IRSTI 55.01.01

Farikha Nauruzbayeva[1], *Assemay Imankulova*[1], *Gaukhar Seitkaliyeva*[1], *Shakhnazar-Sultan Manbay*[1]
[1]«SDU University», Kaskelen, Kazakhstan
*e-mail: 221107045@stu.sdu.edu.kz, 221107057@stu.sdu.edu.kz, 221107021@stu.sdu.edu.kz, s.manbay@sdu.edu.kz

REVIEW: EFFICIENT DEEP PACKET INSPECTION ALGORITHMS FOR INTRUSION DETECTION SYSTEMS

Abstract. This study explores the field of intrusion detection systems (IDS) and network security, with a special emphasis on the role that deep packet inspection (DPI) techniques play in protecting computer networks from a variety of online threats. Intentional intrusions into computer networks, or network attacks, can have criminal undertones. The attacker may want to take over the system, obtain sensitive data, interfere with network functions, or even take down websites and servers. The growing frequency of attacks on IT infrastructure highlights the need of intrusion detection, as per the report. It points to a gap in the body of knowledge about how well DPI algorithms work to counter vulnerabilities. The analysis highlights how bad network security is right now, including startling data on ransomware assaults, cyberattacks that target small organizations, and data breaches that cause significant financial losses. According to the report, human error is still a major weakness in network security. Intrusion detection systems (IDS) are frequently used to prevent unwanted access and safeguard systems. IDS entails identifying and evaluating system or network events in order to spot possible intrusions and stop malicious activity. Network security is emphasized as a major concern since malicious actors are constantly coming up with new ways to attack networks.

This study's main goal is to investigate and compile the body of knowledge regarding different DPI intrusion detection techniques. It suggests creating a morphological framework for IDS in order to help us comprehend these systems on a deeper level. The study provides a thorough explanation of the various DPI kinds and the algorithms that go along with them.

Keywords: Deep Packet Inspection, Medium Packet Inspection, Shallow Packet Inspection, Matching Algorithms, Intrusion Detection System, cybersecurity, malware.

I. Introduction

An intentional (perhaps criminal) intrusion into the operating system of distant or local computer networks is referred to as a network attack. The perpetrator of the attack could be an individual or a group. The hacker grants himself administrative rights using specialized tools, taking control of the system. Cybercriminals' primary objectives are to extract hidden information, disrupt the operation of websites and servers, or fully shut them down (confidential user data, texts of documents). Although James Anderson J.P. first proposed the idea of intrusion detection in 1980, it has become increasingly important in recent years as a result of ongoing attacks on IT infrastructure [1]. There is not much information on the Internet and articles about algorithms and their use in certain types of attacks. There are articles that describe separately algorithms and separate types of attacks, but there is no study of how effective these algorithms are and which of the currently known algorithms is more effective. If we talk about statistics, it's getting sadder than it could be. For example, in January 2019 alone, more than 1.76 billion corporate records leaked. Ransomware attacks occur every 14 seconds and 43% of cyberattacks target small businesses. The damages are simply amazing, for example, the average cost of a corporate data breach in 2020 will exceed $150 million. But the main vulnerability of the system remains a person[2]. In order to protect a system from unauthorized access, an intrusion detection system (IDS) is used. Intrusion detection is the process of recognizing different events occurring in a system or network, analyzing them for the potential presence of intrusion, and taking appropriate action in response to malicious activities. For many researchers and internet users, network security has emerged as their biggest challenge. This is as a result of bad users consistently introducing new forms of attacks. The data must be safeguarded by recommending a reliable intrusion detection system. Unnecessary data modifications in computer system files can be found using an intrusion detection system. There are many articles describing machine learning algorithms for an intrusion detection system, and research has been done on the types of attacks. But there is little research on the topic of deep packet analysis algorithms and their effectiveness on vulnerabilities. The main purpose of this study is to explore the existing literature on various deep packet inspection algorithms for intrusion detection, to systematize the intrusion detection system (IDS), to develop a morphological framework for IDS to facilitate understanding.

This study provides a detailed overview of the types of the deep packet inspection and its algorithms. The rest of the paper is organized as follows: Introduction and related concepts about Firewall are discussed in section 1.

Section 2 presents Packet Inspection Techniques, thus providing a global-view for readers to better understand this field, whereas Section 3 is about algorithms and their realizations. In section 4 we presented Algorithms of DPI and in section 5 is about methods and methodology of research. Also section 6 contains Results and Discussion, whereas section 7 contains Conclusion.

II. Firewall

As the Internet continues to grow at an incredible rate, such that it has permeated every life of modern man. New inventions never cease to amaze. But with this there is a big risk, as it increases the likelihood of vulnerabilities in more complex software and network systems. In recent years, much attention has been paid to the method of preventing attacks[4].

An instrument for security protection in the area of computer network security is the firewall. Between intranet and extranet, it is utilized. The initial one is acknowledged as a secure network. The latter network is noted as being substantially less secure. Hardware and software make up a firewall. Only through a firewall can communication between an intranet and an extranet occur. The primary service mechanism that ensures network information security is a firewall. It effectively defends. At the same time, by taking over the security policies (permission, rejection, monitoring, etc.), the flow of information into and out of the network can be exposed and intercepted. A firewall serves as an analyst, understanding how to evaluate information flow additional separator. System can filter the stream of parsed data using a separator. Additionally, it acts as a restrictor by limiting the transmission of information that is deemed risky, such as viruses and other harmful software. Deny access to the intranet while allowing safe information flow there. So, it acts as guards and defend servers as shows in Figure 1. Give your approval for a secure information flow to the intranet. As a result, network security can be successfully protected. Ensure the intranet's safety [5].

