Implementation of Aho-Corasick String Matching Algorithm on FPGA

BY

USAMA BIN ZAHID

01-244181-025

SUPERVISED BY

DR. ATIF RAZA JAFRI

Session-2020

A Report submitted to the Department of Electrical Engineering

Bahria University, Islamabad

In partial fulfilment of the requirement for the degree of MS (EE)

CERTIFICATE

We accept the work contained in this report as a confirmation to the required standard for the partial fulfilment of the degree of MS(EE).

Head of Department	Supervisor
Internal Examiner	External Examiner

DEDICATION

This thesis is dedicated to my wonderful parents and family who have been always a great source of encouragement and support.

DECLARATION OF AUTHORSHIP

I hereby declare that content of this thesis is my own work and that it is the result of work done during the period of registration. To the best of my knowledge, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Parts of this thesis appeared in the following publications, to each of which I have made substantial contributions:

(Student Signature)

ACKNOWLEDGEMENTS

First and foremost, I offer the gratitude to Allah Almighty who bestowed me with all the wisdom, strength and ability to carry out this work. Then I am thankful to my thesis supervisor Dr. Atif Raza Jafri Dean of the department of Electrical Engineering at Bahria University Islamabad for all his assistance, guidance and encouragement. Last but not the least I must express my deepest gratitude goes to my parents and family for their constant and unconditional support and prayers.

ABSTRACT

A review of the existing pattern matching algorithms shows that the software-based solutions pertaining to pattern matching do not encounter today's throughput network systems. Pattern matching is used to examine Ethernet packet contents against thousands of predefined malicious or suspicious patterns. To accelerate the throughput of pattern matching architectures, hardware-based solutions are getting more popularity. In this thesis, pattern matching architecture of open source network intrusion and preventions system (Snort) is proposed and implementation on FPGA using Aho-Corasick algorithm. According to Aho-Corasick algorithm, while matching input string in one pass there are three possible transition states i.e. Goto, Failure and Output. The Aho–Corasick algorithm used for pattern matching of snort IP, HTTP and TCP packet keywords considering a standard Ethernet packet size of 1500Bytes. The achieved results are evaluated on Xilinx (ISE) design suit tool which indicates that throughput and number of rule sets in the projected mechanism is higher as compared to other approaches. Many previous works have been proposed in this domain, however, solutions for limited rules have been discussed. Moreover, rule set level parallelism study while considering trade-off among resource utilization, operational frequency and resulting throughput has not been discussed. In this thesis we have presented the results of parallel implementation of a rule sets while dividing rule set into small sub-sets. A comparison of FPGA resources, operation frequency and throughput is also presented to evaluate parallelism efficiency of proposed architecture. It has been shown that throughput increases upto 27% by dividing rulesets into small subsets.

TABLE OF CONTENTS

LIST OF FIGURES

List of Tables

ABBREVIATIONS

IDS	Intrusion Detection System
NIDS	Network Intrusion Detection System
HIDS	Host Intrusion Detection System
GPP	General Purpose Processing
ASIC	Application Specific Integrated Circuit
FPGA	Field Programmable Gate Array
DDOS	Distributed Denial of Service
FSM	Finite State Machine
NFA	Non-Deterministic Automaton
DFA	Deterministic Finite Automaton
BSIC	Byte Shift invariant Code
LUT	Lookup Table
HDL	Hardware Descriptive Language
RTL	Register Transfer Level
AC	Aho-Corasick
IP	Internet Protocol

HTTP	Hypertext Transfer Protocol
TCP	Transmission Control Protocol

Chapter 1

Introduction

CHAPTER 1. INTRODUCTION

The increasing intricacy of network traffic and increasing number of attacks over the internet cause various security threats. An intrusion detection system (IDS) is one of the choice toward identify the different security threats [1] . The term intrusion is an unauthorized entry into the internet network. A software application or any communication device that have capability to monitor communication devices or incoming networks traffic against the malicious activities, is commonly known as an intrusion detection system (IDS) [2]. There are commonly two types of IDS, i.e. Network intrusion detection system (NIDS) and Host intrusion detection system (HIDS). Out of these two, NIDs monitors the incoming traffic from the internet source while operating system files are monitored by the HIDs. The focus of this work will be on implementation of NIDs.

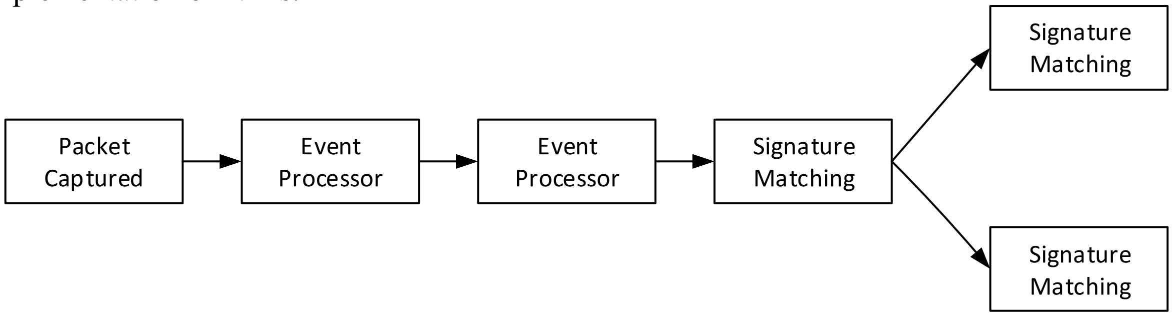


Figure 1.1 Mechanism of Signature-Based IDS

IDS have been more and more used through communities round the globe to discover and defend against attacks as they are free of charge and involve an extensive municipal of specialists. However the multiplied rapidity of communication, the pace at which attacks are performed and complexity of latest threats make detecting and vindicating them very difficult.

1.1. Types of IDS

There are three types of IDs are available in literature i.e. signature based, anomaly based and host based [2]. Signature based IDS refers to the detection of attacks by looking for specific patterns, anomaly based IDS mention to the monitoring of both network and computer intrusions and searches for any abnormality occurs in the specific activity, whereas host based IDS refers to the monitoring of the internals of the computing system.

1.1.1. Signature Based IDS

This IDS possess an attacked description that may be matched to detected attack manifestations. The question of what data has relevancy to associate IDS depends upon what it's making an attempt to observe. E.g. DNS, FTP etc. IDS is configured to infer a definite sequence of packets, or a definite part of knowledge confined in those packets. For instance, associate IDS that watches internet servers can be programmed to appear for the string "phf" as an indicator of a CGI program attack. Most signature analysis systems are primarily based of easy pattern matching algorithms. In general, the IDS merely appearance for a sub string inside a stream of knowledge carried by network packets.

1.1.1.1. Drawback of Signature-Based IDS

- They are incompetent to observe innovative threats.
- False alarms are involved.
- Need to reconfigure once more for each new pattern to be detected.

1.1.2. Anomaly-Based IDS

In this type of IDS, everyday utilization of network is monitored. Anything distinct from regularity is assumed to be an interruption action. E.g. overflowing a host with excessive packet. The predominant strength is its capability to identify original attacks. Anomaly detection strategies expect that all intrusive actions are essentially anomalous.

1.1.2.1. Statistical Methodologies

Primarily, conduct outlines for topics are created. As the device remains running, the abnormality indicator continuously creates the alteration of the current profile from the authentic.

