasets were generated by MIT Lincoln laboratory. DARPA datasets [5] have been used by many researchers for testing their models for over a decade. However, according to [4], the author has highlighted the need of more recent traffic data to build more robust models for attack detection. The author suggests that the datasets the researchers use should be a realistic dataset to model and test the detection algorithms. The recent dataset would consists of more recent malware and its variants. Understanding the requirement, the authors of [12] has used Metaspoilt [11] to simulate HTTP benign and attack traffic.

4.1.1 Benign Traffic

The first sets of experiments were conducted using DARPA benign traffic data [5]. The DARPA datasets consists of three weeks of benign traffic. For the purpose of comparison, similar to the research in [12, 2], we have used five days of training data from the DARPA datasets in this research. The details of the data set are provided in Table 2

Table 2: DARPA Benign Dataset

Day	Number of Data Packets
1	28,187
2	34,446
3	33,051
4	44,185
5	26,315

The simulated benign traffic consists of 42,128 data packets. This dataset consists of simulated traffic from wordpress [21] website. According to [12], the simulated traffic resembles original traffic and is made as realistic as possible. The details of the dataset are provided in Table 3.

Table 3: Simulated Benign Dataset

Number of Data Packets
42,128

4.1.2 Attack Traffic

We have used four types of attack datasets to test the benign model. The HTTP attack data consists of generic attacks, shellcode attacks, CLET attacks and simulated attacks. The authors of [9] have provided the generic attack dataset publicly. The generic attack dataset consists of 66 different attacks namely, denial of service, shellcode, and more. Generic attack dataset consists of ASCII bytes. Shellcode attack comprises of 11 types of attacks which injects code exploiting the buffer-overflow vulnerability. The CLET attack dataset comprises of 8 variants of shellcode attack. The details of the HTTP attack datasets are provided in Table 4.

Table 4: DARPA Attack Dataset

Attack dataset	Number of Data Packets
Generic attack	205
Shellcode attack	93
CLET attack	792

The complete attack dataset consists of four attacks. The complete attacks, related to PHP based vulnerabilities, were generated by simulating the traffic for a wordpress website. Table 5 provides details of the simulated attack datasets from the article [12]. In our experiments, the simulated benign model is tested against the complete attack dataset.

Table 5: Complete Attack Dataset [12]

Attack	Average packets per attack
WordPress OptimizePress Theme File	24
Wordpress W3 PHP code execution	55
WordPress Foxypress code execution	12
WordPress lastpostdate code execution	17

4.2 Results

In this section, we describe the different experiments conducted for each of the research methodology explained in Chapter 3. We also present the results and observations for each of the experiments. The n-gram technique and the dataset used to evaluate the models are explained for each of the methodology.

4.2.1 Random Bytes Approach

As mentioned in Chapter 2, pattern counting methodology is a simple and powerful detection technique. This n-gram technique detected the attack packets in less than 0.1ms [12] with very high accuracy and minimal processing. In this research, we analyze the robustness of the methodology by obfuscating the attack data. Obfuscation technique used in this approach is insertion of dead bytes from benign dataset.

Inserting data from benign set means that the number of attack patterns matching the benign patterns would increase. The attack packet would resemble the benign dataset even more. Experiments with varying the number of bytes to append, that is, to obfuscate the attack packets were conducted and presented in this section. The results for n-gram size 2 and 3 are shown in Figure 9 and 10.

Figure 9 show that the obfuscation of attack data defeats the score from the score in [12], for all HTTP attack packets. However, for n-gram size of 3, the scores are almost perfect even when appending 100,000 bytes to the shellcode attack dataset, from benign set. This shows that the n-gram techniques are robust.

Figure 11 shows the payload size for the shellcode attack packets, and payload size for the benign packets. It can be seen that the payload size of attack packets are higher than those of the benign packets. The benign model consists of high raw

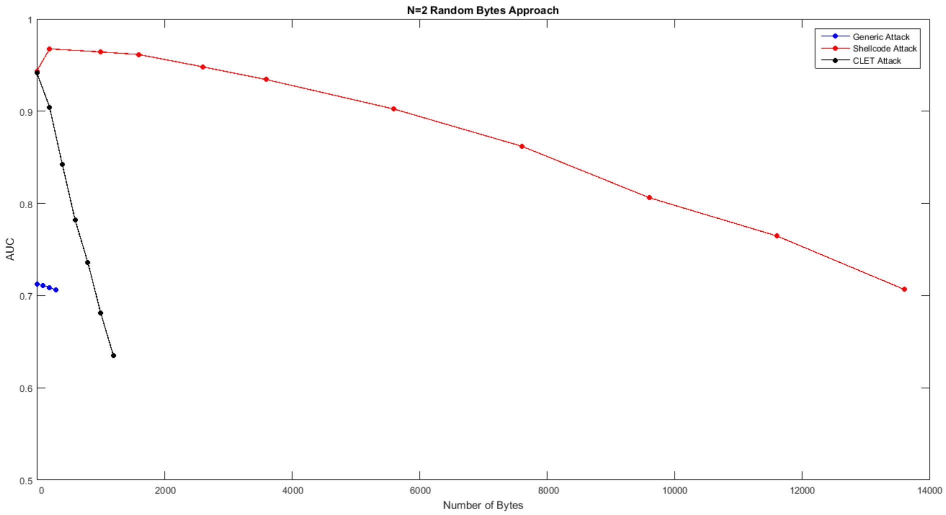


Figure 9: N=2 Random Bytes Approach Method Results

scores. This is because the scores are normalized based on the payload size. In order to defeat the attack score for this technique, when incoming attack packet is scored using the benign model, the attack score should reach the benign score. However, even when inserting large number of bytes, and the scores are normalized to the payload size, the attack scores may not reach the benign scores. Figure 12 shows the scatter plots for n-gram size of 2.

4.2.2 Maximum Occurring n-gram

Random bytes approach showed that the n-gram technique is robust and is difficult to defeat. In order to defeat the technique, a novel obfuscation technique is required. The maximum occurring n-gram technique is an 'intelligent' version of random bytes insertion technique. Instead of choosing n-gram randomly from the benign set, the n-gram to append is chosen using the methodology explained in Chapter 3. For these experiments, the DARPA benign dataset was used to model the approach. The model was evaluated for three types of attack dataset as explained in Section 4.1.