In other words, firewalls guard against both planned and accidental network attacks. Using the rules set forth in the filtering configuration and other protocols, firewalls serve as the network's gatekeepers, allowing or disallowing packets to enter or leave the network. As a result, the firewall inspects each packet that enters or leaves the network for specific attributes, discarding it if it does not match the filtering settings and accepting it otherwise [6].

A cyberattack can be directed by individuals, communities, states, or even from an anonymous source. Hackers usually carry out network attacks to change, steal or corrupt personal data. At the moment, according to the authors

of the article[7], the best and most effective methods of dealing with such types of threats are Intrusion Detection Systems (IDS).

An IDS is a software application or hardware device that monitors traffic for malicious activity or policy violations.

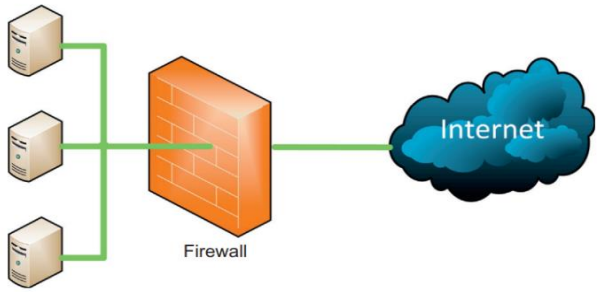

Figure 1. How Firewall works

In addition, IDS is designed to be deployed in a variety of environments, also divided into a host and network detection system. The host intrusion detection system(HIDS) is installed on the client computer, in turn, the network intrusion detection system(NIDS) is placed on the network. Recently, IDS have been based on deep learning and have shown good results, which means that they are effective [7].

System administrators can be alerted and detected by intrusion detection systems, but malicious traffic cannot be blocked or rejected. Systems for intrusion prevention can obstruct traffic. A device called Deep Packet Inspection (DPI) may decode network traffic and display its payload or content. To identify attack indicators like DDoS attacks or malware, Deep Packet Inspection is frequently used by Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), Advanced Firewalls, and many other specialist cybersecurity solutions [8].

III. Packet Inspection Techniques

The packet inspection technologies that have been used in network environments can be divided into three classes: shallow packet inspection, medium packet inspection, and deep packet inspection [9].

Packet inspection techniques are useful in terms of classifying and identifying network traffic. Currently, packet inspection methods are divided into three types: shallow packet inspection, medium packet inspection, and deep packet inspection. Normal packet inspection includes inspection based on network and/or transport layer headers. The only packet inspection method that uses normal packet payload inspection as well as packet header inspection.

Deep Packet Inspection can perform tasks at all seven layers of the Open Systems Interconnection (OSI) model. With the exception of Deep Packet Inspection, the remaining two methods, Shallow Packet Inspection and Medium Packet Inspection, do not inspect the packet payload and cannot work at all levels of the Open Systems Interconnection (OSI) model. In addition, it should be noted that SPI and MPI are examples of a static packet filtering method, while DPI is an example of a dynamic packet filtering method [9].

In this section, we will briefly discuss these three package inspection methods. Figure 2 shows level of inspection for SPI, MPI and DPI in OSI model.

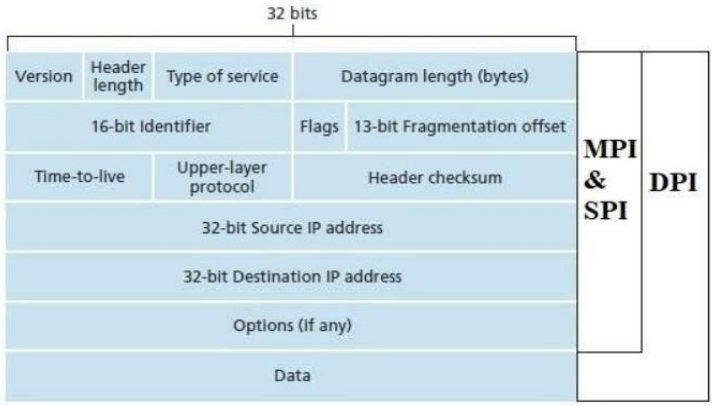

Figure 2. Level of inspection for SPI, MPI and DPI in OSI model

III.A Shallow Packet Inspection (SPI)

Shallow packet inspection relies solely on checking network packet headers and detecting suspicious or malicious packets and statistical analysis. Due to the low requirements for computing resources, SPI can analyze large amounts of passing traffic. Shallow data validation does not reveal the contents of the package, making SPI the lightest of the three data validation methods. This technology is one of the first methods for checking data flows based on the information that was embedded in the packet headers, in most cases on IP addresses, as well as on the ports of the corresponding network applications. Therefore, packet sniffing is only used to discover the source and destination IP addresses, destination port number, source port number, and protocol name of an incoming data packet. Using the port number of the data packets, this technology can also identify the application name to compare with the application name for that particular port, and in cases of a match, the SPI protects that the traffic is safe, otherwise the packet is considered malicious. This technology is useless nowadays because the port numbers can be changed randomly, which confuses this method and it can no longer give accurate results.

In addition, packet inspection is performed only at the data link and physical layers of the Open Systems Interconnection (OSI) model [9][10].

III.B Medium Packet Inspection (MPI)

Mid-packet inspection is accomplished using specialized software operating on intermediate nodes, also known as intermediate blocks, distributed throughout the network and able to capture header information as well as some of the data payload. The government might utilize this technique for surveillance. The check is not carried out at all levels of the Open Systems Interconnection (OSI) model, but only at the transport, network, channel and physical levels. Using Medium Packet Inspection, you can restrict network users from downloading infected videos, photos, music, etc. over the Internet [9].

