A Survey on Deep Packet Inspection

Reham Taher El-Maghraby
Computer Engineering Department.
Arab Academy for Science and Technology, College of Engineering
Cairo,Egypt
rehamelmaghraby@aast.edu

Nada Mostafa Abd Elazim
Computer Engineering Department.
Arab Academy for Science and Technology, College of Engineering
Cairo,Egypt
nadamostafa@aast.edu

Ayman M. Bahaa-Eldin
Misr International University,
On leave from Ain Shams University,
Cairo,Egypt
ayman.bahaa@eng.asu.edu.eg

***Abstract*— Deep Packet Inspection (DPI) considered as one of the most important parts in content-aware network applications such as copyright enforcement, Intrusion detection system (IDS) and other applications will be discussed later. DPI rely on comparing to parts payload and signature (IP header). IT compares them with known signatures to decide if the packet is harmful (similar to any of attacks database signatures) and delete it or pass it through the network flow.**

it deals with the content below the 4th layer of the IP packet that includes source and destination ports, source and destination addresses and type of protocol. It classifies type of the application depending on its port number .

For signature comparison, many algorithms are applied such as regular expressions (most popular) and others discussed later. Nowadays many applications rely on DPI for inspecting packets in network stream.

This survey gives a brief idea about challenges in DPI and some of the design objectives. Then explaining in short words different matching algorithms with their limitations. At the end, some of the most popular techniques using DPI.

Keywords—Deep Packet Inspection; Intrusion Dtection System; Matching Algorithms; DPI Challenges

I. INTRODUCTION

Deep packet inspection (DPI) is an enhanced way of network filtering, applied on the application layer of OSI architecture. DPI, detects data part (packet content) and signature part (packet ID). Devices of DPI deals with streaming packets in real time. It used to detect viruses and harmful packets by looking for protocol non-compliance, intrusions, or described principles to choose the packets to pass or be neglected or if there another path to be taken (another destination).

DPI makes network filtering by examining the signature of the payload packet either by string matching algorithms like Wu-Manber, Aho-corasick, and SBOM, or by regular expression matching algorithms which is used in NIDS of Snort [1] and Bro [2] and L7-filter in Linux[3].

The DPI advantages summarizes in finding, detecting, categorizing, blocking and redirecting packets that can't detected by normal packet filtering ways. Also, supports communication service providers to assigns different accessible resources to streamline traffic flow. In Addition, used to control transferring data to avoid network abusing in order to improve network performance for other users [ ].

On the other hand, DPI had some disadvantages first it can produce new weaknesses point, can also decrease computer performance by increasing processor load. In Addition, DPI increases the complexity of the firewall and other security software.

There are two ways to collect packets to DPI, port mirroring and optical splitter. Port mirroring is a mechanism that inspects each layer of OSI layers then sent the aimed information to network manager. A mirror of all layers is formed then filtering data in the application layer. In optical splitter, packets information is collected and sent to network manager that used to filter data. Algorithms and ways of filtering data will be discussed in the following sections.

The remaining of the paper is orderly as follows. Section II discusses the implementation of deep packet inspection. Section III illustrates some challenges on DPI. Section IV illustrates the Design issues and objectives of using DPI. Section V presents some algorithms used to filtering the payload packets. Section VI gives a description of some DPI techniques. Section VII gives a description of some DPI Application. Finally, we conclude in Section VIII.

II. DPI IMPLEMENTATION

DPI is approximately categorized to three different implementations are signature-based identification, application layer based identification and behavior based identification [29].

A. Signature-based identification

Different applications are used by different protocols, and each protocol labeled with its signature. Each signature may contain certain port or an order of string or bit sequence. Detecting is done by matching the signature of the packets depending on this matching DPI can recognize this data flow goes to any of the applications.

B. Application layer based identification

In some applications, control and services are different and some features may lack later. In these cases, identification of

the application gateway is the only way by identifying control flow and relative service flow at last define particular protocol. Applications such as SIP and H323 protocols identified by application gateway. It works by negotiating data channel by swapping specific signals.

C. Behavior-based identification

Behavior identification used in the case of data flow can't be recognized by any protocol. Although it judges depending on user behavior or specific terminal such as SPAM mails (SPAM mail identified by specific behavior). Flow then relative service flow at last define particular protocol. Applications such as H323 and SIP protocols identified by application gateway. It works by negotiating data channel by swapping specific signals.

III. CHALLENGES

DPI faced some challenges can be classified into two groups. First challenges in challenges faced while developing DPI systems (1). Second DPI challenges to build DPI.

A. Challenges of Developing DPI Systems

In DPI systems, to reach high performance while dealing with high speed links considered as big challenge depending on some issues [28]:

- Large number of signatures in networked application
- What is the difficulty of signature patterns
- Signature location randomness in the stream of the network, as well as, in the packet data
- How DPI implementing in operating systems and hardware levels

B. DPI signature series challenges

In the last years, there is large increasing happened in the application signatures, reasons due to development of systems , malicious applications and facilities. DPI used to attempt to check the packet signature of the total millions of applications signatures or rules. On the other hand, increasing the series of signatures, leading to increase the extensiveness of the identified traffic and decrease the DPI systems performance. DPI had the most complicated rules compared to other systems such as SNORT , L7 filter and XML filter. The requirement to develop the rules of DPI system is always required.

C. OS and hardware bottlenecks challenges:

In the operating system there what known with bottleneck. The simplest example of bottleneck is when the process waits a wanted resource to be available and kernel designing make bottleneck happened. Interrupt latency is what to be worried about (it's the time between the interrupt request and the start of serving. In Addition, exchanging data between threads or between threads and interrupt contexts is another fear. Bottleneck happens when more than one thread need the same resource and its locked to other thread. Also, there is what known by hardware bottleneck. Summarized in the processor ,memory and hard disk and the connection between them that is represented as a bus interface. Large memory and high-speed processor don't assure high performance. The devices are not connected in a physical way, the main memory act as a shared interface offering data exchange abilities. In the case , data need to pass through main memory it may causes bottleneck to occur.

D. DPI challenges in intrusion detection systems:

Using DPI in intrusion detection systems (IDS), group of challenges must be considered. These challenges are briefed as follows [29][27].