1.1.2.2. Predictive Approach

This technique of IDS attempts to envisage coming actions on the basis of activities that have previously happened. The hassle with this is that some interruption situations that are not defined through the rules will now not be labelled intrusive.

1.1.2.3. Weaknesses of Anomaly Detection IDS

- Capability to be misled by a suitably provided threat.
- Generation of false alarms degrade its efficiency.

1.1.3. Host Based IDS

The host working framework records in the review information. These review material includes occasions just like the utilization of distinguishing proof and verification methods i.e. account logins, opening any file and execution of program and administration exercises. This inspection is then examined to discover record of intrusions.

1.1.3.1. Weaknesses of the HIDS:

- The data required to be monitored in is a material of knowledge.

- Random sorting of messages may significantly rise the assessment and investigation loads.
- Careful monitoring runs the danger that threat appearances might be overlooked.

1.1.3.2. Advantages of the HIDS:

- Threat authentication.
- Checking significant modules.
- Near Instantaneous recognition and reaction.
- No surplus hardware required.

1.2. NIDS and its Modules:

A NIDs is comprises of four modules i.e., 1) packet capturing, 2) packet decoder, 3) packet pre–processing and 4) pattern matching [3]. In terms of hardware implementations, the packet capturing module is responsible to hold incoming network traffic while packet decoder is responsible to split packet header and payload. The pre–processing module is required to organize incoming packets for the pattern matching module. Finally, the pattern matching module is responsible to take decision either the packet will be accepted or rejected.

Intrusion detection structures do have a number of known challenges that may be greater work than an organization is inclined or in a position to tackle. False positive (i.e., producing signals when there is no real problem). "IDSs are infamous for generating false positive" Researcher said, adding that indicators are typically are dispatched to a secondary analysis platform to help contend with this challenge. This undertaking also places pressure on IT teams to always update their IDSs with the right data to become aware of professional threats and to distinguish these actual threats from allowable traffic. It's no small task, professional said. "IDS structures need to be tuned by IT administrators to analyse the desirable context and limit false-positives. For example, there is little benefit to examining and presenting signals on net activity for a server that is covered in opposition to known attacks. This would generate thousands of beside the point alarms at the price of elevating meaningful alarms. Similarly, there are occasions where flawlessly legitimate things to do may generate false alarms really as a depend of probability," Researcher said, noting that businesses regularly decide for a secondary analysis platform, such as a security Incident and Event Management (SIEM) platform, to assist with investigating alerts.

1.2.1. Recruitment

Given the requirement for understanding context, an organization has to be ready to make any IDS match its very own special needs, experts advised. "What this capability is that an IDS can't be a one-size-fits all configuration to operate precisely and effectively. And, this requires a savvy IDS analyst to tailor the IDS for the pursuits and desires of a given site. And, knowledgeable trained system analysts are scarce".

1.2.2. Missing a respectable threat

"The trick with IDS is that you have to comprehend what the attack is to be capable to discover it. The IDS has usually had the patient zero problem: You have to have located any individual who acquired in poor health and died before you can discover it," Hanselman said. IDS technological know0how can also have trouble detecting malware with encrypted traffic, specialists said. Additionally, the velocity and distributed nature of incoming site visitors can limit the efficiency of an intrusion detection system in an enterprise.

1.3. Difference between Firewall and IDS

IDS are a devoted and collaboration utilized to screen the critical framework. Today's security structure is becoming extremely complex, it includes firewalls, reconnaissance and verification methods, access management, encoding methods, scanning Tools, and more.

There are some difference between and IDS and Firewall listed below.

1. Firewall is a network safety scheme that cleans arriving and departing network stream based on determined rules, while an IDS is a scheme that screens a traffic for suspicious action or rule disruption and refers alarm upon recognition.
2. Firewall screens network stream based on IP address and port number, while IDS notices incoming traffic and looks for traffic arrangement or patterns of attack and then cause alarms.
3. In an organization firewall would be first line of resistance while IDS would be positioned after firewall.
4. Firewall will stop the stream upon illegal traffic recognition whereas IDS created alerts/alarms on recognition of irregularity.

Pattern matching is an art of comparing a set of incoming streams of characters with the elements of stored patterns in the data sets. Multiple platforms have been used for the implementation of pattern matching algorithms. The most commonly used are central processing units (CPUs), General Purpose Processors (GPPs), Application Specific Integrated Circuits (ASICs) and FPGAs [4]. Both, CPUs and GPPs are flexible but with limited throughput. ASICs provide higher throughput with no flexibility. Besides all these, FPGA is an alternative to achieve both flexibility (by its inherent re-configurability feature provision) and throughput [4] [5].

1.4. Motivation

The inspiration of this examination originated from the modern difficulties of delivering a security framework that would guarantee accessibility, privacy and respectability in respects of the present province of Internet security. Following quite a long while of modern experience and keeping in mind that taking a shot at the plan of a security framework for a 10GB apparatus, some intriguing thoughts brought up not many issues marks. An ongoing move by trend-setters to equipment based IDS suggests a rapid and a great many packets prepared simultaneously. In any case, equipment based arrangements have restrictions in their capacities to execute specific programming capacities. For example, there is no normal articulation framework completely executed in equipment. Right now, it is observed that 65% snort rules are with normal articulation. Furthermore, equipment based IDS have a great deal of area issues as there is not really any allocation of dynamic memory. While the equipment improves speed, the product execution of IDS has more highlights despite the facts that there are at present numerous threats that go undetected. Verifying Internet frameworks is to an ever increasing extent testing. Likewise, a great number of organizations depend on IDS and IPS to ensure their frameworks [5].

After examination, apparently a hole is still persists to fill equipment based IDS convey speed however are over the top exclusive, programming arrangement, despite the fact that there is no solution to tackle speed offered by the equipment IDS, they can bolster a more extensive scope of functionalities. From these ends, the creator begun taking a shot at a framework that won't just build the bundle handling rate yet in addition offer further Examination so as to empower the most extreme identification furthermore, moderation conceivable. So we have to add-on additional hardware to improve working capabilities of snort rules. Furthermore we will discuss in details about this solution in next chapters. Then again, this thesis has a solid enthusiasm for adding to the open source as it is a domain where specialists encounter, talk about and cooperate. In such manner, a portion of our work has delivered an IP & HTTP and TCP code which basically used to monitor the performance of confrontation of IDS against scrap sources. The beginning of this thesis will be the point of view of having a framework that works for example a framework that can adapt to ongoing threats and innovation headway, a framework that produces less bogus positive while improving the discovery rate.

1.5. Problem Description

Throughput enhancement and memory decrease are two central point donating in execution assessment of matching a pattern. This research objectives is to making a structure for Network Intrusion Detection Systems that is proficient to remain against the most harmful efforts handled by the Internet today for example DDOS threats and parallel stages attacks for example assaults dispersed among Packets. The new structural designs intends to be a multi-dimensional parallel structure that will utilize promptly accessible modules for example broadly accessible programming and hardware equipment.In endeavoring to create such a framework, the accompanying destinations would be accomplished:

1. Design and actualize IDS based Snort utilizing Aho-Corasick approach.
2. Identify Snort explicit search engine part shortcomings
3. Yield a framework that will recover the recognition frequency while diminishing the false alarms.
4. Keep fully informed regarding industrial progression by proposing a redundant Implementation of our design.
5. Provide the cutting edge of Internet Security

For implementing the traditional Aho–Corasick algorithm, DFA creation is required for pattern matching. The DFA requires one transition state for single character matching. Therefore, the number of transition states will be increased as the number of required matching characters are increased which incurs the higher memory cost. For Snort rule sets pattern matching, the throughput optimization of the Aho–Corasick algorithm is highly desirable. This research will provide the hardware architecture for pattern matching of open source network intrusion prevention system snort datasets using the standard Ethernet packet size of 1500Bytes in a real time environment.