This technology acts as an intermediary between client computers and the Internet. Figure 3 offers a visual example of how this might appear in a network [11].

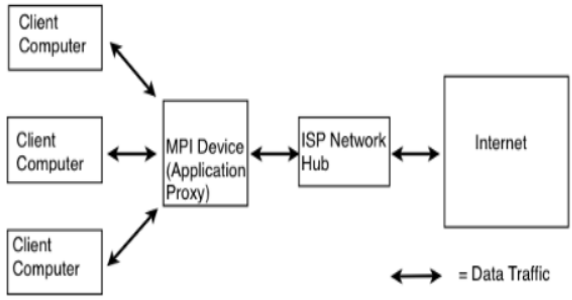

Figure 3. MPI Device Inline with Network Routing Equipment

III.C Deep Packet Inspection (DPI)

As mentioned above, deep packet inspection is used to analyze data packets and performs inspection on all layers of the model's operating system. Deep Packet Inspection firewalls add another layer of intelligence to our firewall capabilities.Because they are able to analyze traffic content. While simple firewalls can only track the structure of the network traffic itself [12].

According to Thaksen J.Parvat and Pravin Chandra [13] the packet consists of five tuples (source IP address, destination IP address, source port, destination port, and network layer protocol). There are hardware and software solutions to optimize and improve performance. The latter represents efficient algorithms and high-speed matching.Misclassification can lead to false positive or false negative results. At this point, the correct method guarantees an accurate

result. All methods are based on ports, which vary in many applications. The perfect string match is based on the sequence of characters or numbers in the payload. Based on numerical methods, the arithmetic properties of multiple packets are used to detect data changes between hosts. Behavioral analysis and heuristics are often combined to provide evaluation options. Some of the biggest DPI shortcomings are building a signature library, achieving fast, low-latency response, and characterizing network activity in detail. For deep packet inspection classifier, high and accurate classification is important. The three tasks of packet classification are single field classification, multiple field classification, and deep packet classification (DPC) Boyer-Moore string search algorithms and their modifications perform fast packet classification. The average time complexity ranges from O(n) to O(min) [13]. In the Figure 4 provides a visual representation of the depth of packet inspection at different layers of the network.

In the past, security solutions employing Deep Packet Inspection (DPI), such as proxies and firewalls, primarily concentrated on the lower layers of network packets, specifically the header at Layer 3 of the OSI model. However, due to the growing complexity of attacks, intrusion detection analysts now need to direct their attention to all pertinent layers of the network packet, encompassing Layers 2 through 7 of the OSI model.[14]

IV. Algorithms of DPI

IV.A Regular Expression Matching Algorithm

This algorithm is used in the Snort and Bro network intrusion detection systems (NIDS) and the L7-filter in Linux.

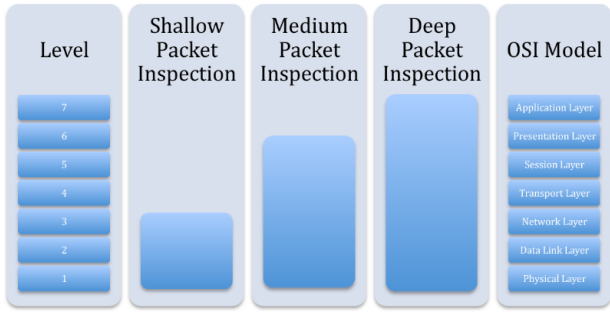

Figure 4. Packet inspection depth in OSI model

Problem Scale	DFA States	Memory Cost	Solutions	Performance
Simple	10K	10MB	No need	Tens of Gbps
Medium	100K	100MB	Compression	about 10 Gbps
Large	1M	1GB	Compression or scalable FAs	about 1 Gbps
Huge	Infeasible		Scalable FAs	about 0.1 Gbps

TABLE I
FSM solutions and results

Finite State Machine (FSM, also called State Machine) is a regular expression language state machine. This machine consists of several nodes, called states, and several labeled directed edges connecting the nodes. There are also two states ”Initial” and ”Acceptance.” The principle of operation is as follows: it starts with the initial state and reads the first byte of the payload then goes to the next state according to the labels on the edges. And in the end, when accepting states are reached, the payload and matching signatures are assumed to match [15].

In Table I, we present a comparison of research papers discussing Finite State Machine (FSM) solutions for regular expression matching. This algorithm finds applications in various network intrusion detection systems, including Snort and Bro, as well as the L7-filter in Linux. The table showcases different problem scales, corresponding Deterministic Finite Automaton (DFA) states, memory cost, solutions employed, and performance metrics for FSM-based solutions.

This automaton-based pattern matching is widely used in many DPI applications these days. FSM has two types: deterministic and nondeterministic finite automata (DFA, NFA), they are the same in strength of expression but are handled differently, dfa has only one transition and only one active state, while NFA has numerous transitions and numerous active states. Also, the time complexity can be seen in Table II.