- Complexity of searching algorithm

  Signature based systems such as DPI use different searching algorithms with different complexity that considered as an important factor. Based on that the system is busy most of the time because of string matching. Running time is important as same to string matching so effective algorithm is necessary.

- Huge number of intruder signatures

  Now a day, more types of attacks has been established. For this reason, number of intruder signature series must increase that affect the DPI system. The system must be scalable.

- Multiple overlapping signatures

  Intruder signature divided to groups depending on sharing some of the same properties such as protocol type. That is due to that these signatures are not general that leads to signature overlapping. Packet process is requirement before string matching step.

- Location of signature

  Comparing DPI with other systems, DPI used to check all the whole packet and the pattern.

- Incorrect Alarms

  IDS used to generate thousands of alarms some of these alarms is a false alarm. All the alarms must be revised can accelerate rapidly causing task to be very difficult to achieve.

- Inspection speed

  Communication current speed is 10GbE/OC192 and impending to 40GbE/OC768 [29]. Therefore, DPI speed must be increased, leads to potential bottleneck.

- Data Encryption

  DPI can't deal with encrypted data so, inspection process must be allocated before or after decryption process.

IV. DESIGN ISSUES AND OBJECTIVES

In designing DPI, group of topics must be satisfied in order to keep in pace with attack growth and increasing channel speed communication. These objectives are[29][27]:

- System Scalability:

  Scalability of the system not affect software system but need fast processor. On the other hand, considered as a big issue in hardware systems not because of speed but the main problem is they aren't rrestructuring.

- Availability of signatures

DPI can detect different signatures of intruders and must be updated must be always updated with the new intruders' signatures in order to be detected and secure the connections.

- Quality of Service (QoS)

  ISP uses DPI to provide privilege services to their customer. In order to fix this DPI must address some of QoS problems such as bandwidth management.

- Memory efficiency

  In software implementation, memory access time is one of the main bottlenecks in DPI system. On other hand, it is critical in hardware design and memory scarcity.

V. ALGORITHMS

A. Regular Expression Matching Algorithm

They are used in NIDS of Snort and Bro and L7-filter in Linux.

Finite state machine (FSM) [4] is a machine that has same language expressed by regular expression. It consists of some states and directed edges, there is node called "initial" state and some "accepting" states. It works as follows: first, it starts from the initial state and read the first byte of the payload. Second, it transfers to the next state according to the labels on the edges. Finally, if it reaches any accepting states so this payload and corresponding signatures are matched. Most of applications on DPI uses automaton based for signature matching.

There are two types of FSM, deterministic and non deterministic finite automata (DFA, NFA), they are the same in expression power but processes in a different way, the DFA has only one transition and one active state , but NFA has many transitions and many active states. Also, the processing cost per step of DFA has order O(1) but NFA has order O(n^2m).

B. String Matching Algorithms

Aho-corasick algorithm:

This paper [5] presents a simple and efficient string matching algorithm to set all the appearances of a number of finite set of keywords in a string of text. This algorithm consists of building a finite state pattern matching machine from the keywords and then using it to process the text string in a single path. The time taken to build this machine is proportional to the sum of the lengths of the keywords. This paper describes a pattern matching machine that locates keywords in a text string. Let k be keywords and x be a text string, this matching machine locates and identifies all substrings of x which are keywords in k, it is a program that takes input x and produces the locations of keywords in x, also it consists of a number of states, and each one is represented by a number. The pattern matching machine consists of three functions: a goto function that makes mapping of a state and an input symbol to a state or message failure, a failure function that makes mapping of a state to a state, and the output function associates a set of keywords with every state.

The limitation of this algorithm, it provides low performance and more overhead due to code complexity.

It is concluded that the algorithm improving the speed of a library bibliographic search program by a factor of 5 to 10. Also the pattern matching machine is being convenient for applications when we are searching for occurrences of huge numbers of keywords in text strings, searching can be made over arbitrary files.

Another paper [6] taking about parallel Aho-corasick in deep packet inspection based on graphic processing unit. Parallel Aho-corasick string matching approach (PAC-K). This algorithm works as follows; an input string is divided into number of chunks, each one is addressed to its own thread. To solve the boundary detection problem [7], scan each thread across the boundary. In addition to that, the input string is splitted into number of sub-chunks with k characters. And each sub-chunk allocated to a thread to detect the patterns that start at any position in the sub-chunk. When there is a failure, it adapts a new starting position in a sub-chunk, and the required number of threads is decreased by factor k. The construction of the PAC-k DFA is described as follows; a string matching is performed by the thread starting from the first character in a chunk. If a goto function occurs, the next state is determined using the PAC-k DFA for the target patterns, if not check the validity of the failure transition. If the current and next states in the same level after the failure transition, a new starting location is acceptable, then the thread performs the string matching from the new position. If this new starting position is not in range, it results overlapping between two matching strings of different threads. To control this overlapping, checking procedure must be added as described in the paper.

This paper concluded that the PAC-k approach based on GPU can enhance the performance, and making it more flexible and updatable.

Wu-Manber Algorithm:

This paper [8] presents an algorithm used to search for multiple patterns at the same time; it is faster than any other algorithms and supports huge number of patterns.

This algorithm used in many applications such as data filtering to searching for selected patterns, security applications to detect certain suspicious keywords, searching for patterns such as dates, glimpse to search for all items at the same time by using Boolean queries and DNA searching.

The algorithm consists of two phases, preprocessing and scanning phases. The first part in the preprocessing phase is to calculate the minimum pattern length, and consider in each pattern only the first m characters. The preprocessing phase contains three tables called: SHIFT table, HASH table and PREFIX table. The values inside the SHIFT table determine how many characters can be shifted when the text is scanned. When the shift value is equal to zero, determine if there is a match in HASH table and check the values in the PREFIX table. The second phase is the scanning phase, it is described as follows, calculate a hash value h based on the current B character from the text, check the value of SHIFT[h]: if it is greater than zero so shift the text and go to the previous step, if not so calculate the hash value of the text prefix, finally, check for each pointer p, if PREFIX[p] equal to hash value of the text prefix, check the actual pattern against the text directly.