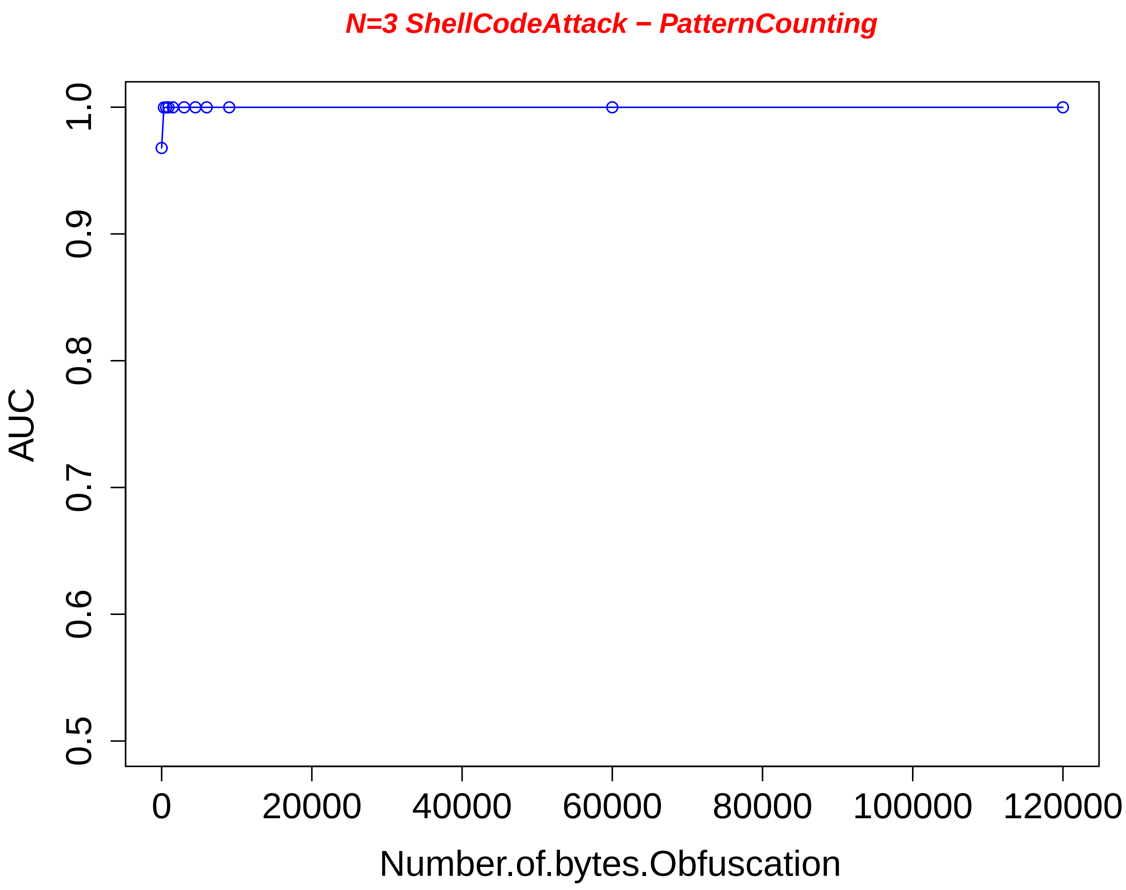


Figure 10: N=3 Shellcode Attack - Pattern Matching Technique Random Bytes Method Results

The results for this approach are summarized in this section.

Figures 13, 14 and 15 show the line plots for n-gram sizes 2–5 for generic attack data, shellcode attack data and CLET attack data respectively. The results show that the n-gram techniques are robust and the scores are difficult to defeat. However, with 'intelligent' large number of dead bytes inserted, the scores get defeated. Attackers would try to insert large number of dead bytes to go undetected. The insertion of dead bytes makes the attack packet look similar to benign packet. The reason why it is difficult to defeat the scores by inserting dead bytes, is that, the scores computed