	Compile m regular expressions into m FAs		compile m regular expressions into an integrated FA	
	Processing complexity	Storage cost	Processing complexity	Storage cost
NFA	$O(mn^2)$	$O(mn)$	$O(mn^2)$	$O(mn)$
DFA	$O(m)$	$O(m2^n)$	$O(1)$	$O(2^{mn})$

TABLE II
Time and space complexity comparisons of NFA and DFA in various strategies

IV.B Aho-corasick algorithm

Alfred Aho and Margaret Korasik created the method in 1975 to five to ten times speed up a library’s bibliographic search program [16]. This technique

effectively locates every instance of any one of a limited number of keywords within a text stream. The utilization of the boron data structure and the creation of a finite deterministic automaton on its foundation form the basis of the algorithm. It is crucial to keep in mind that locating a substring within a string can be easily accomplished in quadratic time, hence it is crucial for the efficient functioning of all Aho-Korasik components that none of them asymptotically surpass the line with regard to the length of the strings [17] [18]. The authors of this article [16] explain the process as in Figure 5.

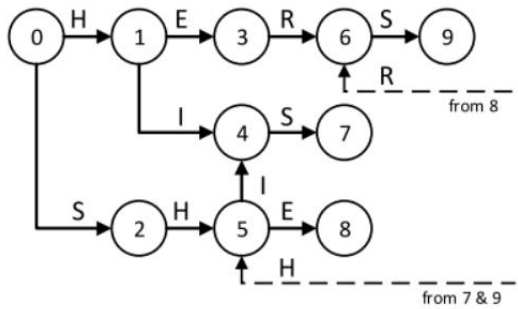

Figure 5. Example DFA for patterns “SHE”, “HERS” and “HIS”. Failure transitions are omitted for clarity

The graphic serves as a DFA example for the ”SHE,” ”HER,” and ”HIS” templates. In this case, the IDS is required to intercept any built-in templates that may have occurred after the template was approved. To avoid using phony templates, this is done. In the figure, dashed lines may be seen. They are simply the longest prefix match jumps, tedious to simplify the circuit. For example, the device starts from state 0 and uses ”SH”, in this case it is the input character and then goes to state 5. Then if you enter the character ”I”, the DFA goes from state 5 to state 4, so the system most likely assumes that the word is ”hello”. But if instead of the previous pattern, the input is ”R” after ”SH”, the DFA transitions from state 5 to state 0, since there is no such pattern starting with ”SHR”. For each state in DFA, there are specific transitions for each character in the ASCII alphabet. Algorithm for pattern matching machine:
INPUT: a text string

$$x = a1, a2. . . an$$

where each

$$ai$$

in this case is an input symbol and template matching machine with goto function g, failure function f. OUTPUT: Locations at which keywords occur in x [16].

Algorithm 1. *Aho-corasick algorithm realization*

```
1. begin
2. state ← 0
3. for i ← 1 until n do
4. begin
5. while g (state, ai) = fail do state ← f(state)
6. state ← g (state, ai)
7. if output (state) ≠ empty then
8. begin
9. print i
10. print output (state)
11. end
12. end
13. end
```

IV.C Wu-Manber Algorithm

This algorithm is used to search for multiple patterns in parallel; it is faster than any other algorithms and supports a huge number of patterns. This algorithm is used in many applications related to traffic filtering to search for selected patterns and security applications to detect certain words that may be suspicious.The algorithm consists of several stages. The first stage includes pre-processing of pattern sets.

According to this article [19], applications benefit from the fact that they use a fixed set of templates. Three tables are created at this stage: the SHIFT table, the HASH table, and the PREFIX table. The SHIFT table is used to determine the number of characters in the text that can be shifted (skip) when parsing the text. If the shift value is 0, then the other two tables HASH and PREFIX are used in this case. These tables recognize which pattern matches and whether a match check is needed. deciding which pattern is a candidate for a match, and to test for a match. This algorithm has limitations; it requires patterns to be of the same length, is less efficient, and cannot handle more than a few hundred patterns [20].

In the authors' implementation of this article [19], we set the size of 3 tables to 215 = 32768. Running the algorithm for preprocessing (an empty file) took only 0.16 seconds for more than 9000 templates.

Algorithm 2. *The pseudocode of Wu-Manber Algorithm. Partial code for the main loop*

```
1. while (text <=textend) {
2. h = (*text<<Hbits)+(*(text-1)): /* The hash function (we use Hbits=5) */
3. if (Long) h = (h<<Hbits)+(*(text-2)): /* Long=1 when the number of patterns warrants B=3 */
4. shift = SHIFT[h]: /* we use a SHIFT table of size 215 = 32768 */
5. if (shift == 0) {/* "h=h & mask_hash" can be used here if HASH is smaller then SHIFT */
6. text_prefix = (*(text-m+1)<<8) + *(text-m+2);
7. p = HASH[h]:
8. p_end = HASH[b+1];
9. while (p++ < p_end) {/* loop through all patterns that hash to the same value (h) */
10. if(text_prefix != PREFIX[pl) continue;
11. px = PAT_POINT[p];
12. qx = text-m+1;
13. while (*(px++) = *(qx++)); /* check the text against the pattern directly */
14. if (*(px-1) == 0) {/* 0 indicates the end of a string */
15. report a match
16. }
17. shift = 1;
18. }
19. text += shift;
20. }
```

IV.D p ^{3}FSM Algorithm

p^{3}FSM is one of the fastest compact and fully programmable pattern matching engines. Although it mimics very similar DFA-based hardware pattern matching properties, it is based on software.

The system uses DFA to maintain deterministic performance, however DFA is stored in a new way so that minimal memory is required. P3FSM achieves these properties by maintaining only one memory entry for each state in the DFA, rather than maintaining multiple DFA transitions as in typical memory-based DFA implementations. The code makes a prediction due to the fact that it contains all possible subsequent states. p3FSM consists of three parts: pattern analysis, state analysis, and FSM components. One of the downsides of DFA is that it requires a lot of memory to store. p3FSM solves this problem by keeping only one code per state in the DFA. From these status codes, the FSM

is formed, which ensures fast system operation. The result is a pattern matching engine that combines the benefits of both hardware and software systems.