1.6. Thesis Objectives

In this thesis, Hardware implementation of pattern matching Aho-Corasick algorithm based on finite state machines (NFA & DFA) is proposed. Because of increasing network bandwidth in advances digital communication technologies, demanding hastening of data monitoring to withstand the throughput. Details of the proposed architecture is presented in chapter 3.

The rule set in open source network intrusion and prevention system explored and matched using AC algorithm. By reducing number of transition states throughput as well as memory optimization achieved.

1.7. Thesis Organization

Introduction chapter describes importance of network intrusion detection system, its types and mechanism of signature-based IDS. The remaining portion of thesis is organized as follows. In chapter 2 some well-known network security techniques and algorithms are explained. In chapter 3 explains the mechanism of proposed architecture based on Aho-Corasick Algorithm. Achieved results and simulations are analyzed and compared with existing solutions in chapter 4. Future work and conclusion is presented in chapter 5.

Chapter 2

Literature Review

CHAPTER 2. LITERATURE REVIEW

For pattern matching, the most commonly used technique is finite state machine (FSM) [6], [7], [8], [9], [10], [11], [12] .FSM consists of two types i.e., deterministic finite automaton (DFA) and non-deterministic finite automaton (NFA). In DFA, one transition state is required to process single character matching while multiple states are required to process single character matching in NFA. For multi-character matching, the number of transition states are mainly depending on number of characters. To implement pattern matching algorithms, various hardware architectures have been reported in literature

2.1. Research Method

This examination has been executed utilizing the orderly writing audit strategy. The appropriate information or data in a particular research area is described presenting to an exploration rules for investigating the momentum inquire about examinations significant to the built questions. In this way, the classifications definition and research standards are depicted in Sec. 2.2 and Sec. 2.3 individually.

2.1.1. Categories Definition

So as to sort out the chose research examines, we have characterized five classifications. This classification increment the exactness of the reactions to build questions, created in Sec. 1. Therefore, characterization subtleties are given in the accompanying:

2.1.2. General Class

There can be a few models where the parcel catching, bundle translating and occasion location modules are all the while talked about in a solitary equipment arrangement. For instance, the arrangements appeared in [6] talk about the data capturing, translating and occasion identification/motor modules without depicting the bundle pre-handling module. There may be various models where suspicions are taken out on approaching arrangement of characters from Gigabit organize for data capturing. Along these lines, all the previously mentioned designs will be fused right now.

2.1.3. DFA Class

Finite state machine is one of the unmistakable method, utilized in the occasion recognition/motor module, for design coordinating and is computationally serious for the advancement of NIDS. For the most part, it performs like a finite state machine (FSM) to perceive the put away examples inside the approaching floods of info characters. The calculation of FA for the most part relies on its five tuples: (1) calculations of finite states, (2) arrangement of characters, (3) Forward states (4) initial state and (5) last state. It establishes two unique sorts: deterministic limited machine (DFA) and non-deterministic limited automaton (NFA).

DFA, needs just one change state to process each info character for the example coordinating and requires higher measure of memory to keep each progress state. DFA based example coordinating calculations and strategies are helpful to accomplish higher throughputs with the cost of higher memory uses. In any case, various equipment structures have been accounted for to diminish the requirements of memory while utilizing the DFA mapping. For instance, a productive usage of the FPGA look-into tables (LUTs) for mapping single character and change states are appeared in [8], [9], division of bigger examples sets into littler sets with one of a kind and non-one of a kind examples is exhibited in [13] and bit-parting design to diminish the quantity of progress edges in each state is appeared in [14] and so on. Every one of these designs will be included right now.

2.1.4. NFA Class

NFA based example coordinating calculations and methods are memory proficient yet then again they give restricted throughput because of the contribution of numerous states per input character.

Different models have been accounted for to enhance the throughput of the NFA based example coordinating calculations and procedures. An engineering for making an interpretation of the information design into a NFA and afterward encoding utilizing bit covering is introduced in [13]. In addition, parting the bigger examples into the littler sets is appeared in [2]. Every one of these models [10], [6], [11] will be combined.

2.1.5. Hash Classification

Mapping of variable length of info information onto a fixed length is known as hashing. There are various models, where hashing strategy have been used, either on the approaching information bundles from the Ethernet [7] or on the put away factor design lengths into the memory [3].

2.1.6. Tree Classification

By and large, tree alludes to the information structures that have an ability to store the approaching floods of characters. A few designs have been proposed to store the sub word reference of fixed length designs into the information structures as utilized in [2]. Besides, to diminish the memory necessities, different models have been viewed as where the bigger example sets were partitioned into gatherings of littler sets and the littler assembled sets are put away in the information structures as introduced in [3].

2.2. Overview of Pattern Matching Techniques

Various architectures and systems for pattern matching are categorized by characterized classifications i.e., general, DFA, NFA, hash and tree. It is critical to have any kind of effect that a system is a combination of rule sets to a particular issue where the information and the result are officially indicated. There are generally two types of string matching algorithm and techniques in literature review i.e string based and regular expression based techniques. Explanation of both types are as follows:

2.2.1. String Based Techniques

There are 27 chosen reads for string coordinating calculations and procedures. Out of these 27 chose considers, 23 investigations send design coordinating calculations while staying 4 examinations use diverse example coordinating strategies. As discussed the recognized string coordinating calculations are Aho–Corasick, Knuth Morris Pratt, Bowyer–Moore, State–Traversal, Shift–And , Bit–Split, Bit–Parallel, Bloom–channel, Pattern Matching, Cuckoo and BST. The distinguished diverse example coordinating strategies are Diode–Resistor rationale utilizing CMOL, merged state change approach, Automaton mapping and Range tree connect list. In the accompanying, the distinguished string coordinating calculations and methods are additionally depicted by the characterized classifications.

For the general class, the Aho–Corasick calculation is utilized in [1] and [8], KMP calculation is considered in [2] , Booyer–Moore calculation is used in [3] , State–Traversal calculation is utilized in [5] and Shift–And calculation is executed in [4], [6] and [7]. A high throughput design for string coordinating utilizing Diode–Resistor rationale with sub-atomic FPGA (CMOL) innovation is exhibited in [9]. The run time rule-reconfigurable engineering, exhibited in [1] diminishes the framework vacation at whatever point the standard set is required to be refreshed. Multi organize pipelined engineering for the execution of Aho–Corasick calculation utilizing fast inspecting by methods for on request confirmation approach is accomplished in [8]. A versatile quickening agent, appeared in [2], is progressively arranged towards the force enhancement of string coordinating. For transmission control convention (TCP) parcels, a continuous on-line string coordinating is performed utilizing a framework on chip (SoC) stage in [3]. As far as memory improvement, the arrangement, appeared in [4] utilizes the division of long example sets into littler sets. The order and gathering of homogeneous traffic, and afterward forward it to the pertinent equipment hinder for handling is considered in [6]. A multi character string coordinating arrangement utilizing location and revision of string arrangement is introduced in [7]. Towards memory enhancements, one of the fascinating answer for string coordinating is appeared in [5].