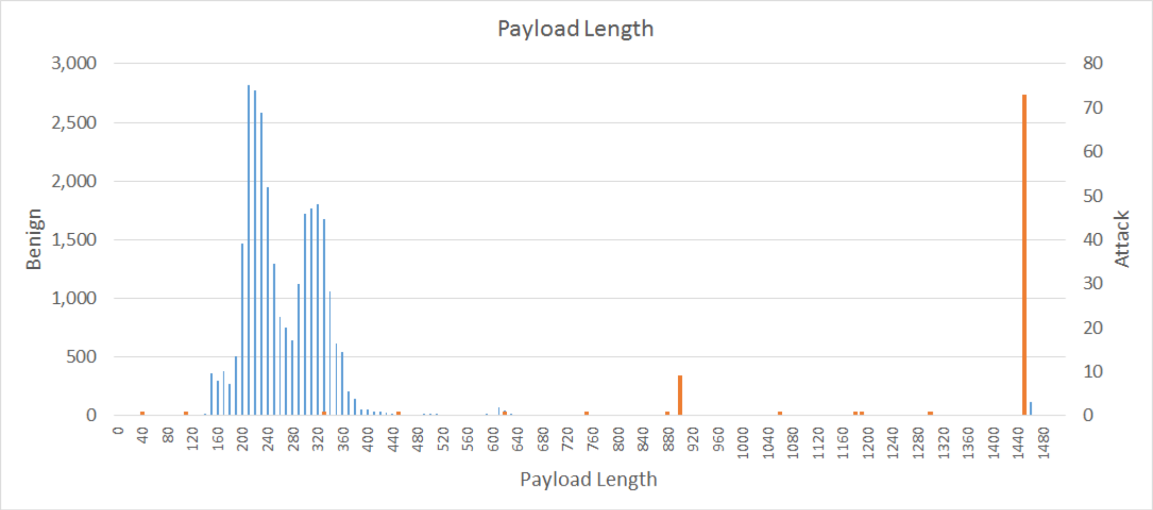


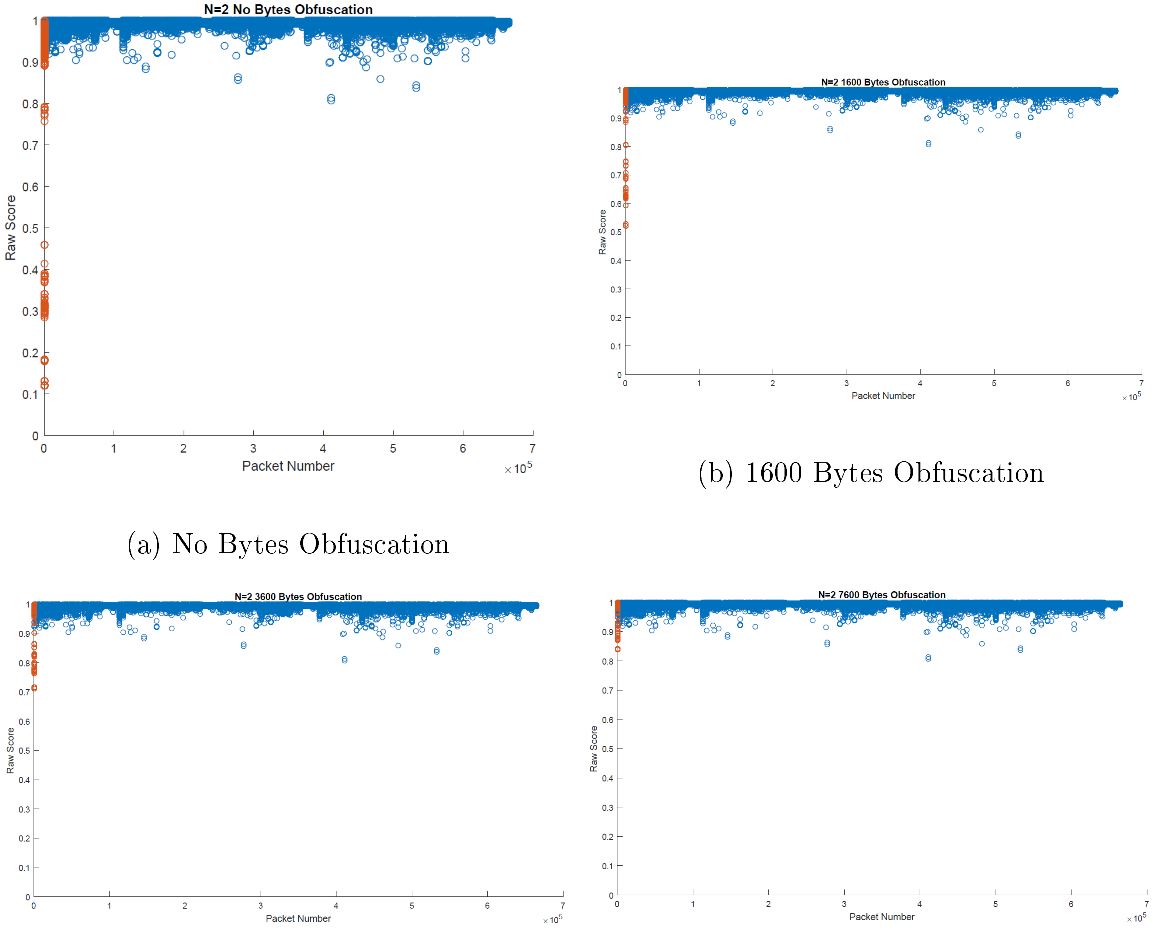
Figure 11: Payload Size

for pattern counting technique are normalized to payload size.

Table 6 shows the number of bytes required to defeat pattern counting technique scores. Figure 16 shows the 3D plot of number of bytes required to defeat the scores for each of the attack datasets for the pattern counting n-gram technique. As we recall, generic attack dataset consists of ASCII bytes, and is difficult to detect using statistical techniques. The results in our experiments show that generic attack requires lesser number of bytes to be obfuscated to the attack dataset to defeat the score. CLET attack, being the variant of the shellcode attack, requires almost the same number of bytes to obfuscate, to defeat the score.

Table 6: Number Of Dead Bytes Inserted to Defeat Pattern Counting N-gram Technique

n-gram size	Generic Attack	Shellcode Attack	CLET Attack
2	400	12,000	1,000
3	10,000	66000	72,000
4	24,000	30,000	30,000
5	16,800	25,600	25,600



(a) No Bytes Obfuscation

(b) 1600 Bytes Obfuscation

(c) 3600 Bytes Obfuscation

(d) 7600 Bytes Obfuscation

Figure 12: N=2 Random Bytes Pattern Counting Technique for varying number of bytes

Figure 17 shows the trend in the number of dead bytes to be inserted for shellcode attack. The number of dead bytes to be inserted increases for n-gram size of 3. For n-gram size 4 and 5, there is a decreasing trend. This is plausible because the number of unique n-grams increases for n-gram sizes 4 and 5. The number of patterns that match with the model increases, and hence, the number of bytes required to defeat the score decreases.