The p^3FSM is fast, memory efficient, easy to program and completely portable [21].

***Algorithm 3**. Pseudocode for generating state code (p3FSM Algorithm)*
Input: State Trans. Table: T
Result: Indep. Group Set R
1. Search_Indep_Group (T)
2. begin
3. R = 0;
4. Graph(SDFA, CA);
5. while |G'j| ≠ Ø do
6. c = MaxClique(G'j) :
7. add c to set R;
8. remove c from G'j;
9. end
10. end
11. Graph (bool SDFA, bool CA)
12. begin
13. T'- T;
14. if SDFA then
15. T' = SDFA(T)
16. end
17. G - StateCirouping(T') :
18. G'= Create_Graph（G） :
19. G'j = G';
20. if CA then
21. G'j = Character_Aware(G')
22. end
23. end
24. Create Graph (G)
25. begin
26. while G≠Ø do
27. remove a group gi: from G;
28. add a vertex vi, for group gi on G' ;
29. for vertex Vvj, on G' do
30. if gj ⌒gi. = Ø then

31. add a edge between vj and vi
32. end
33. end
34. end
35. end
36. Character_Aware (G')
37. begin
38. for each char ci in G'j do
39. for other chars cj, in G'j do
40. NC = non-complete sub-graphs for nodes(ci,cj);
41. remove all edges in NC from G'j;
42. end
43. end
44. end

IV.E BOM Algorithm

According to this research work [22], some kinds of algorithms such as Boyer-Moore look for pattern suffix ratios, but it is also possible to match some pattern prefixes by looking for a window character from right to left and then increasing the length of the shifts, all this is possible through the use of the reverse pattern oracle suffix. This data structure is a very compact automaton that recognizes at least all the suffixes of a word and a little more than other words. A string matching algorithm that uses a reverse pattern oracle is called a reverse oracle matching algorithm. During the search phase, the Backward Oracle Matching algorithm analyzes the window symbols from right to left with O(xR) automaton starting from the state q0. The process continues until no more transitions are defined. At this moment, the length of the longest template prefix, which is the suffix of the scanned part of the text, is less than the length of the O(xR) path from the initial state q0 to the last encountered final state. Knowing this length, it is easy to calculate the length of the shift to be performed.

The Backward Oracle Matching algorithm has quadratic time complexity in the worst case, but is optimal on average. On average, it performs O(n.(logm) / m) text character checks, reaching the best bound shown by Yao in 1979 [23].

Algorithm 4. *Pseudocode of BOM Algorithm*

BOM(p = $p_1p_2…p_m$、 T=t_1 t_2 t…n)

1. Pre-processing
2. Construction of the oracle of pr

```
3. Search
4. pos ← 0
5. While (pos <= n - m)  do
6. state + initial state of Oracle(pr)
7. j ← m
8. While state exists do
9. state ← image state by T[pos + j] in Oracle(pr)
10. j←i—1
11. EndWhile
12. If  j = 0 do
13. mark an occurrence at pos + 1
14. j←1
15. Endif
16. pos ← pos + j
17. EndWhile
```

V. Methodology

V.A Data Collection and Source

Our research is focused on the comprehensive evaluation of Deep Packet Inspection (DPI) algorithms to mitigate threats like malware, viruses, and other malicious entities in network traffic. The primary dataset was sourced from the reputable Canadian Institute of Cybersecurity website. This institution has prepared an extensive collection of logs associated with different malware types, giving us the foundation for our investigation. For a visual representation of our research approach, please refer to Figure 6.

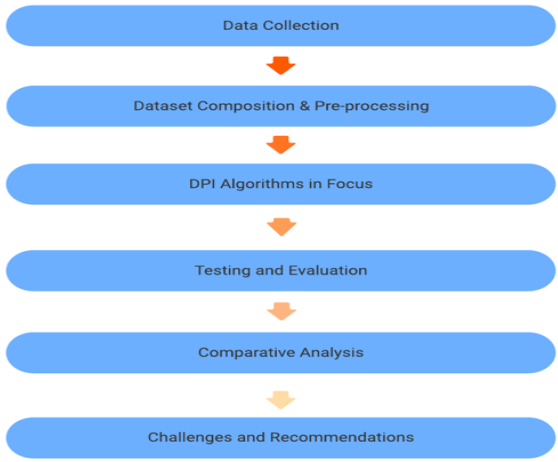

Figure 6. Flowchart of the DPI evaluation methodology from data sourcing to

recommendations.

V.B Dataset Composition and Pre-processing

The chosen dataset offers insights into approximately 50 malware families. Detailed across more than 250 columns and encompassing over 15,000 rows, the data captures diverse data points from varied geographical locations. The information, while stored as binary code in the system and available in csv format, required intensive pre-processing. Using Python, combined with specialized data science libraries, we undertook a complex decoding and cleaning process to ensure the data's accuracy and usability.

V.C DPI Algorithms in Focus

Our central objective revolved around exploring the efficiency of several DPI-based pattern matching algorithms. The algorithms at the forefront of our study included:

Aho-corasick - Aho-Corasick, known for its swift multi-pattern string searches, Subsequently, the resulting FSM has a root state indicating that no string has been matched or partially matched, with all pattern characters enumerated from the root.[24].

Wu-Manber - Famed for its ability to search multiple patterns simultaneously. p_3FSM - A unique combination of software and hardware attributes, ensuring deterministic performance. Backward Oracle Matching - Recognized for its optimal average time complexity.