For DFA classification, Aho–Corasick calculation is executed in [10] and [13] while the Bit–Split calculation is considered. Towards memory advancements of the Aho–Corasick calculation, the failure-less pipelined DFA has been built in [10], decrease of state chart to a character trie has been considered in [11] and productive usages of FPGA LUTs are focused in [13]. Memory advancements of the Bit–Split calculation is appeared in [11] and [12]. A gathering of longer example sets into a few sub patterns with a fixed length is considered in [12] though the gathering of one of a kind and non-one of a kind sub patterns are focused in [11].

For NFA class, Aho–Corasick calculation is used in [11], Shift–And calculation is focused in [8] and Bit–Parallel calculation is considered in [9]. The half and half arrangement utilizing both NFA and DFA in [12] is exhibited by investigating multi characters in equal which acquires arrangement issue. Thus, a security arrangement regarding right hand advances for the arrangement issue is introduced in [2]. For Shift–And calculation, a pipelined answer for advance throughput of the example coordinating is appeared in [7]. Utilizing Bit–Parallel calculation, a powerful reconfigurable engineering for the specific and expanded example sets is considered in [6] . Consolidated state change approach utilizing the mix of both shared regular prefixes and no disappointment advances are conveyed in [5]. For organize security and bioinformatics applications, a finite automata is developed.

For the hash classification, Bloom–Filter [12], Pattern Matching [11] calculations are considered. The reconfigurable arrangement, exhibited in [3], is increasingly situated towards the area and power advancements. The Pattern Matching calculation, proposed in [8], performs content coordinating utilizing some factor design lengths. An exceptionally versatile and power proficient string coordinating arrangement, is considered in [5]

2.2.2. Regular Expression Based Techniques

We have recognized 22 investigations for normal articulation coordinating calculations and methods, as appeared in Table 4. Out of these 22 chose contemplates, 11 examinations use design coordinating calculations while the remaining 11 investigations use diverse example coordinating strategies. The pre-owned calculations for the usage of ordinary articulation coordinating are Parallel Multi–Stride, Left–most, J–DFA, Parallel–DFA, Multi–Stride, Shift–And, Gluskhov–

NFA, MIN–MAX, YARA and Bloom–channel. Other utilized methods for normal articulation coordinating incorporates the Memristor crossbar approach, Tiled ordinary articulation, Deterministic robot mapping, Finite machine mapping, Run length encoding, Distributed information in cooperative effort, Simple state consolidate tree, Multi–way theory, Non–deterministic trie mapping, Spatial–Stacking system and Token based coordinating. As per the characterized classes, the distinguished calculations and methods for ordinary articulation coordinating are given in the accompanying.

For general class, the used example coordinating calculations are Parallel Multi–Stride and Left–most. To enhance three plan parameters (territory, throughput and force) simultaneously, a Memristor crossbar based methodology is also utilized in regular expression type pattern matching. A tweaked arrangement as far as versatility for the RegEx coordinating is appeared in [12]. Finite state machine (FSM) based arrangements are acquired in [9] where DFA and limited automaton (FA) are developed. For rapid interruption recognitions, a mix of Multi–Stride calculation with the FA strategy is converged in [16]. One of the fascinating work is considered in [4] where furthest left coordinating of RegExs are performed by using the equipment assets effectively.

For DFA classification, J–DFA, Parallel–DFA and Multi–Stride design coordinating calculations are used. Towards, memory streamlining of the DFA, run length encoding strategy is used in [34]. For NIDS, three unique methods i.e., distributed information in cooperative effort, Simple state consolidate tree and Multi–way hypothesis are coupled to fulfill 100G Ethernet guideline. Another intriguing arrangement is introduced in [13] for the 100G Ethernet standard. The architectures are available in regular expression based pattern matching are incredibly worried about the memory enhancements of DFA.

For NFA class, the used example coordinating calculations are Shift–And, Gluskhov–NFA, MIN–MAX and YARA system. Towards rapid, rather than byte coordinating in customary procedures, a token based coordinating arrangement is presented. Another arrangement is given by using the pipelining and Spatial–Stacking procedures. Throughput proficient arrangements by utilizing the memory based mapping of NFA are considered. Moreover, a string channel is likewise used to recognize the traversal of a NFA. As of late, for interruption location in the modern substation machine organize (IEC–61850), a reconfigurable arrangement is introduced. The utilization of Shift–And calculation with the priority encoder decreases the enormous number of states while developing the NFA. A unique reconfigurable solution for parallel RegEx coordinating decrease the utilization of over the top memory. A FPGA prototyping of the interruption recognition system/device i.e., yet another Recursive Acronym (YARA) is also presented in literature review. At last, for Hash class, a memory streamlined arrangement utilizing the Bloom–Filter calculation is used.

2.3. Research Standards

In view of above literature review, we have recognized research process to proceed further. The insertion and elimination of input pattern, and also the extraction of useful data from set of rules.

2.3.1. Insertion and avoidance procedures

We delimit some strong standards for the incorporation and prohibition of research works, as appeared in the accompanying:

- **Relevant-to-the-subject:** Contain just those examinations which are relevant to our consideration. It must give adequate subtleties to help the reactions of our questions and ought to be appropriate to one of the pre-built classes. Avoid every one of those examinations which don't identify with any of the predefined classes.

- **Most later (2010–2019):** Selected contemplates must be distributed from 2010 to 2019. Bar every single other examination those are distributed before 2010 to ensure the thought of latest investigations.

- **Publication associations:** Chose just those examinations which must be distributed in any of the five logical databases i.e., IEEE [9], SPRINGER [10], ACM [11] and PLOS [13] through 2010–2019.

- **Hardware based outcomes:** Pick just those examinations which give equipment based answers for the NIDS whichever executed on reconfigurable stage i.e., FPGA or CMOS.

- **Repetition:** All the exploration considers in a particular research setting can't be incorporated. In this manner, avoid those investigations which are comparable in the given research setting and only one of them is incorporated.

2.4. Search Technique

The fundamental perseverance of search technique is to choose the exploration work as per the incorporation and prohibition rules, characterized in Sec 2.2.1. So as to execute the hunt technique, distinctive pursuit terms have been utilized like IDS FPGA, NIDS FPGA, equipment structure NIDS, equipment design NIDS, reconfigurable models NIDS, adaptable engineering NIDS, SoC NIDS, Pattern coordinating NIDS, Signature coordinating NIDS and String coordinating NIDS and so on. So as to choose these hunt terms, a few logical distributions have been arbitrarily chosen and considered. From that point onward, the chose search terms have been utilized to get the potential research considers identified with the interruption identification frameworks. The quest terms close by the results for each logical database are orchestrated in Table 1. Four different databases are used in this work to search keywords. Some conference paper and higher impact journals are included in these databases. These databases are IEEE, Elsevier, Springer and ACM. There are five phases of research work selection. First step is review of Systematic literature review, Second step is the selection of successful and rejected researches. In remaining steps selection of researches is either Title based, Abstract based, General study or detailed study.

Phases	Research discovered per phase	Rejected Researches	Selected Researches	Remarks
1	35	-	-	Search based selection
2	35	5	30	Title based selection
3	30	7	23	Abstract based selection
4	23	4	19	Selection on basis of General study
5	19	6	13	Selection on basis of Detailed study

Table 1 Research work selection process

The stages, executed during the consideration of research works, are appeared in Figure 2.

1. We have proposed different terms in five logical information bases and examined around 35 selected results.

2. Out of 35 researches, we have omitted 5 papers based on their Title according to the consideration and prohibition rules.