It is concluded that the algorithm is fast, and very simple as it is improved the running time when the number of patterns more than 8000 patterns, the time used for the preprocessing

stage for the program is very fast up to 1000 patterns taking time of 0.1 second, it becomes smaller for more patterns.

The limitation of this algorithm [9], the length of the patterns must be the same, it is less efficient, and can't handle more than few hundred patterns.

SBOM Algorithm:

In this paper [10], it introduces the algorithm that constructs the factor oracle and discusses the on-line construction algorithm of the automaton oracle, that illustrate a method of constructing the automaton by reading the input one by one and from left to right. Also it illustrates a new matching algorithm called Backward Oracle Matching (BOM).

C. *Finite State Machine*

One of the most implements designed for hardware application, include two important types are Non-Deterministic Finite State Machine (NFA) and Deterministic Finite State Machine (DFA).

NFA is directed graph which consists of states represented by nodes and directed edges. NFA had one or more final state with only one initial state. NFA is important in parallel processing in DPI based system, so it can input character in many branches of NFA and may output many acceptance states for single input.

DFA is used in wide range to execute regular expression in linear time. It consists of a transition function and a finite set of inputs symbols and states. to move from one state to another. Predictable Bandwidth is important property of DFA that helps researches to reach optimal solutions.

Hybrid FA is modified DFA, in order to achieve different requirement of memory compared to that of NFA, and typical case of the requirement of memory bandwidth unlike that of DFA. They are bright to accomplish worst case memory bandwidth linear to the number of regular expression containing counting constrictions and mark start representation regardless of number of states in automata.

The algorithms discussed in this paper are very efficient, simpler to implement, and require small memory.

VI. DPI TECHNIQUES

A. *DPI using Bloom Filter*

Bloom filter is used in filtering applications and used to recognize malicious action such as viruses in high speed networks.

Bloom Filter in paper [11] is constructed using array that holds i-locations consists of one bit and uses k hash functions, initially, all the locations in the array have value zero. If we have n elements that belong to a set of signatures, we apply k hash functions to an element. The value of each hash function is used as an index to the m-bit array. All the k- locations are set to one. These steps must repeat for all the signatures [12]. If any location in the array is zero, so the element is considered as unregistered signature. If all the locations in the array are set to one, so the element is considered as member of the signature, as shown in figure 1.

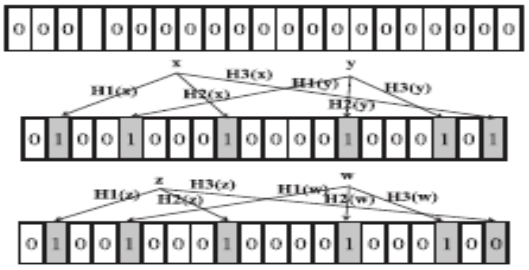

Fig. 1. Bloom Filter

This paper [13] presents a parallel bloom filter in Deep packet inspection based on hardware-based technique. A Bloom Filter (BF) is a structuring of data that stores signatures and computes multiple hash functions on each member of the signatures. The storage space of BF doesn't depend on the string length. BF doesn't include false negatives but includes false positives.

Parallel bloom filter is stored a set of signatures grouped by length. It is consists of number of bloom filters with different signatures length. Bloom filter engine reads a data stream arriving with rate one byte per clock cycle. Using this technique enhances the throughput.

One of the properties of BF that it is impossible to delete an element stored in the filter. To solve this problem, Counting Bloom filter is used. It adds and deletes an element by increments or decrements the counters corresponding to the hash values. When the value of the counter is changed to one, so the bit is added in the bit vector. When the value of the counter is changed to zero, so the bit is deleted from the bit vector.

In this paper [14] it introduces a fingerprint counting Bloom filter (FP-CBF). It is used to improve the performance of CBF, and reduce the false positive rate.

FP-CBF uses an array of m locations each location has two fields, a counter with c bits and a fingerprint with f bits. The number of bits in FP-CBF is (c+f)*m bits. It is needed k+1 hash functions to insert/delete and query for elements. The first k hash functions are mapped an element to one of the m locations. The last hash function $h_{fp}(x)$ maps one element to one of the 2^f possible values of the fingerprint field.

FP-CBF consists of three operations: insert, delete and query operations. When inserting element x in FP-CBF, the value of the counters are incremented by one, and the fingerprints are changed by making xor between the stored value and $h_{fp}(x)$. When deleting element x from FP-CBF, the value of the counters are decremented by one, and the fingerprints are changed by making xor between the stored value and $h_{fp}(x)$, to delete the stored fingerprint , the element was inserted as $h_{fp}(x)$ xor $h_{fp}(x)=0$. When querying for an element x, the hash functions stored in the array are read and checked as follows: the element isn't found in the FP-CBF and the query is failed, either if any of the counters is zero, or if the counters have a value of one or check if the fingerprint field is different from $h_{fp}(x)$. The query is successful, if both of the previous two checks are passed.

It is concluded that the FP-CBF is so simple, improve the performance of CBF and reduce the probability of false positive rate.

In this paper [15] it proposes new bloom filter architecture based on pipelining technique. The pipelining bloom filter is consists of two stages of hash functions, the first one to calculate the hash values and the other one to calculate the hash values for the first stage that there is a matching between the input and the signature as shown in figure 2.

Pipelining bloom filter has same number of hash functions as regular bloom filters, it takes the advantages of the network traffic that is free from virus, the advantages of using this technique that if the first stage gets mismatch output so there is no need to enter the second stage in order to decide whether it is a member of the signature set. It is concluded that pipelining can decrease the consumption of power up to 90%.