V.D Testing and Evaluation

Each algorithm was subjected to exhaustive testing. Our benchmarks were set high, focusing on crucial performance metrics such as:

Detection speed: The rate at which each algorithm identified threats.

Accuracy: The precision with which genuine threats were pinpointed, minimizing false positives. Memory usage: Understanding the computational resources consumed, ensuring that the algorithm remains feasible for real-world applications.

V.E. Comparative Analysis

After gathering results, a side-by-side comparison provided insights into the relative merits and demerits of each algorithm. This helped in discerning how each algorithm performed under different network scenarios and conditions.

V.F. Challenges and Recommendations

Throughout our research, we documented any challenges or anomalies encountered. These not only served as lessons but also showed the way for our final conclusions. Drawing from our findings, we also formulated actionable recommendations for future research and practical applications to increase network security using DPI methodologies.

Complexity of Searching Algorithm:

Systems, like DPI, employ various searching algorithms, each with its own complexity, which is a critical factor. The system is often occupied due to continuous string matching, emphasizing the importance of an efficient algorithm to optimize running time.

Huge Number of Intruder Signatures:

As the landscape of cyber threats evolves, the diversity of attacks increases. Consequently, the DPI system needs to accommodate a growing number of intruder signature series, necessitating scalability to ensure effective defense.

Multiple Overlapping Signatures:

Intruder signatures are categorized into groups based on shared properties, such as protocol type. This specialization results in signature overlapping, requiring packet processing before the string matching step.

Location of Signatures:

In comparison to other systems, DPI conducts a thorough examination of the entire packet and pattern. This exhaustive approach ensures comprehensive scrutiny of data but can pose challenges in terms of efficiency.

Incorrect Alarms:

DPI, like IDS (Intrusion Detection Systems), generates numerous alarms, some of which may be false positives. The review process for these alarms becomes a daunting task, posing a significant challenge in maintaining accuracy.

Inspection Speed:

The current communication speed is at 10GbE/OC192, with an impending shift to 40GbE/OC768 [25]. This necessitates an increase in DPI speed to avoid becoming a potential bottleneck in network operations.

Data Encryption:

DPI faces limitations in handling encrypted data, necessitating the allocation of the inspection process either before or after the decryption stage.[18]

VI. Results and discussion

In conclusion, during the comprehensive comparison of these algorithms, it became evident that each of them exhibited its own unique set of strengths and

limitations. However, among this diverse array of algorithms, one particular standout emerged as the most efficient and resource-friendly solution – the p_3FSM algorithm.

Our research showcased the exceptional performance of the p3FSM algorithm. In terms of effectiveness and resource utilization, it surpassed all other contenders. Notably, p3FSM not only achieved superior results in terms of computational performance but also proved to be incredibly memory-efficient, particularly when executed on GPU hardware [16].

The primary challenge in locating the relevant section of the packet capture file stems from the variable lengths of Ethernet frames. This variability makes the prediction of packet offsets within the capture file difficult. Blitzdump addresses this challenge by navigating within the packet capture file, identifying individual packets, and determining the file offsets that mark the beginning and end of the user-specified time window. To achieve this, Blitzdump scans the target file, searching for data sequences resembling IPv4 Ethernet frames, and extracts their timestamps. By slicing the packet capture file based on these identified timestamps, Blitzdump generates a new valid packet capture stream that includes only the packets within the designated time window. The outcome is a filtered packet capture file that is well-suited for transfer to the analyst.

For any algorithm, the speed of reading data packets is important, in this case there is this method. After research studies we find out that we have an alternative method to read capture. Blitzdump enhances the speed of querying packet captures by selectively examining small slices of the capture file that define a specific time window. With Blitzdump, analysts can anticipate prompt results for their complete packet capture requests, measured in seconds rather than hours. The tool reads a designated capture file for each packet capture file stored in a specified directory (excluding subdirectories) using "indirectory" and locates packets within the configured time window, filtering and directing the results to an output file. Blitzdump achieves its rapid performance by employing a random access, binary-search approach to filter packets within a very limited time window within the capture file(s), as opposed to reading the entire capture file sequentially. Notably, on Linux, Blitzdump offers versatility by enabling the utilization of tcpdump for filtering and supporting the Berkeley Packet Filter format through the bpf option.[14]

This discovery marks a significant milestone in the field of algorithmic research and has the potential to revolutionize how various computing tasks are approached. The implications of these findings are far-reaching, opening up

exciting opportunities for improved computational efficiency and resource optimization across a wide range of applications.

VII. Conclusion and future works

The digital era has introduced a lot of network vulnerabilities, making the task of securing network infrastructures both imperative and challenging. This research underscored the pivotal role of packet inspection techniques, predominantly in intrusion detection systems (IDS). By providing an exhaustive overview of the three principal packet inspection methodologies—Shallow Packet Inspection (SPI), Medium Packet Inspection (MPI), and Deep Packet Inspection (DPI)—the study has illuminated their respective utility and importance.

While IDS and its components were elaborated in detail, the study particularly emphasized the significance of DPI. DPI, a cornerstone of network security, uses a variety of algorithms to strengthen network protection against harmful threats like viruses and malicious software. Our study concentrated on investigating the primary DPI algorithms and evaluating their efficacy in combating malware. The analysis and direct comparisons yielded valuable insights into the performance of each algorithm.