3. Out of the outstanding 30 research papers, we have additionally omitted 7 papers based on their Abstract according to consideration and rejection rules.

4. We have performed an overall investigation of the outstanding 23 research papers by perusing diverse important areas of each search. Based on our overall examination, we have avoided 4 research papers don't encounter the incorporation criteria.

5. We have executed a point to point investigation of the outstanding 19 researches and have rejected 6 research papers.

6. Lastly, we have incorporated 13 researches significant to our examination settings.

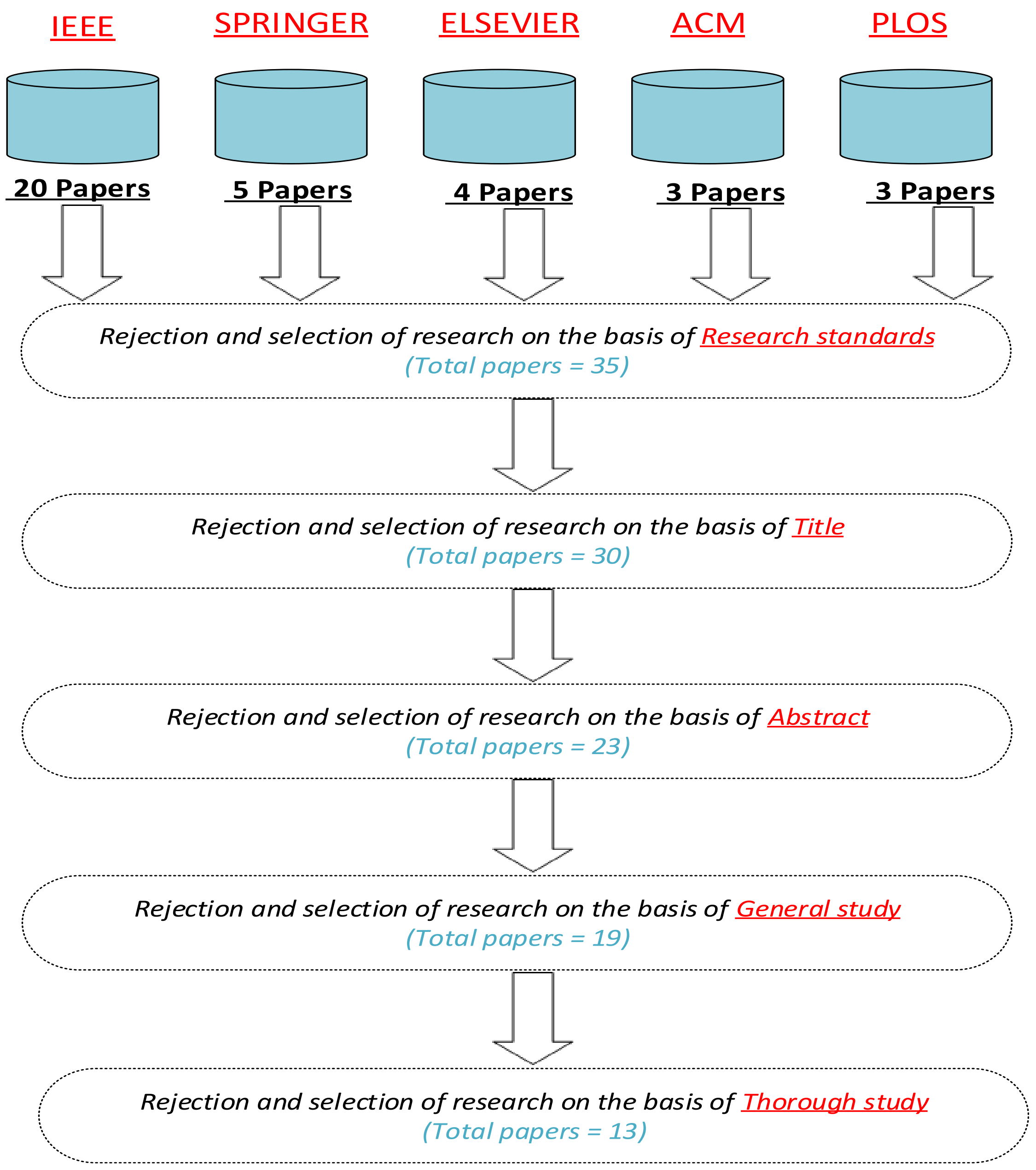

IEEE
SPRINGER
ELSEVIER
ACM
PLOS
20 Papers
5 Papers
4 Papers
3 Papers
3 Papers
Rejection and selection of research on the basis of Research standards
(Total papers = 35)
Rejection and selection of research on the basis of Title
(Total papers = 30)
Rejection and selection of research on the basis of Abstract
(Total papers = 23)
Rejection and selection of research on the basis of General study
(Total papers = 19)
Rejection and selection of research on the basis of Thorough study
(Total papers = 13)

Figure 2.1 Search Technique

2.5. Some overview

Implementation details of hardware architectures reported in literature are summarized in Table 1. As shown in Table 1, the first column provides the serial number whereas the implemented algorithm is shown in column 2 of Table 1. The column 3 of Table 1 is further partitioned into four sub columns for providing the rule set, total number of rules, pattern size and utilized character matching per clock cycle. Similarly, the last column i.e., column 4 of Table 1 is further divided into five sub columns for reporting the targeted platform, throughput (in Gbps), FPGA area (in terms of LUTs, Slices and flip flops), operational clock frequency (in MHz) and reference number.

The most recent solutions, as shown in Table 1, either optimizes throughput or memory of the pattern matching i.e., Aho–Corasick algorithm. The architectures shown in [6] and [12] are more orientated towards the throughput optimization whereas the solutions available in [7], [8], [9], [10], [11] optimizes the memory requirements of the Aho–Corasick pattern matching algorithm. Parallel based architectures introduced to improve the throughput and to optimize memory of proposed architecture by sorting alignment issues and reduction transition states. For Implementation of Aho-Corasick algorithm major contribution is to generate trie diagram of rule sets available in the memory. The peculiarity oriented IDS necessities a specific information on the framework to be secured. In the time frame of learning, the irregularity IDS will assemble sufficient data to frame the reference line, the typical conduct. Notwithstanding profound convention examination, a marginal is at that point characterized as the typical conduct.

Sr. #	Utilized Algorithm	Database related parameters				Hardware related parameters						
		Rule set	# of patterns	Pattern size	Chr/cycle	Targeted Platform	Throughput (Gbps)	FPGA Area			Freq. (MHz)	Ref / Year
								LUTs	Slices	FFs		
1	Aho–Corasick	Snort	16K	200 Bytes	1	Xc7z045	1	190592	–	162378	200	[6] / 2017
2		Clam–AV	82K	–	1	Xc6vsx315t	3	7500	3604	–	230	[7] / 2012
3		Snort	6166	–	2	Ep2s60	1	–	–	–	253	[8] / 2010
4		Snort	2200	–	4	Xc7vx485t	6.9	10315	4093	2713	216	[9] / 2018
5		Backdoor	955	–	1	Xc3s500e	1.536	8985	5191	6914	128	[10] / 2015
6		Snort	1000	–	4	Stratix IV	16.8	102030	–	9403	131.27	[11] / 2013
7		–	300	–	4	Stratix IV	4.3	31258	–	300	67	[12] / 2016

Table 2 Existing Implementations of Aho–Corasick Algorithm on FPGA

For throughput optimizations, a run time rule reconfigurable architecture is proposed in [6]. The term re-configurability means that their proposed architecture does not need reconfiguration whenever updated rules required to add in the rule set. The use of rule base updates, as achieved in [6], causes a downtime problem. To overcome this issue (downtime problem) a stateful packet inspection technique is utilized in [6]. A 2-stage pipelined solution is proposed in [12] where advantages of both DFA and NFA approaches are utilized. Moreover, a multi-character pattern matching is considered in parallel while utilizing promising memory space.