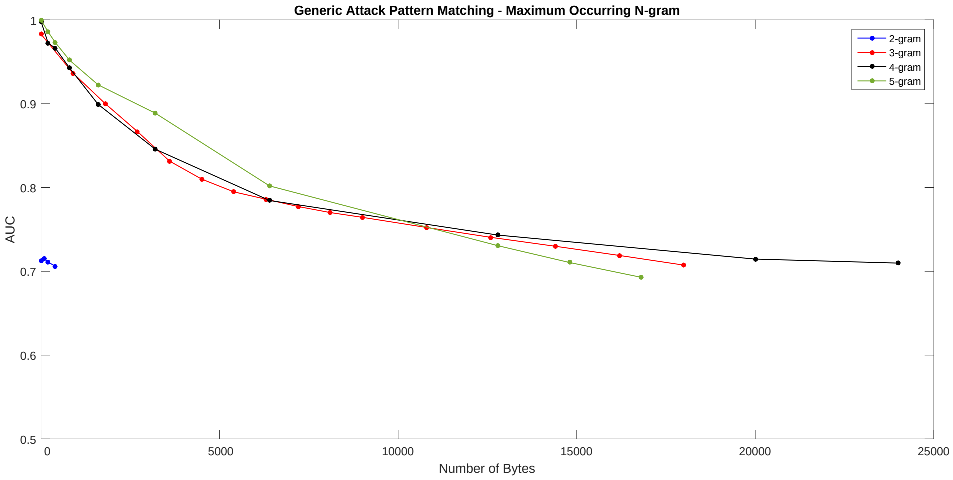


Figure 13: Generic Attack - Maximum Occurring N-Gram Approach

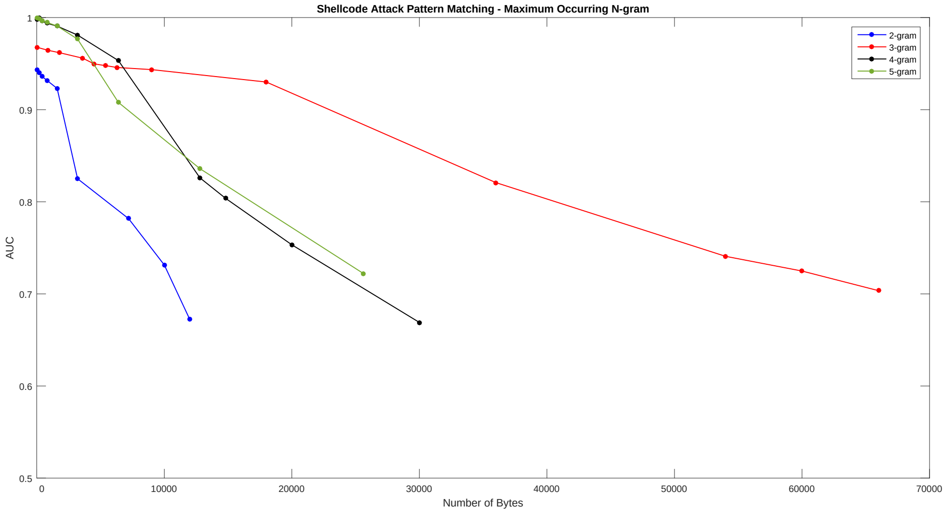


Figure 14: Shellcode Attack - Maximum Occurring N-Gram Approach

4.2.3 Random Benign Bytes

The random benign bytes methodology from Chapter 3 was applied to the DARPA dataset and experiments were conducted. The results in the research in [12] gave perfect AUC value for all n-gram sizes. The experiments with random benign

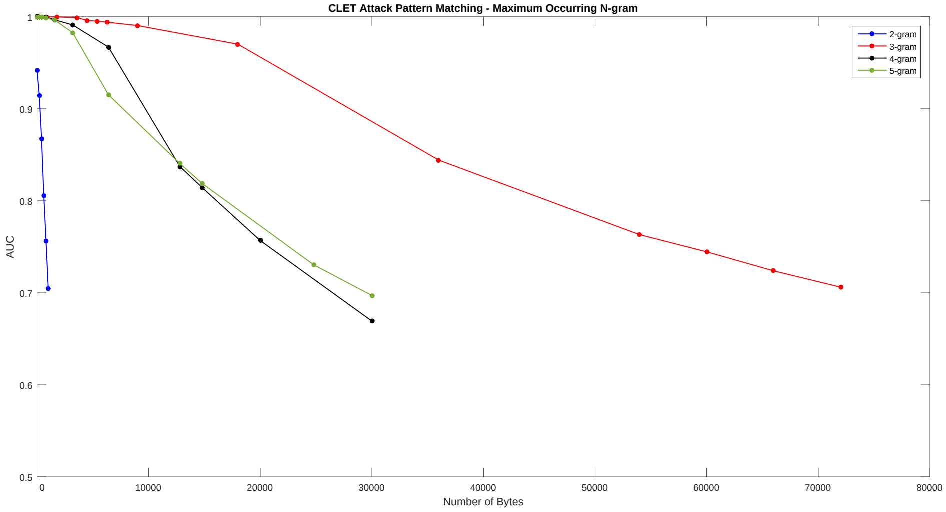


Figure 15: CLET Attack - Maximum Occurring N-Gram Approach

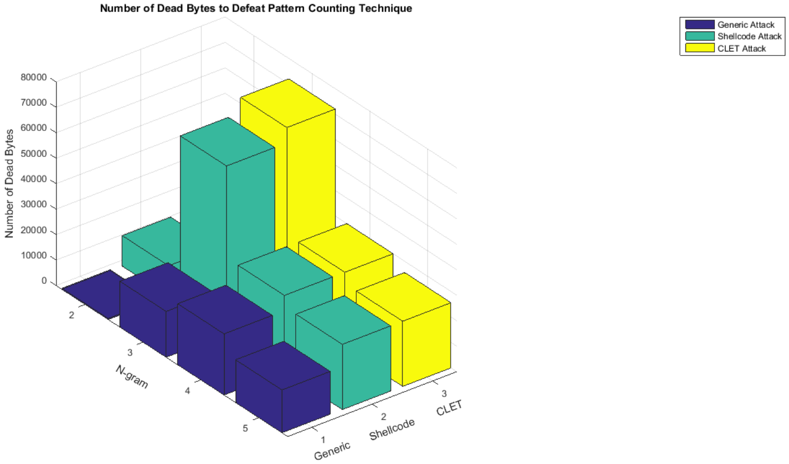


Figure 16: Number Of Dead Bytes Inserted to Defeat Pattern Counting n-gram Technique

bytes insertion were conducted for n-gram sizes 1, 2, 3, 4 and 5. Figure 18 show the result for n-gram size 3 for the generic attacks, shellcode attacks and CLET attack dataset. The figure shows that for all types of attacks, the number of bytes to be inserted are the same. This is plausible because, these n-gram techniques use frequency