While the current research yielded substantial findings, the ever-evolving landscape of network security calls for more investigation. Looking ahead:
Deep Packet Analysis with Machine Learning: Machine learning offers capabilities in pattern recognition and predictive analysis. Incorporating machine learning into deep packet inspection can potentially enhance the accuracy and efficiency of threat detection. Future researchers should study integration of machine learning techniques with DPI.

Optimization Strategies: Algorithmic efficiency remains essential in ensuring real-time threat detection and mitigation. To further our commitment to improving network security, future efforts should focus on developing new strategies aimed at reducing the computational overhead of DPI algorithms. By doing so, we could enhance the responsiveness and agility of intrusion detection systems.

Expanding the Scope of DPI: As digital infrastructures expand and diversify, so do the potential threats. Future research initiatives should also consider expanding the scope of DPI to cater to evolving digital ecosystems, perhaps by encompassing IoT devices, edge computing platforms, and more.

To wrap up, the advancements achieved in this study have shed light on and enriched our knowledge of packet inspection methods. However, future

studies will have more breakthroughs, obstacles, and advancements to come. Our main goal remains the creation of a safer and stronger digital space for all. This survey serves as an initial step in acquainting readers with DPI systems and highlighting open research avenues in the field.

References

1. James P. Anderson Co., Computer security threat monitoring and surveillance, 1980
2. online: https://dataprot.net/statistics/hacking-statistics/
3. D.Ashok Kumar, Intrusion Detection Systems: a review, India, pp.1
4. Sun-young Im , Seung-Hun Shin , Ki Yeol Ryu , and Byeong-hee Roh, Performance Evaluation of Network Scanning Tools with Operation of Firewall, IEEE, 2016
5. Xinzhou He, Research on Computer Network Security Based on Firewall Technology, DOI:10.1088/1742-6596/1744/4/042037, Journal of Physics Conference Series, 2021
6. Robert Hamilton, Wayne Gray, Clifford Sibanda, Subbiah Kandasamy, Raimund Kirner, Athanasios Tsokanos, Deep Packet Inspection in Firewall Clusters, IEEE, 2020
7. Fatima Ezzahra Laghrissi, Samira Douzi, Khadija Douzi, Badr Hssina, IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism, DOI:10.1186/s40537-021-00544-5, Journal of Big Data, 2021
8. Eric Knapp, Joel Langill, Industrial Network Security, ISBN: 9780124201149, 2014
9. Argha Ghosh, Dr. A. Senthilrajan, Research on Packet Inspection Techniques, INTERNATIONAL JOURNAL OF SCIENTIFIC and TECHNOLOGY RESEARCH VOLUME 8, ISSUE 11, NOVEMBER 2019
10. Mohan V. Pawar, Anuradha J, Network Security and Types of Attacks in Network, International Conference on Intelligent Computing, Communication & Convergence(ICCC-2014)
11. Christopher Parsons, Deep Packet Inspection in Perspective: Tracing its lineage and surveillance potentials, Working Paper, January 10, 2008.
12. Jason Andress, The Basics of Information Security (Second Edition), ch10, 151-169, 2014
13. Thaksen J.Parvat and Pravin Chandra, , A Novel Approach to Deep Packet Inspection for Intrusion Detection, Elsevier B.V, 2015.

14. Smallwood, D., & Vance, A. (2011). Intrusion analysis with deep packet inspection: Increasing efficiency of packet based investigations. 2011 International Conference on Cloud and Service Computing. doi:10.1109/csc.2011.6138545
15. Chengcheng Xu, Shuhui Chen, Jinshu Su, Yiu and Lucas C. K. Hui, A survey on Regualr Expression Matching for Deep Packet Inspection:Applications, Algorithms, and Hardware Platforms, IEEE Communications surveys and Tutorials, Vol18, 2016.
16. A. V. Aho and M. J. Corasick, Efficient string matching: an aid to bibliographic search, Communications of the ACM, vol.18, issue.6, pp.333-340,1975
17. Cheng-Liang Hsieh, Lucas Vespa, Ning Weng, A high-throughput DPI engine on GPU via algorithm/implementation co-optimization, Elsevier Inc., 2015
18. Reham Taher El-Maghraby, Nada Mostafa Abd Elazim, Ayman M. Bahaa-Eldin, A survey on Deep Packet Inspection, 2017
19. Sun Wu, A fast algorithm for multi-pattern for multi-pattern searching, 1994
20. Vasudha Bhardwaj and Vikram Garg, "A Comparative Study Of Wu Manber String Algorithm and its Variations", International journalof Computer Applications, Vol 132, December 2015
21. Lucas Vespa, Mini Mathew and Ning Weng, P3FSM: Portable Predictive Pattern Matching Finite State Machine, 2009 20th IEEE
22. Allauzen, Gaspard-Monge, Factor oracle: a new structure for pattern matching, University of Marne-la-Valle
23. A.Khraisat, I.Gondal, P.Vamplew, J.Kamruzzaman, Survey of intrusion detection systems: techniques, datasets and challenges
24. Tamer AbuHmed, Abedelaziz Mohaisen, and DaeHun Nyang, A Survey on Deep Packet Inspection for Intrusion Detection Systems, Information Security Research Laboratory, Inha University, Incheon 402-751, Korea
25. Ajay chaudhary and Anjali Sardana, "Software Based Impplementation Methodologies for Deep Packet Inspection", International conference on Information Science and Applications (ICISA), IEEE, 2011

Фариха Наурызбаева[1], Асемай Иманкулова[1], Гауһар Сейіткалиева[1], Шахназар-Сұлтан Манбай[1]
[1]«SDU University», Қаскелең, Қазақстан