Towards memory optimizations, use of hash tables considering the bit shuffle technique is implemented in [7]. Furthermore, to optimize throughput, a new approach has been introduced using Byte Shift Invariant Code (BSIC). The BSIC consists of suffix sharing concept with the multiple prefix units which results increase in the throughput. Use of DFA results higher throughputs while efficient space utilizations can be achieved by adopting NFA approach. One of the solutions, presented in [8], uses hybrid approach considering combined utilization of advantages provided by DFA and NFA approaches simultaneously. The use of DFA approach in [8] reduces state graph to character trie then its NFA was generated. The term character trie means that the resulting graph after state reduction contains only forward edges/nodes. Moreover, pipelining is also adopted in [8] to improve the operational clock frequency. Another, memory optimized architecture for multi-character pattern matching is shown in [9] where look-up-table (LUTs) based implementation of state transitions are considered. This concept improves the flexibility by mitigating the requirement of hardware reconfiguration whenever new updates are available to add in the rule set. To reduce memory cost in [10], the adopted techniques are 1) by eliminating the failure pointers and 2) sharing the common prefixes with in the DFA. To optimize clock frequency pipelining is adopted in [10]. A string alignment problem occurs when multiple characters for matching are considered in parallel. In order to reduce string alignment problem an assistant transition approach is used in [10] .

The inherent features of architectures reported in [6], [7], [8], [9], [10], [11], [12] are using different characters matching per clock cycle. Therefore, the architectures, shown in [6], [7] and [10] utilizes single character matching per clock cycle. Other solutions, described in [8], [9], [11] and [12] utilizes multi character matching in one clock cycle. All these solutions, presented in [6], [7], [8], [9], [10], [11], [12] provides pattern matching using different packet sizes, such as 200Bytes used in [6]. Respectively, pattern matching on incoming Ethernet packets using a standard size of 1500Bytes is an open research direction.

Chapter 3

Methodology

CHAPTER 3. METHODOLOGY

In this section, current high-tech network intrusion detection results are studied, their drawbacks are highlighted and inspiration for the proposed architecture is described. This section also defines the Aho-Corasick algorithm being used in this work, and how its algorithm tackles the problem stated. Then, difference between Non-Deterministic and Deterministic automaton will explained. After that, open source network intrusion detection system (Snort).

In "An Efficient Multicharacter Transition String-matching Engine based on the Aho-Corasick Algorithm" [11] paper, a real-time Aho-Corasick Non-Deterministic Automaton (AC-NFA) is implemented. The main contribution in [11] is to avoid transition states as in DFA approach the problem is the explosion of transition states. The proposed architecture also resolves the alignment issue. It is noticed that both throughput and hardware cost is directly proportional with the numbers of patterns.

In order to achieve objectives of this work, the essential steps are described as follows:

3.1. Literature Review Part:

AC-algorithm solutions reviewed where standard Ethernet packet size of 1500Bytes are used for pattern matching. Then, their hardware architectures explored to recognize the architectural implementation details relevant to throughput and area optimizations. Based on architectures explained in [6], [7], [8], [9], [10], [11], [12] results are further evaluated based on type of rule set, number of patterns implemented and number of characters matching in one clock cycle. It is noticed that numbers of snort patterns matched in [11] are 1,000 in 131 MHz frequency.

3.2. Architectural Modelling:

After covering the literature part, the next step is the architectural modelling of Aho–Corasick pattern matching algorithm in hardware description language HDL (Verilog). To optimize throughput, AC-NFA approach selected in this work. Primarily, open source network intrusion and prevention system patterns explored for internet protocol (IP), hypertext transfer protocol (HTTP) and Transmission Control Protocol (TCP) packets. Secondarily, C-language implementation of AC-NFA architecture performed and resulting Goto matrix, failure matrix and output matrix achieved which gives information about transition states. Then HDL (Verilog) implementation of pattern matching of snort keywords performed using c-modelling results and synthesizable RTL view evaluated and analysed using Xilinx (ISE) design suit tool. Simulated results show that the output and cost of hardware raise linearly with the respect processed in parallel.

3.2.1. Aho-Corasick Algorithm:

In a matching process the transition states to detect existences of patterns in a string may be goto, failure and output functions. There are two approaches used to implement AC-algorithm i.e. AC-NFA and AC-DFA. In DFA unique next state for respective current state while in NFA there can be multiple next state. As we know that, various transition states cause due to failure functions which makes it problematic to configure AC-Algorithm in DFA.

3.2.2. Aho-Corasick Algorithm with NFA Approach:

This approach primary defines the AC-algorithm and the AC-trie. Aho-Corasick trie converted to NFA by eliminating failure states, known as AC-NFA. The matching algorithm of both tries are also equated.

Figure 3.1 demonstrates an AC-Trie constructed on the rule set {SETID, ODAvM, POST, dump} taken from hypertext transfer protocol snort rules. In this figure, states are represented by circle and output states are denoted by double circle. In addition, goto states are represented by lines and failure states denoted by dashed lines. At any time only one active state is present in Aho-Corasick trie. While matching an input character initially goto functions is monitored. After monitoring if goto function not matched, then through a failure function that states transfers to a new state and checks the goto function of new triggered state. The examining character matched one by one in a matching cycle. Whenever input keyword matched, the resulting output states is non-zero on the other hand output state is zero.

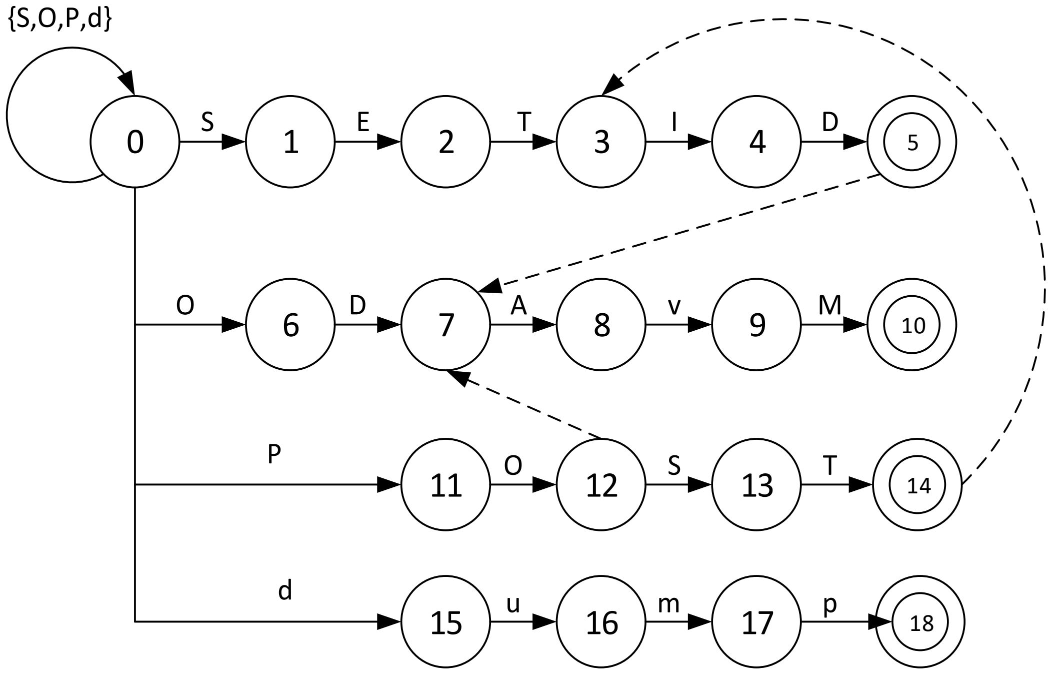
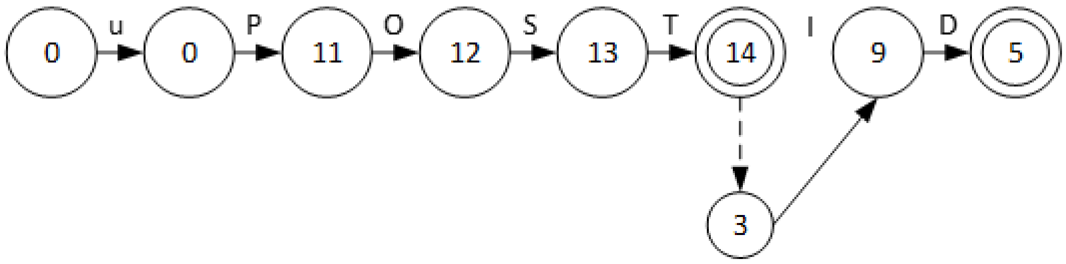