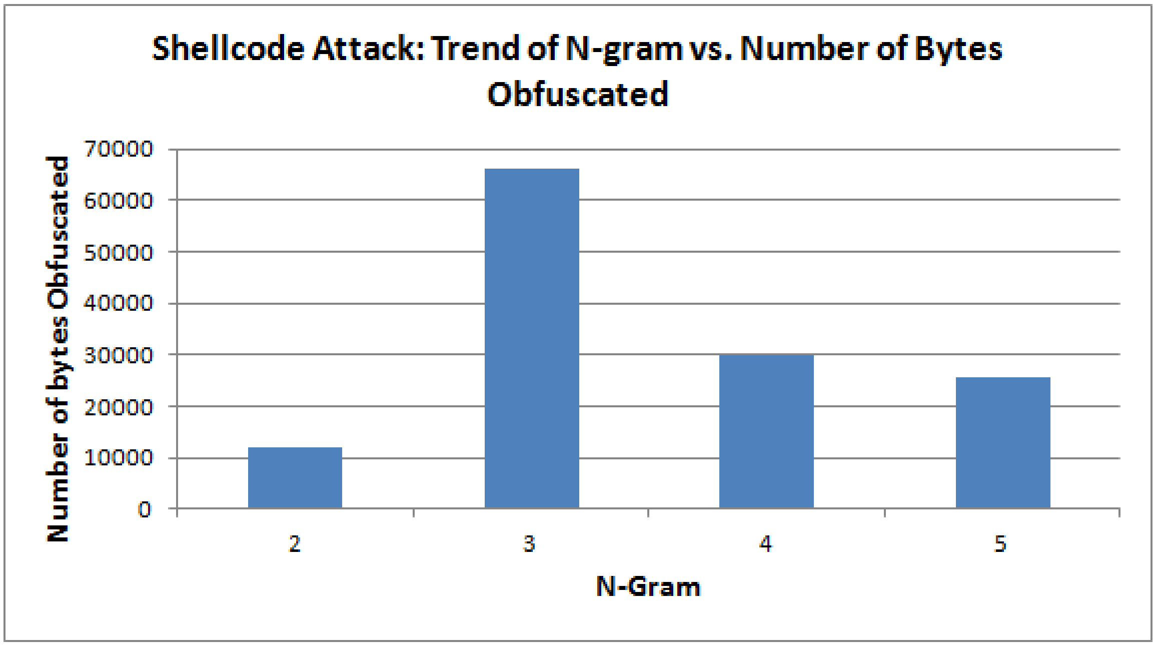


Figure 17: Shellcode Attack - Number Of Dead Bytes Inserted to Defeat Pattern Counting N-gram Technique

of occurrence n-gram in the detection approach.

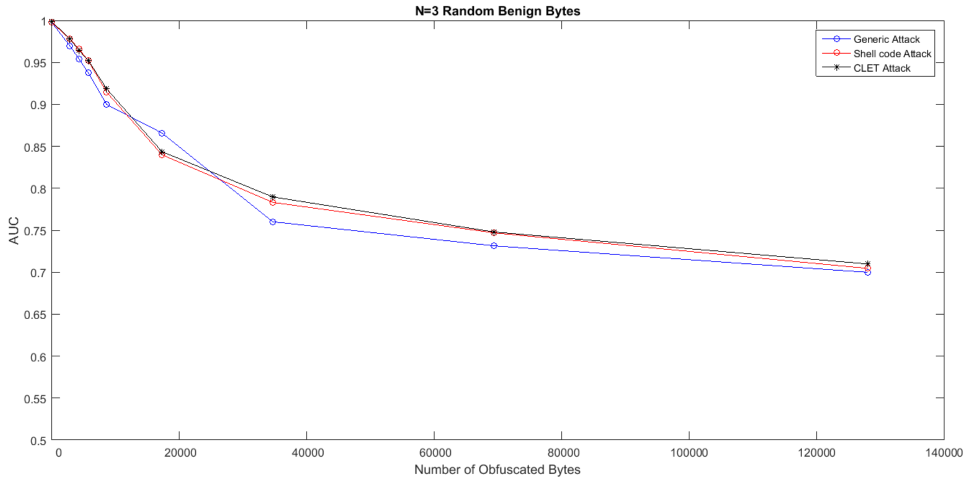


Figure 18: Random Benign Bytes

4.2.4 Codebook Cipher

As we recall, codebook is an ancient cryptanalysis technique. A codebook can be considered as a 'mapping' book, which acts as a simple substitution cipher. The codebook technique is applied to the datasets in Section 4.1. Figure 19 shows the AUC values for n-gram sizes 2, 3, 4 and 5 for the pattern matching technique.