*e-mail: 221107045@stu.sdu.edu.kz, 221107057@stu.sdu.edu.kz, 221107021@stu.sdu.edu.kz, s.manbay@sdu.edu.kz

ШОЛУ: ИНТРУЗИЯНЫ АНЫҚТАУ ЖҮЙЕЛЕРІ ҮШІН ТЕРЕҢ ПАКЕТТІК ТАЛДАУДЫҢ ТИІМДІ АЛГОРИТМДЕРІ

Аңдатпа. Бұл зерттеу компьютерлік желілерді әртүрлі онлайн қауіптерден қорғауда терең пакеттік тексеру (DPI) әдістерінің рөліне ерекше назар аудара отырып, енуді анықтау жүйелері (IDS) және желілік қауіпсіздік саласын зерттейді. Компьютерлік желілерге қасақана кірулер немесе желілік шабуылдар қылмыстық астарлы болуы мүмкін. Шабуылдаушы жүйені басып алуды, құпия деректерді алуды, желі функцияларына кедергі келтіруді немесе тіпті веб-сайттар мен серверлерді жоюды қалауы мүмкін. Есепке сәйкес, АТ-инфрақұрылымына шабуылдардың өсіп келе жатқан жиілігі енуді анықтау қажеттілігін көрсетеді. Ол DPI алгоритмдерінің осалдықтарға қарсы тұру үшін қаншалықты жақсы жұмыс істейтіні туралы білімдер жиынтығындағы олқылықты көрсетеді. Талдау қазіргі уақытта желі қауіпсіздігінің қаншалықты нашар екенін көрсетеді, соның ішінде төлемдік бағдарлама шабуылдары, шағын ұйымдарға бағытталған кибершабуылдар және айтарлықтай қаржылық шығындарға әкелетін деректердің бұзылуы туралы таңқаларлық деректер. Есепке сәйкес, адам қателігі әлі де желі қауіпсіздігінің негізгі әлсіздігі болып табылады. Қауіпсіздікті анықтау жүйелері (IDS) қажетсіз кіруді болдырмау және жүйелерді қорғау үшін жиі пайдаланылады. IDS ықтимал енулерді анықтау және зиянды әрекетті тоқтату үшін жүйе немесе желі оқиғаларын анықтауды және бағалауды талап етеді. Желі қауіпсіздігі басты алаңдаушылық ретінде атап өтіледі, өйткені зиянды актерлар желілерге шабуыл жасаудың жаңа жолдарын үнемі ойлап отырады.

Бұл зерттеудің негізгі мақсаты әртүрлі DPI енуді анықтау әдістеріне қатысты білімдер жиынтығын зерттеу және құрастыру болып табылады. Бұл жүйелерді тереңірек түсінуге көмектесу үшін IDS үшін морфологиялық негіз құруды ұсынады. Зерттеу әртүрлі DPI түрлерін және олармен бірге жүретін алгоритмдерді мұқият түсіндіреді.

Түйін сөздер: Пакеттерді терең тексеру, орташа пакеттерді тексеру, таяз пакеттерді тексеру, сәйкес алгоритмдер, шабуылды анықтау жүйесі, киберқауіпсіздік, зиянды бағдарламалар.

Фариха Наурызбаева[1], Асемай Иманкулова[1], Гауһар Сейиткалиева[1], Шахназар-Сұлтан Манбай[1]
[1]«SDU University», Каскелен, Казахстан

ОБЗОР: ЭФФЕКТИВНЫЕ АЛГОРИТМЫ ГЛУБОКОГО АНАЛИЗА ПАКЕТОВ ДЛЯ СИСТЕМ ОБНАРУЖЕНИЯ ВТОРЖЕНИЙ

Аннотация. В этом исследовании рассматривается область систем обнаружения вторжений (IDS) и сетевой безопасности, с особым упором на роль, которую методы глубокой проверки пакетов (DPI) играют в защите компьютерных сетей от различных онлайн-угроз. Преднамеренные вторжения в компьютерные сети или сетевые атаки могут иметь криминальный подтекст. Злоумышленник может захотеть захватить контроль над системой, получить конфиденциальные данные, вмешаться в сетевые функции или даже отключить веб-сайты и серверы. Согласно отчету, растущая частота атак на ИТ-инфраструктуру подчеркивает необходимость обнаружения вторжений. Это указывает на пробел в знаниях о том, насколько хорошо алгоритмы DPI противодействуют уязвимостям. Анализ показывает, насколько плоха сетевая безопасность в настоящее время, включая поразительные данные об атаках программ-вымогателей, кибератаках, нацеленных на небольшие организации, и утечках данных, которые приводят к значительным финансовым потерям. Согласно отчету, человеческая ошибка по-прежнему остается основным недостатком сетевой безопасности. Системы обнаружения вторжений (IDS) часто используются для предотвращения нежелательного доступа и защиты систем. IDS предполагает идентификацию и оценку системных или сетевых событий с целью обнаружения возможных вторжений и прекращения вредоносной деятельности. Сетевая безопасность рассматривается как серьезная проблема, поскольку злоумышленники постоянно придумывают новые способы атак на сети.

Основная цель этого исследования — изучить и собрать совокупность знаний о различных методах обнаружения вторжений DPI. Он предлагает создать морфологическую основу для IDS, чтобы помочь нам понять эти системы на более глубоком уровне. В исследовании дается подробное объяснение различных видов DPI и алгоритмов, которые с ними связаны.

Ключевые слова: Глубокая проверка пакетов, средняя проверка пакетов, неглубокая проверка пакетов, алгоритмы сопоставления, система обнаружения вторжений, кибербезопасность, вредоносное ПО.