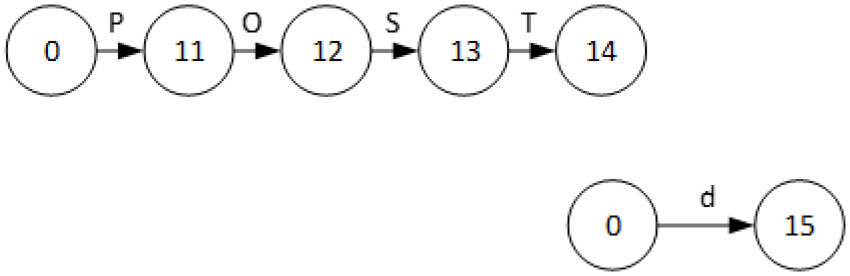
Figure 3.1 AC-Trie for Keywords {SETID, ODAvM, POST, and dump}

Figure 3.2(a) shows the operation of AC-trie matching process. Let’s assume that the input string “uPOST”. Character “u” doesn’t match with “S”, “O”, “P” and “d”. So, state 0 retained. The next four characters “POST”, the states transfers from 0-11 and then from 11-14 consecutively to give the output state. In Figure 3.2(b) matching process of input “uPOSTd” is exemplified.



(a)

Input: **u** **P** **O** **S** **T** **d**

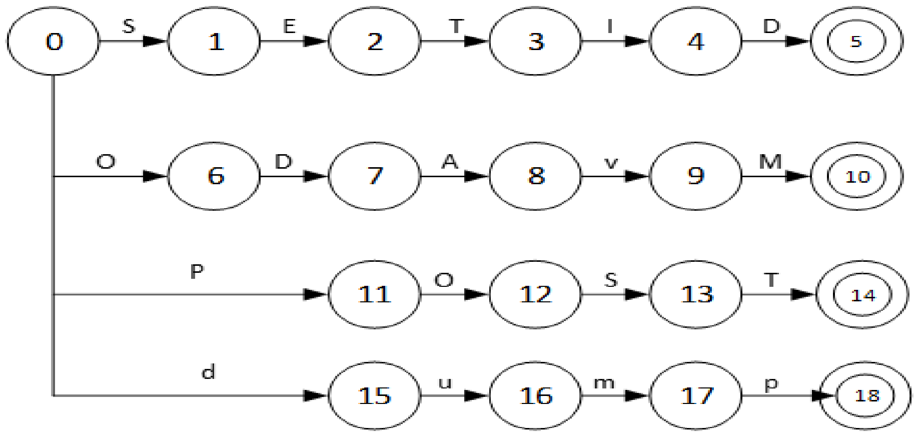


(b)

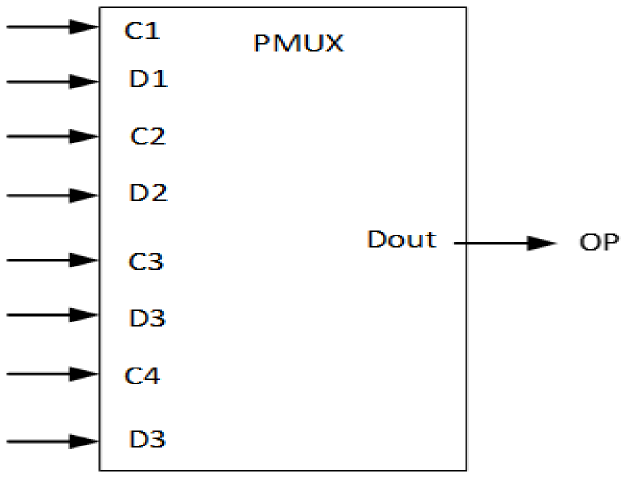
Figure 3.2 Matching Process of Aho-Corasick

3.2.3. AC-NFA Approach:

AC-NFA trie is achieved after eliminating the failure functions from AC-trie. Figure 3.3 shows that by eradicating failure functions AC-NFA trie is formed from AC-trie. After obtaining AC-NFA, all transition states will be matched simultaneously. Only goto functions of original AC-tries remains in NFA-trie. Figure 3.3(b) is utilized to select output from multiplexer circuit.



(a)



(b)

Figure 3.3 AC-NFA trie and Output selection Multiplexer

3.3. Snort Overview:

The essential assembly is exemplified in Figure 3-1. At the point if a data packet reached at the destination, Snort heeds and catches it. Then analysis of packet is performed and directed to the suitable pre-processor for further examination, for example, the "http_decode" dependable of regularising HTTP network stream. The minfrag pre-processor is also case of pre-processor and it manages smaller than usual (minor) sections. Any little piece found on the system is then sent to the minifrag pre-processor for more examination.

The pre-processors are likewise alluded to as modules. There are as of now three kinds of modules in Snort which are pre-processors modules, identification modules and output modules. When the pre-processors work completed, the packet are delivered to the identification portion that will make Snort either to give an alert, or withdraw the packet of data if IPS is used.

Figure 3.2 shows the basic architecture of snort. Packet sniffer modules sniffs the incoming and outgoing packets and sends it to packet decoder module, Then packet decoder module extracts the header and sends packet to pre-processor module. Finally Detection engine gives the result depending on the matching of incoming data stream with the available keywords in the memory. The decision is on the basis of whether incoming packet will be malicious or not. In this work snort rule sets of IP and HTTP and TCP protocol is identified, analysed and tested in real-time. It is noted that clock frequency used while matching a rule-set is inversely proportional to the time required for pattern matching, also throughput depends on the clock cycles consumed in pattern matching.