Suggested future work is to perform experimental evaluation of these pipelined Bloom filter architectures

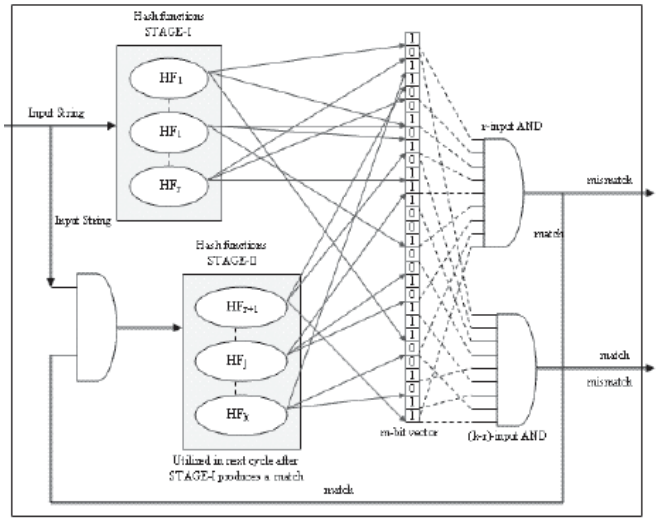

Fig. 2. Pipelining Bloom Filter

The proposed system for Deep packet inspection uses QF to monitor the traffic of the network and to check the incoming stream of network data, if these data are a string of interest. If it doesn't match then the data can be cancelled so a false negative is impossible, but if it matches then it can be a possible string with a very high probability. So the proposed system is able to process a greater database with suitable resources, and supports fast changes in the database. The proposed system consists of four stages: quotient filter, input manager, match manager, and output manager. The quotient filter makes query that check if the signature is matched or not. Input manager has the inputs that enter to the quotient filter the inputs are arranged in buffer as FIFO. Match manager makes match notifications that get from quotient filter. Output manager sends the packet to its original path. The system sets to zeros the quotient filter, and then starts the program of the quotient filter that depends on set of signatures, specifies the window size that the system works on it that has different size, checks the content of the window and QF for each incoming packet. If the QF responds that match exists then process the suitable action, if not match so shift the window to check the next data in the packet till the end as shown in figure 3.

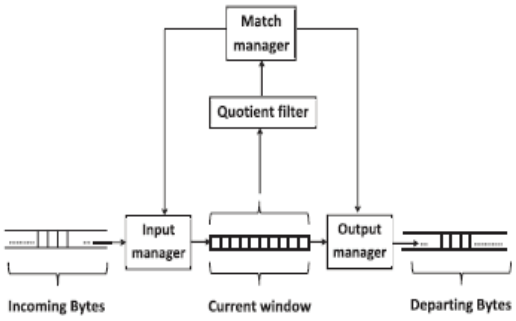

Fig. 3. Proposed DPI Algorithm in quotient filter

B. *DPI using Quotient Filter*

this paper [11] illustrates a brief comparison between the quotient filter (QF) and bloom filter (BF) and which one is better and introduces a proposed algorithm for DPI using quotient filter.

QF is more accurate than BF as it has low false positive rate. It has high throughput and make query faster than BF, finally it is able to update the signatures dynamically as it supports insertion and deletion of any element.

QF works as it stores the n-bits fingerprints of elements, each fingerprint divides into two parts one is the quotient and the other is the remainder. The quotient part represents an index to a bucket that stores a remainder. A collision may be occurred in the programming phase of QF when q (quotient) of two different fingerprints are the same, to solve this problem; linear probing strategy is used to store the remainder part in an ascending order. Each slot in the QF consists of three additional bits: is_occupied, is_continuation and is_shifted. If a new fingerprint has to be inserted, a quotient part used as an index to the bucket and set the value of is_occupied bit by one, if there is any collision occurred the remainders put in an ascending order and the other two bits (is continuation and is shifted)have value one.

Suggested future work to apply high throughput and online packet inspection in high bandwidth networks, this algorithm can be implemented on parallel architectures such as FPGAs and GPUs.

C. *DPI using Cuckoo Filter*

In this paper [16], it presents a new technique for Deep packet inspection by using cuckoo filter (CF). CF is a matching check tool in DPI to overcome any weakness in the other matching check tool BF and QF.

CF enable insert and delete any element in O(1) time. It has hash table to save fingerprints for each element. Each element has two hash functions h1(x) and h2(x) which saved in an array of buckets. First, to insert a new element in the hash table first examine if this space is free or not, if it is free insert it but if not make relocation. Second to search for an element x, the algorithm should compute its fingerprint and two buckets, then the two buckets are examined if either one of them contains the required fingerprint, in this case the false negatives can't be occur. Third, to delete an element from CF, examine both of the two buckets if either one of them contains the required fingerprint so eliminate one copy of the required fingerprint. Finally, mapping all the signatures into CF. After mapping, the algorithm will start by examining all the packet's

bytes by sending query to CF looking for any match, if there is a match the algorithm stops searching and waiting for the next packet, if there is no match, it forwarding the packet to its original destination as shown in figure 4.

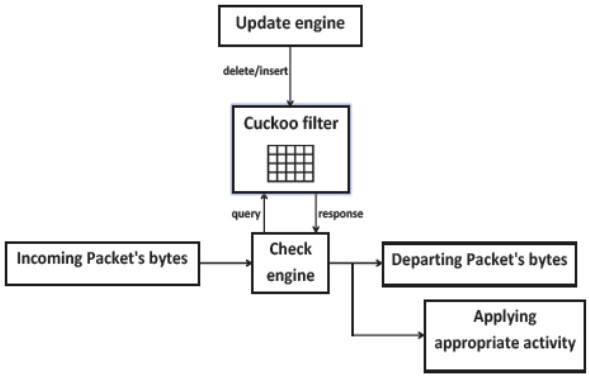

Fig. 4. Architecture of proposed DPI Approach in cuckoo filter

This paper concluded that CF consume less memory, greater insertion throughput, greater lookup performance, supporting delete operation and less false positive rate than BF and QF.

Suggested future work to apply better performance, this algorithm can be implemented on parallel architectures such as FPGAs and GPUs.

D. Deep packet inspection based on many-core platform:

Due to the growing of the computer networks technology both network bandwidth and traffic increase gradually. This paper [17] presents deep packet inspection based on many-core platform TILE_Gx36 that results both high performance and low power consumption compared with X86 platform.

DPI system based on TILE_Gx36 is divided into three modules: package receiving module, packet processing module and packet forwarding module. Receipt and Dispatch packets through mPIPE that getting message from the port of Ethernet then dispatch it to packet processing module. Packet processing module consists of two models: pipeline model and worker model.