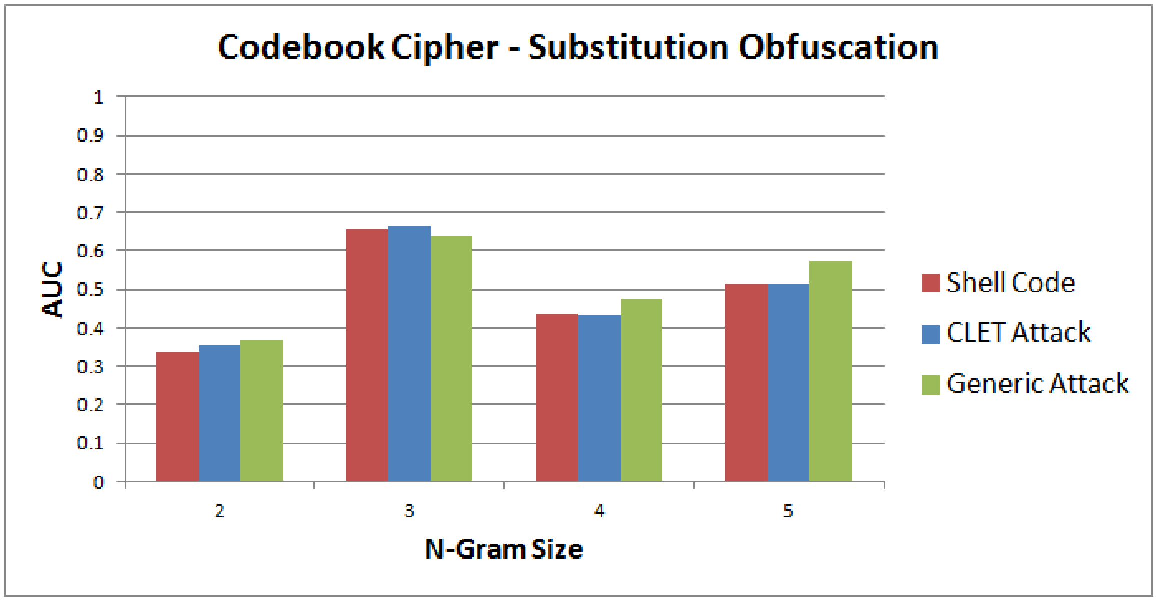


Figure 19: Codebook Cipher - Pattern Matching

The codebook cipher methodology is a substitution obfuscation technique. In this methodology, the length of the packet do not change. Hence, it is easier to defeat the n-gram techniques with this obfuscation techniques. As explained in Chapter 2, the scores for n-gram techniques are normalized based on the length of the payload. Since the substitution methodology do not increase the size of the payload, the scores are defeated easily. As an attacker, the codebook cipher obfuscation methodology is easier to implement and is efficient.

4.3 HMM-based Techniques

The HMMPayl technique explained in Chapter 2 was applied to the DARPA dataset and the experiments were conducted. The results in the research in [2] showed that there was perfect detection for all n-gram sizes. The experiments were conducted for n-gram size 2 and with 1 HMM. This case is the simple case of HMM implementation. Figure 20 shows that even for n-gram size 2, the AUC scores were not defeated, even after adding 25,000 additional bytes. This shows that the HMM-based scores are even more robust as compared to the n-gram techniques.

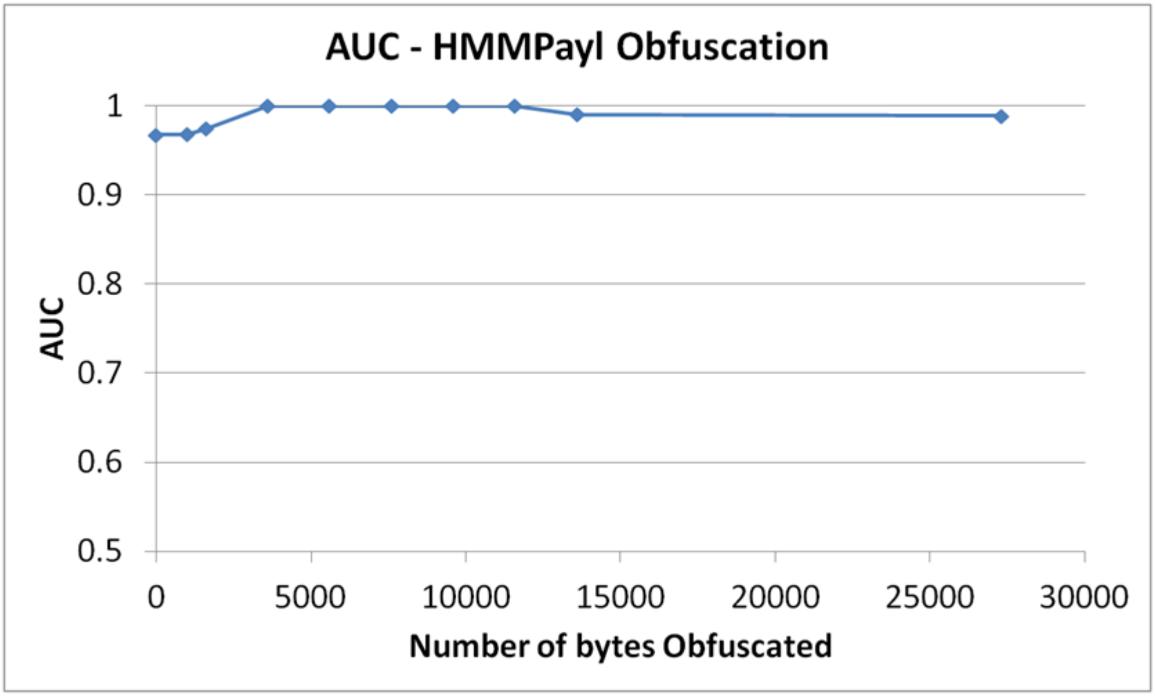


Figure 20: HMMPayl n-gram size 2 Shellcode attack

A more sophisticated multiple classifier system is adopted in the next approach. The multiple classifier system is implemented for a more accurate attack detection. Statistical methods like average, minimum, maximum, standard deviation are used as fusion classification scheme. Figure 21 shows the result with maximum rule as explained by the authors of HmmPayl [2]. It can be seen that there is no noticeable difference in the AUC between single HMM and multiple classifier system. However, sometimes when more than one HMM is used in classification, one or more HMM can perform worse compared to the others. In such cases, the multiple classifier scheme proves to be very efficient as in [2].

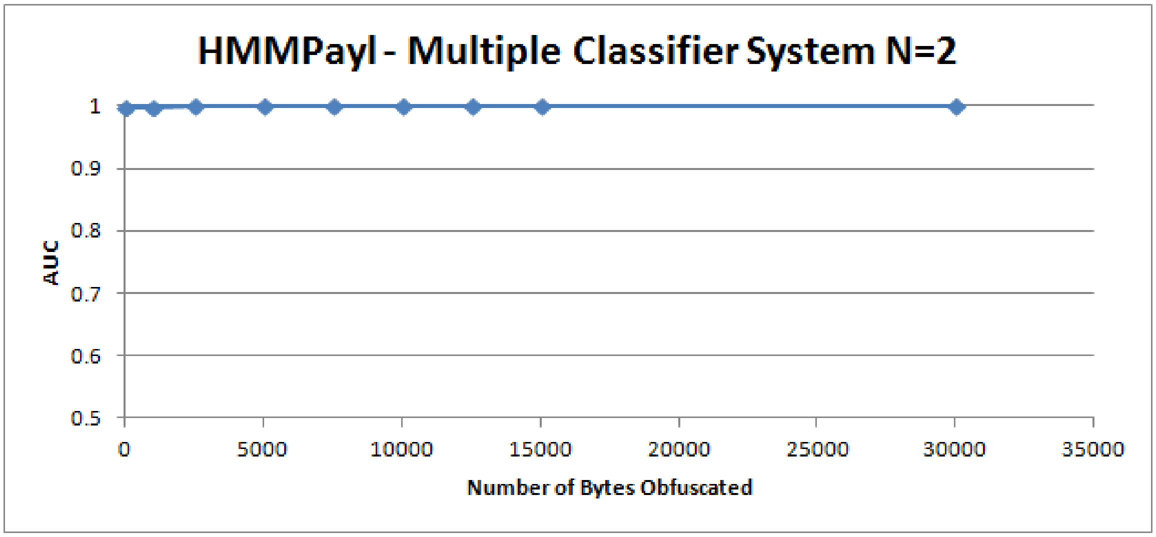


Figure 21: HMMPayl n-gram size 2 Shellcode attack - Multiple Classification System

4.4 Insertion of Obfuscation Data in Random Position

Inserting obfuscated data in random position instead of at one position can be an interesting obfuscation technique. The results in Figure 22 show that this obfuscation technique performs as comparable to the append of benign data to the attack packets. Since the pattern counting technique and chi- squared technique are both scored per packet, the results are comparable to the maximum occurring n-gram technique and random benign bytes technique.

4.5 Summary of Results

The major highlights of the results in this research are summarized in this section. Pattern counting technique is surprisingly robust to obfuscation using insertion of

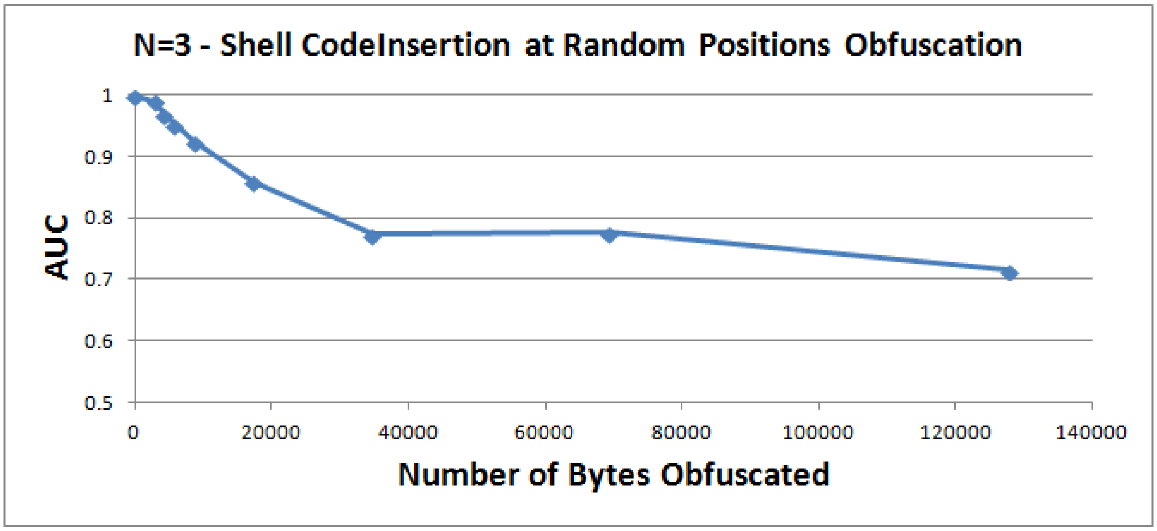


Figure 22: RandomPosition

dead bytes. This can be observed from Figure 10 which shows that the technique is robust even when 100,000 bytes are inserted. However, with 'intelligence', the technique can be defeated. Carefully identifying the bytes to be inserted, based on maximum occurrence of n-gram across all possible combinations, defeats the score. This type of obfuscation defeats generic attacks using lesser number of inserted bytes compared to shellcode attacks and CLET attacks, as shown in Figure 16. This is possible since generic attacks consists of ASCII characters, while shellcode and CLET attacks consists of non-ASCII executable bytes. Results show that even when the bytes are inserted at random positions, the scores are robust, but can be defeated. The number of bytes required to defeat shellcode and CLET attacks are the same for all n-gram sizes. This means that the attackers can obfuscate the attack and its morphed variants in a similar manner and defeat pattern counting technique.

χ^2 technique and ad-hoc n-gram technique are also robust to obfuscation using insertion of dead bytes. These techniques require the same number of obfuscated dead bytes for all types of attacks. This is because these techniques consider frequency of occurrence of n-grams as opposed to binary pattern matching of n-gram in the pattern counting technique. Moreover, these techniques are equally robust irrespective of whether the attacks consists of ASCII or non-ASCII characters.

Substitution obfuscation techniques are more efficient in terms of defeating the n-gram techniques compared to insertion obfuscation techniques. Figure 19 shows that the pattern counting technique is defeated by implementing codebook cipher as the obfuscation methodology. This is because the scores for n-gram techniques are calculated by normalizing with the payload size. In substitution techniques, the payload size do not increase. However, the number of patterns that now resemble benign patterns increase. With these observations, this research emphasizes the need to develop more robust and simple network intrusion detection techniques.

The HMM-based techniques proved the robustness and established the need for a more costlier approach for intrusion detection. The detection scores did not get defeated even for n-gram size 2. Though the implementation is complex and costlier, the efficiency of the technique is very robust.

CHAPTER 5

Conclusion and Future Work

This chapter summarizes the motivation for this research and provides an overview of the different research methodologies adapted to defeat n-gram techniques. Possible extensions to this research work will be outlined as part of the future work.

5.1 Research Summary

With the increasing threat to cybersecurity, different techniques are being evolved to detect malicious attacks. These include the less sophisticated signature-based detections to more sophisticated n-gram techniques. However, the robustness of such techniques need to be analyzed, and improved techniques should be developed. This research is aimed at analyzing the robustness and defeating the n-gram techniques as a step forward towards developing more robust detection systems.