The definitions of Pipeline model: each main central point is responsible for one stage of the packets processing. Subsequently distributing the whole packets that related to the same flow of main central point, accordingly the performance effect positively on the worker model.

Suricita, is the software architecture that packet processing based on, it means that packet received from mPIPE is processed and forwarded to forwarding module through PCIe. At Final, it sends the occurred results to the application layer ,upper layer, after it had processed. In this type of module, packets used to be sent to mPIPE by Ethernet port. After that, mPIPE sends these packets to the packet processing module , processing thread, according to the link layer protocol second layer protocol had decoded by the thread. Finally, third and fourth layers deals with the information of the packet.

By making function and performance tests it is concluded that the system has accomplished the functions of deep packet inspection, if the number of cores increase the power consumption increase, and if issued rules or not when the number of core increase the throughput increase, if the number of cores continue to increase, the curve will be declined.

Suggested future work is to optimize the algorithm to improve the performance.

E. Deep packet inspection based on TCAM:

Paper [18] display a high-speed deep packet inspection algorithm using TCAM((content of Triangular addressable memory) in order to secure the network versus malicious hitting. TCAM is a kind of memory used to be a high speed rate to looking for multiple elements simultaneously. Each cell has one of the three states (0, 1, or don't care). If the result of the search matches multiple, TCAM will snap back the location of the first attack or the locations of the multiple attacks.

In this algorithm, m-byte jumping window used instead of sliding window to gain parallelism for payload inspection, scanning time in jumping window has been improved. The algorithm is consisting of two phases: a preprocessing stage and a scanning phase. TCAM had been created by the preprocessing stage entries from the set of style. By scanning the payloads packets, the scanning phase used expose malicious attacks.

Concluding that the throughput of the m-byte jumping window is the sliding window, the throughput of m-byte jumping window is 0.81 million packets per second, and the throughput of sliding window is 0.17 million packets per second. So the throughput will have increased if m is large because the scanning time can be decreased. Accessing the TCAM in m-byte jumping widow increases slowly, but sliding window increases rapidly.

F. Deep packet inspection based on FPGA:

This paper [19] presents a deep packet inspection approach for network intrusion detection system based on FPGA. Firstly, describes static and dynamic pattern matching, and then discusses the architecture of DPI engine to handle multi-pattern signatures that support both static and dynamic patterns.

Static pattern matching uses Cuckoo-hashing method [20] that consists of 16 modules and priority circuit as shown in figure 5. Each module detects patterns with specific length relative to their location. The priority circuit selects the longest pattern from the 16 modules. Dynamic pattern matching uses regular expression patterns.

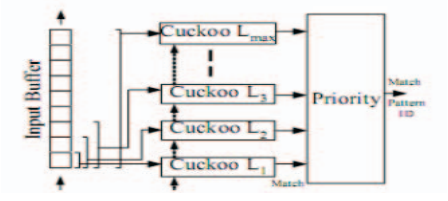

Fig. 5. Cuckoo based pattern matching engine

The proposed architecture for the deep packet inspection uses multi-pattern signature, the static and dynamic patterns, and consists of three groups of Snort rules, the first group has rules with a single pattern, and the second group has rules with multiple patterns with dependent relationship, and the third group has rules with multiple patterns that have no relationship between each others. The architecture consists of

cuckoo-based multi-pattern matching (CMPM) that include the core cuckoo-based pattern matching module (CPM), a mapping module, and three separated circuits to handle different groups of rules. Firstly, the input character enters the CPM engine to examine the presence of predefined patterns, and then the mapping module processes the indexes of these patterns to find which group forwards the matching indexes. Secondly, the converter takes the input character before transfer it to the CPM module. Finally, the input character enters the dynamic string group that will function like the third group rule discussed before. After that all the outputs result before enter the combined module as shown in figure 6.

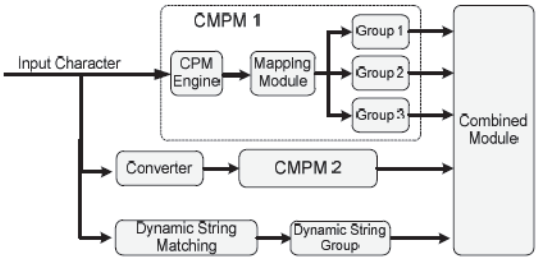

Fig. 6. Structure of Multi-pattern signature matching system

It is concluded that the throughput of the proposed architecture is 1.1 Gbps and it works on real environment.

G. Deep packet inspection in Software defined network (SDN):

SDN architecture as interviewed in [21] constructed using parallel DPI module. This paper had been divided into three main topics DPI software architecture using SDN, parallel DPI in SDN and at last adapted highest random weight (AHRW) hash scheduling algorithm with additional feedback.

The first part, used to exhibits the architecture of DPI in SDN controller. SDN controller governs of three layers are Application Programming Interface (API), abstraction and Openflow layers. API layer explains the DPI module in abstraction layer service as interface to application layer. Although the abstraction layer calls the Openflow protocol service by the manage module. At the last, Openflow layer achieves the Openflow protocol specification used to communicate with data layer. DPI module communicates with policy management module that provides the flow policy of DPI module and forwarding module, after that provides an interface to send flow table and management module. At the end, it had been concluded that the role of SDN controller is to manage the DPI rule table and flow table then sends the flow table with the flag and switch processes packet to the output port as shown if figure 7.