5.2 Conclusions

The number of web applications and the number of malicious attacks exploiting the vulnerabilities are growing exponentially. Even if few of them are not detected, they can cause enormous damage. Hence, robustness in the detection techniques is a necessity. We have elaborated on two obfuscation techniques in this paper to defeat the detection techniques explained by the authors of [12]. The techniques are insertion of dead bytes and codebook cipher.

The n-gram techniques were very robust for insertion of dead bytes. It required large number of benign bytes to be appended which is not uncommon. generic attack dataset gets defeated with lesser number of benign bytes for pattern counting technique than shellcode attack dataset and CLET attack dataset. For example, the robustness can be observed for n-gram size 3 in Figure 16, which shows that it requires about 70,000 additional benign bytes to defeat the detection score of shellcode and CLET attack. For the same n-gram size with generic attack dataset, the scores get defeated with only about 10,000 benign bytes. However, malware writers could exploit the vulnerability of n-gram techniques by inserting large amounts of benign data that could go undetected. With the growth in the Internet speed over every year, the detection techniques need to be robust and efficient in detecting the packets at this speed, even after obfuscation.

The codebook cipher,which substitutes bytes instead of inserting additional bytes, is better to defeat the robust n-gram techniques. In the dead bytes insertion method, additional bytes also increase the overall payload size, thereby decreasing the effectiveness in defeating the detection scores. In the codebook cipher technique, the payload size remains the same with increased obfuscation, thus increasing the effectiveness in defeating the detection scores. Both the obfuscation techniques implemented in this research are simple and very effective. The codebook cipher methodology establishes that there is a need for more robust techniques to detect HTTP attacks.

Authors of [2] have implemented a costlier Hidden Markov Model (HMM) based technique to detect HTTP attacks. In this research we consider the benefit of this costlier detection technique based on HMM. Obfuscation techniques were be implemented and robustness of HMM technique were analyzed. The obfuscation techniques do not get defeated even for n-gram size of 2. This proves the need for a more costlier and sophisticated approach to attack detection.

5.3 Future Work

This research has considered two most common obfuscation techniques. There are many other efficient obfuscation techniques, mentioned in the literature. Malware writers often also implement sophisticated obfuscation techniques. Identifying sophisticated obfuscation techniques to defeat detection scores can be a useful direction to enhance this research. The use of a combination of substitution technique and insertion of dead bytes technique can be another direction for future research to defeat detection scores. Another area for future researchers to explore could be by inserting the bytes in random positions before the TCP segmentation is done. This will help to analyze the robustness of the n-gram techniques more accurately.

LIST OF REFERENCES

[1] I. Ahmed and K. Lhee, Classification of packet contents for malware detection, *Journal of Computer Virology*, 4(7):279–295, 2011

[2] D. Ariu, R. Tronci, and G. Giacinto, HMMPayl: An intrusion detection system based on hidden Markov models, *Computers and Security*, 30(4):221–241, 2011

[3] A. P. Bradley, The Use of the Area Under the ROC Curve in the Evolution of Machine Learning Algorithms, *Pattern Recognition*, 30:1145–1159, 1997

[4] H. Dalziel, *How to Defeat Advanced Malware*, Elsevier, 2014

[5] DARPA '99 dataset for Intrusion Detection, `http://www.ll.mit.edu/ideval/data/1999data.html`, Accessed on May 23, 2016

[6] V. DeMarines, Obfuscation - how to do it and how to crack it, *Network Security*, 2008(7):4–7, 2008

[7] Y. Dodis and J. Spencer, On the (non) universality of the one-time pad,*In Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium*, 376–385, 2002.

[8] T. Fawcett, ROC graphs: An introduction to ROC analysis, *Pattern Recognition Letters*, 27(8):861–874, 2006

[9] K. Ingham, H. Inoue, Comparing anomaly detection techniques for HTTP, *Recent Advances in Intrusion Detection*,4637: 42–62, 2007

[10] H. Mekky, R. Torres, Z. Zhang, S. Saha, A. Nucci, Detecting Malicious HTTP Redirections Using Trees of User Browsing Activity, *In INFOCOM, 2014 Proceedings IEEE*, 1159–1167, 2014

[11] Metaspoilt, Metaspoilt Penetration Testing Tool, `http://www.metasploit.com` , Accessed on May 22, 2016

[12] A. Oza, K. Ross, R. M. Low, and M. Stamp, HTTP attack detection using n-gram analysis, *Computers and Security*, 45:242–254, 2014

[13] K. Scarfone, P. Mell, Guide to Intrusion Detection and Prevention Systems, National Institute of Standards and Technology, 2007

[14] E. E. Schultz, J. Mellander and D. R. Peterson, The MS-SQL Slammer Worm, *Network Security*, 2003(3):10–14, 2003

[15] M. Stamp, *Information security: principles and practice*, John Wiley & Sons, 2011

[16] A. H. Sung, J. Xu, P. Chavez, S. Mukkamala, Static Analyzer of Vicious Executables (SAVE), *Proceedings of 20th Annual Computer Security Applications Conference (ACSAC), IEEE Computer Society Press*, 326–334, 2004

[17] Symantec, Internet Security Threat Report, April 21, 2016,
`https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf`

[18] P. G. -Teodoro, J. E. D. -Verdejo, G. M. -Fernandez, E. Vazquez, Anomaly-based network intrusion detection: Techniques, Systems and Challenges, *Computer and Security*, 28, 18–28, 2009

[19] G. Vigna, W. Robertson, D. Balzarotti, Testing Network-based Intrusion Detection Signatures Using Mutant Exploits, *In Proceedings of the 11th ACM Conference on Computer and Communications Security*,21–30, 2004

[20] K. Wang, S. Stolfo, Anomalous Payload-Based Network Intrusion Detection, *Recent Advances in Intrusion Detection*, 3224, 203 –222, 2004

[21] Wordpress, Wordpress Software,
`www.wordpress.org`, Accessed on May 23, 2016