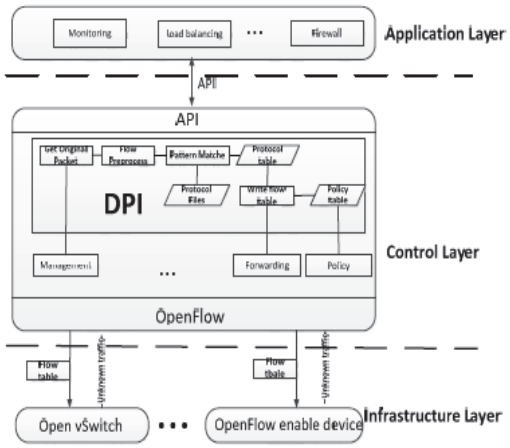

Fig. 7. Architecture of DPI in SDN

In the second part, the paper explains parallel DPI in SDN architecture as shown in figure 8 and figure 9. The reason of that DPI speed becomes the system's bottle neck is that SDN controller handle all other network traffic. The architecture above divided to four stages are get original packet, flow preprocessing, pattern matching and finally writing flow table. First, open flow switch directs the packets directly towards controller. After that, Open Flow packet had distributed by DPI module to perform the original packet. The step after is applying pattern matching technique by applying regular expression. At last, pattern matching result are concatenated to the policy and resend it to the switch. In this paper, Parallel DPI architecture used to enhance the DPI module speed. It contains three threads, preprocessing spreads, group of matching threads and write flow table thread as shown in figure 9. First preprocessing threads, receives the open flow packet from switch and divide it to obtain the original packet. The matching thread consists of a queue to determine different types of flow within it. The third type, flow table thread, combines all the processes obtained by the flows that have produced by MT's, then dispatch the flow table towards the switch.

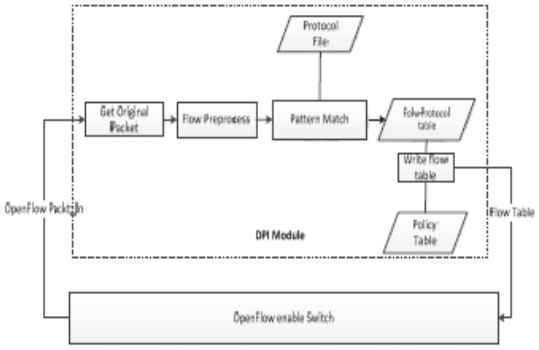

Fig. 8. Parallel DPI Algorithm in SDN Architecture

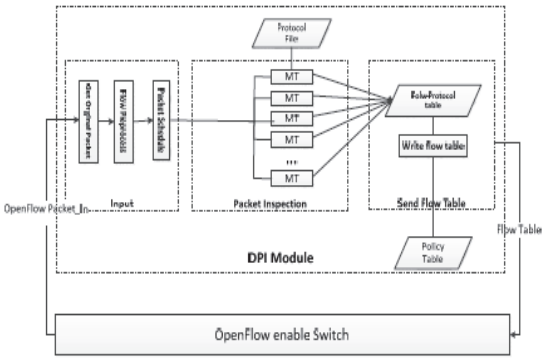

Fig. 9. Parallel DPI Algorithm in SDN Architecture

Finally, it illustrates the adaptive highest random weight hash function based on queue length, in a real time system hash mapping is weak, and the system has traffic imbalance problem. The system uses feedback circuit to adjust the scheduler and make benefit from randomness of the highest random weight hash. Another paper [22] adapts the weight on processing unit then plans the groups of pictures (GOP) according to their higher possibility. This paper uses multicore architecture to schedule packet on SDN controller. It applies pattern matching based on regular expressions, system is developed by obtaining feedback vector depend on real time queue length and match string length.

It is concluded that the parallel throughput accelerates linearly, after testing the algorithm of SDN controller based on multicore, the core utilization is decreased by 18% and throughput increased by 21% compared to highest random weight (HRW).

Suggested future work is to do traffic analyze using SDN and DPI.

Another paper [23] introduces method to detect DoS flooding attack on global environment for network innovations (GENI) using SDN.

The system architecture used to detect and mitigate the DoS flooding attack is described as follows, it contains several components and algorithms, it contains a client (node) that act as malicious user or regular user uses a network, and a server that runs web server, the system consists of several client nodes and only one server under attack. The system uses two algorithms, monitor and correlator. The monitor has two functions, first to operate as a sensing node using IDS, second to send alert messages containing some information about the attack to the correlator when it detects any malicious action. The correlator found in SDN controller, it receives alerts from the monitor and attach it with information processed by open virtual switch (OVS) switch for DPI. If it detects any attacks in a network, it knows the port and node from which the attack traffic come from and drops the malicious traffic by using OpenFlow API. The communication protocol between the client and server is discussed as follows, initially communicates the controller with the OVS to permit the modification of network flow. When the user sends request to the server, it is reflected to the monitor for inspection. The controller spreads a flow rule to the OVS. After request sends to the server, the server responds the content to the client through the OVS. If there is no alerts send to the correlator so the traffic flows normally. But if there is alert send to the correlator, the correlator sends query to the OVS to collect some sensitive information like port number, mac addresses and ip addresses related to the topology. The OVS will response with the flow tables needed to approve attack happened. The controller finally sends a command to the OVS to takes action to solve attack traffic if found.

When testing this approach on GENI, it is concluded that the solution of this approach is able to improve for processing high volume of traffic and large scale attacks.

Suggested future work is to improve a systematic methodology for this approach.

Another paper [24] introduces traffic scheduling for deep packet inspection in SDN using 2-phase algorithm to find the paths for incoming flows and solve the IPAD problem by minimizing the total delay of flows. The first phase of the algorithm uses K-Median (KM) approach can support entry switches to proxy nodes such that it minimizes the total distance between all switches to their proxies, or K-Center (KC) approach is to minimize the maximum distance between the switches and their proxies. The second phase finds the shortest path using modified dijkstra's algorithm. Modified dijkstra's algorithm states that arranging all switch proxy sessions in an ascending or descending order depend on their traffic rate, then labelling each link with initial value of available bandwidth and if there is a shortest path for switch-proxy session it updates the value. The algorithm continues iterations until all paths are established.

It is concluded that this paper wants to make balanced penalty for all incoming switches in DPI, also it studies an IPAD problem, and it focuses on minimizing the delay in overall transmission of all switch-proxy sessions by using 2-phase algorithms that select proxy and find the shortest paths for its session.

Another paper [25] presents the creation of deep packet inspection in SDN that solve many problems in packet traffic. This application used by the operator to detect the type of applications used by the users in the network and analyzing the real-time data. Also, it uses analytical capabilities to group its trends and displays understandable format for the users.

Suggested future work is related to integrate Apache Spark for data analysis, add machine learning algorithms for data analytics and make the north-bound application flow compatible to helium version of ODL that is one of the problems in this application.

H. Deep Packet Inspection in Encrypted Traffic

Middle boxes of the network achieve DPI to satisfy group of tasks such as intrusion detection (IDS) ((reveal if the attacks included in packets from cooperated sender), exfiltration detection (penetrating for watermarked and confidential data transmitted out scheme network) and parental filtering (prohibit under age from gaining access to adult materials). Packets flow through the network could be encrypted so HTTPS and middleboxes cannot deal with. For this reason, Blind Box is proposed [30].

Blind Box enables two computation classes of DPI each of them characterized by its secrecy agreements

1) exact match privacy

2) probable cause privacy.

First class, rely on DPI applications exact string matching such as watermarking, parental filtering and IDS. Middle Box acquires at which position in a flow attack keywords occur, for substring of the flow that do not match any of the key words attack, middlebox learns virtually nothing. For achieving this DIPEnc [30] and blind box [30] detect protocol used. Also, Obfuscated Rule Encryption technique [30] used to allow middle box in obtaining encrypted rules depending on rules from middle box and private key of end points. Second class, on the other hand support all types of DPI applications especially which contain regular appearance and scripting. In this model, use of Probable Cause Decryption a technique that middle box had the facility to access decrypted individual packet (fi and only if the packet contains a known attack key word). Middle box cannot get an access to any packet doesn't contain any known attack word.

VII. DPI Applications

A. Network Security

In present, there is a speedy development in network applications such as online banking e-commerce online shopping and e-government. All these applications form new challenges to the hackers trying to steal sensitive data and profits by hacking remote computer or systems by developing complicated malware hacking such as viruses trojan and different malicious codes. DPI prepared network operators to distinguish different signatures of attacks before it affects any of network user or system by combining primitive IDS (inspect threats and activates alert) and new IPS (study measures to prevent probable threats). Also, there is data loss prevention (DLP) that prevent important data from leaving the intranet. DLP occupy DIP to discover confidential information covered in network streams then prohibit them from outgoing the organizational boundary via vary communication protocol. Other branch is network forensics which care about latching and storing information of faults and instructions for authorized sign or different shapes of regulation implementation achievement [4].

B. Bandwidth Management:

Bandwidth management is procedure of controlling the use of network connections to prevent decreasing network performance or network bottleneck. Bandwidth management choose the right mechanism such as traffic shaping, schedule algorithm and congestion revocation depending on traffic species that used to categorize the traffic into different classes based on distinguish protocols or applications. DPI to apply bandwidth management uses what known with prioritizing real time interactive application and blocking the harmful connections. Using DPI in bandwidth management in two major applications, First application P2P traffic management. P2P is considered a threat because of overwhelming the network connections, that causes some of latency services such as VoIP in critical case. These problems increase with the growth of HD streaming videos and transferring huge files such as live streaming media and on-demand streaming media. DPI solve these problems by giving latency services higher priority than others when the network faces heavy load. Second application is quality of service (QoS) guarantee, it is about ISP appealing users to subscribe the wanted services by assuring them guaranteed specific level of quality [4][26].

C. User Profiling / Ad injection

Ad injection is pushing different advertisements to whom may concerned.ad injection depends on studying and knowing the profile and the parts of interest, by monitoring And analyzing the user data and different sites that had been visited by him. That causes many companies and sites to inject ad to users by studying their profile such as Google, Facebook and others . ISP had the ability to check every packet that are injected or passing through streaming devices to produce complete the profiles of different users with DPI. Nevertheless, ad injection influenced user browsing security and privacy [4].

D. Copyright Enforcement

The rapidly growth in file sharing applications push copyright possessors such as Sony BMG and Universal music to find provision from ISP for the enforcement of copyright. The use of DPI will be in inspecting if any of the files in the streaming flow are copyrighted files. DPI can't apply techniques such as traditional bit matching or hash methods for copyright protection. Audible Magic solved this issue by applying a fingerprinting technology, rights holders using this software produce a unique signature to all the materials that are protected and kept in registry. On the other side, DPI calculate the fingerprint for each passed packet and compare it with the set of fingerprinting database registered by right holders [4] [31].

E. Governmental surveillance

Most of the governments in the world requires from ISP surveillance or detection abilities for law enforcement units and national security agencies. The work of DPI is to make full scale surveillance by capturing and examining all packets passed through streaming flow in the network and inform the government. In US National Security Agency (NSA) deployed PRISM to gather all information and activities of specified person. On the other hand, Government content censorship is a DPI application that only catch the illegal packets passed through network concerned to the transmitted content. For the reason of high cost of real content based censorship, URL is the best used of censorship [4] [26].

F. Content regulation

This application uses DPI as a recognition system to identify harmful or illegal content to be blocked. This blocking can be one of two types either narrowly constrained or broadly to anything threating government [31].

G. Statistics

DPI permits ISP to collect important information about traffic monitoring and statistical analysis of the traffic . These information helps in network planning, take the right decisions and planning new services [31].

VIII. Conclusion

This paper introduces a survey on DPI. Deep packet inspection plays a vital role in network security; it detects any malicious packet by using matching algorithms as described before in section V. This survey includes some challenges for DPI. It surveyed many techniques used in deep packet inspection and showing the future work handle on these

techniques. Also, it surveyed many applications that make benefits of deep packet inspection.

ACKNOWLEDGMENT

The authors of this paper hope that this work would be useful for researchers interested to start a research work in the interesting research area of deep packet inspection or for engineers who interested to provide their knowledge with the benefits and the terminology of the future network security and network managements.

REFERENCES

[1] Snort v2.9.9(2016). [Online]. Available: http://www.snort.org/

[2] Bro Intrusion Detection System. (2014). [Online]. Available: http://www.bro.org/

[3] Application Layer Packet Classifier for LINUX. (2009). [Online]. Available: http://l7-filter.sourceforge.net/

[4] Chengcheng Xu, Shuhui Chen, Jinshu Su, S.M. Yiu and Lucas C. K. Hui, "A survey on Regular Expression Matching for Deep Packet Inspection: Applications, Algorithms, and Hardware Platforms", IEEE Communications surveys and Tutorials, Vol. 18, No. 4, 2016

[5] A. V. Aho and M. J. Corasick, "Efficient string matching: An aid to bibliographic search," Communications of the ACM, vol. 18, no. 6, pp. 333–340, 1975.

[6] Thienluan, seung rohk and Hyunjin KIM, "PAC-k: A Parallel Aho-Corasick String Matching Approach on Graphic Processing Units Using Non-overlapped Threads", IEICE Transactions on Communication, Vol.E99-B, No.7, pp. 1523-1531, july 2016

[7] C.H. Lin, C.-H. Liu, L.-S. Chien, and S.-C.Chang. "Accelerating pattern matching using a novel parallel algorithm on GPUs", IEEE Transactions on Computers, Vol 62, no.10, pp.1906-1916, 2013

[8] S. Wu and U. Manber, "A fast algorithm for multi-pattern searching," Dept. Comput. Sci., Univ. Arizona, Tucson, AZ, USA, Tech. Rep. TR-94-17, 1994.

[9] Vasudha Bhardwaj and Vikram Garg, "A Comparative Study Of Wu Manber String Matching Algorithm and its Variations", International journal of Computer Applications, Vol 132, No.17, December 2015

[10] C. Allauzen and M. Raffinot, "Factor oracle of a set of words," Institute Gaspart-Monge, University de Marne-la-vallee, Champs-sur-Marne, France, Tech. Rep. TR-99-11, 1999.

[11] M. Al hisnawi and M. Ahmadi, "Deep packet inspection using quotient filter", IEEE Communications Letters, Vol 20, No.11, November 2016

[12] M. Kim, K. H. Oh, H. Y. Youn, and S. W. Lee, "Enhanced dual Bloom filter based on SSD for efficient directory parsing in cloud storage system," in Proc. Int. Conf. Comput., Netw. Commun. (ICNC), pp. 413–417, 2015.

[13] S. Dharmapurikar, P.Krishnamurthy, Todd S. Sproull and John W. Lockwood, "Deep packet inspection using Parallel Bloom Filter", 11th symposium on high performance interconnects, Vol. 24, No.1, 2004

[14] S. Pontarelli, P. Reviriego and Juan A. Maestro, "Improving counting Bloom filter performance with fingerprints", Information Processing letters, Vol 116, No. 4, pp. 304-309, Nov 2015

[15] Taskin Kocak and Ilhan Kaya, "Low-power Bloom filter architecture for deep packet inspection", IEEE Communications Letters, Vol. 10, No.3, March 2006

[16] M. Al hisnawi and Mahmood Ahmadi, "Deep packet inspection using Cuckoo filter", Annual conference on new trends in information and communications technology applications, march 2017

[17] Ya-Ru zhan and zhao-shun wang, "Deep packet inspection based on many-core platform", journal of computer and communications, Vol. 3, pp. 1-6, may 2015

[18] Jung-Sik Sung, Seok-Min Kang, Youngseok lee, Taeck-Geun Kwon, and Bong-Tae Kim, "A Multi-gigabit Rate Deep packet inspection Algorithm using TCAM", IEEE Global Telecommunications conference GLOBECOM, 2005

[19] Tran Ngoc Thinh, Tran Trung Hieu, and Surin Kittitornkun, "A FPGA-based deep packet inspection engine for Network intrusion Detection System", in 9th IEEE international conference electrical engineering/electronics, computer, telecommunications and information technology, 2012

[20] T.N. Thinh and S. Kittitornkun "Massively Parallel Cuckoo Pattern Matching Applied for NIDS/NIPS", IEEE Int. Symp. on Electronic Design Test & Applications, pp. 217-221, 2010

[21] Yunchun Li and Rong Fu, "An Parallelized Deep Packet Inspection Design in Software Defined Network", 2nd international conference on information technology and electronic commerce (ICITEC), 2014

[22] K Kuang, Jilong, Laxmi Bhuyan, Haiyong Xie, and Danhua Guo. "EAHRW: An Energy-Efficient Adaptive Hash Scheduler for Stream Processing on Multi-core Servers", 2011 ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems, 2011.

[23] Tommy Chin Jr, Xenia Mountrouidou, Xiangyang Li and Kaiqi Xiong, "Selective Packet Inspection to Detect DoS Flooding Using Software Defined Networking (SDN)", IEEE 35th international conference on distributed computing systems workshops, 2015.

[24] Huawei Huang, Peng Li and Song Guo, "Traffic scheduling for Deep Packet inspection in software defined networks", special issue paper, Vol. 29, No. 16, august 2017.

[25] B. Renukadevi and Daniel Madan Raja, "Deep packet inspection management application in SDN", 2nd international conference on computing and communications technologies (ICCCT), 2017

[24] Huawei Huang, Peng Li and Song Guo, "Traffic scheduling for Deep Packet inspection in software defined networks", special issue paper, Vol. 29, No. 16, august 2017

[25] B. Renukadevi and Daniel Madan Raja, "Deep packet i

[26] Deep Packet Inspection- wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Deep_Packet_Inspection

[27] Tamer Abuhmed, Abdelaziz Mohaisen and DaeHun Nyang, "A survey on Deep packet inspection for intrusion detection systems",Magazine of Korea Telecommunication Society, vol 24, No. 11, pp.25-36, March 2008

[28] Rafael Antonello, Stenio Fernandes, Carlos Kamienski, Djamel Sadok, Judith kelner, Istvan Godor, Geza Szabo and Tord Westholm, "Deep Packet Inspection tools and techniques in commodity platforms:challenges and trends", journal of network and computer application, Vol. 35, No. 6, pp. 1863-1878, November 2012

[29] Ajay chaudhary and Anjali Sardana, "Software Based Implementation Methodologies for Deep Packet Inspection", International Conference on information Science and Applications (ICISA),IEEE, 2011

[30] Justine Sherry, Chang Lan, Raluca Ada Popa and Sylvia Ratnasamy, "BlindBox: Deep Packet Inspection over Encrypted traffic", ACM SIGCOMM Computer communication review, Vol. 45, No. 4, pp. 213-226, October 2015

[31] Ralph Bendrath, Milto Mueller , "The end of the net as we know it? Deep packet inspection and internet governance", SAGEPub Journals, Vol.13 , No.7 , pp. 1142-1160, April 2011