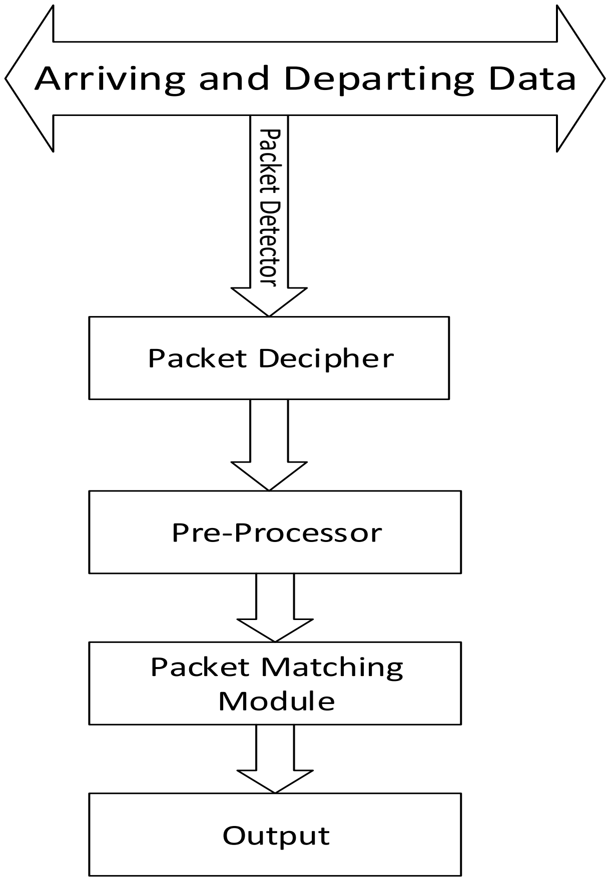


Figure 3.4 Basic Architecture of Snort

3.4. FPGA Prototyping:

After generating the RTL code, then implemented proposed architecture on FPGA. For FPGA prototyping, the system on chip (SoC) of ZYNQ 7000 series used, as shown in Figure 1. Relevant to our requirements, the supported features of ZYNQ 7000 series are as follows:

• Available board:	ZC702
• FPGA device:	Artix–7 (XC7Z020)

- NIC: 88E1116R (1–Gbps data rate)

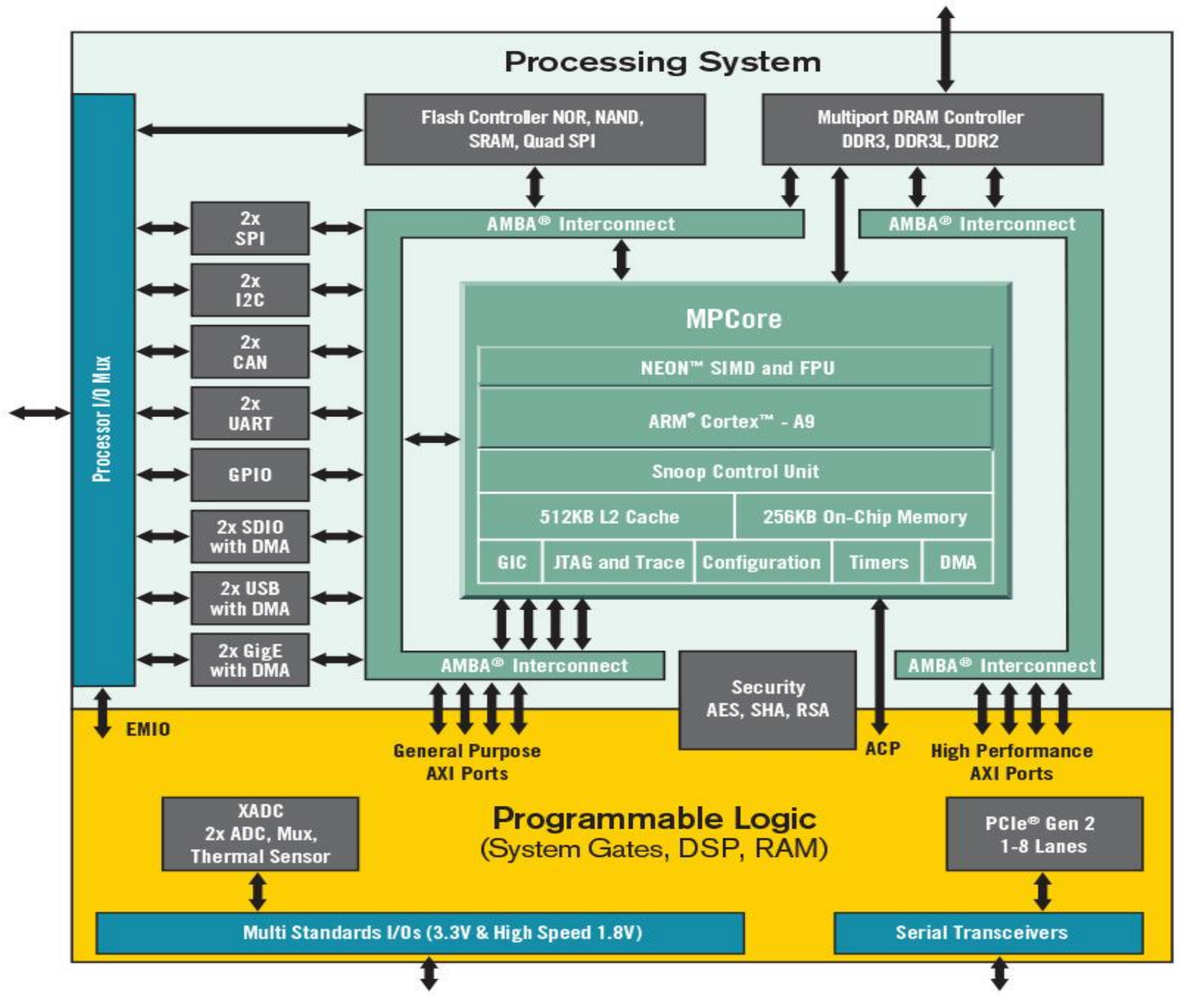


Figure 3.5 Overview of ZYNQ 7000 (SoC)

3.4.1. Details

The previously mentioned gadget family offers the flexibility and adaptability of a FPGA, while giving execution, power, and comfort consistently associated with ASIC and ASSPs. The extent of contraptions in the Zynq-7000 family allows designers to point cost-sensitive similarly as first class applications from a lone stage using industry-standard mechanical assemblies. While every device in the Zynq-7000 family contains a comparative PS, the PL and I/O resources vacillate between the devices. Along these lines, the 7000 and 7000S SoCs can help a wide extent of employments including:

• Automotive driver help, driver information, and infotainment

• Transmission camera

• Industrialized motor controller, present day frameworks organization, and machine vision

• IP and efficient camera

• LTE radio and baseband

• Medical diagnostics and imaging

• Multifunction printers

• Video and night vision equipment

The Zynq-7000 building enables execution of custom method of reasoning in the PL and custom programming in the PS. It thinks about the affirmation of exceptional and isolated system limits. The consolidation of the PS with the PL licenses levels of execution that two-chip courses of action (e.g., an ASSP with a FPGA) can't facilitate in view of their obliged I/O information move limit, inertness, besides, power spending plans.

Xilinx offers a colossal number of fragile IP for the Zynq-7000 family. Free and Linux contraption drivers are available for the peripherals in the PS and the PL. The Vivado® Design Suite progression condition enables a fast thing headway for programming, gear, and structures engineers. Appointment of the ARM-based PS also brings an extensive extent of untouchable gadgets and IP providers in blend in with Xilinx's present PL natural framework. The thought of an application processor enables raised level working structure support, e.g., Linux. Other standard working systems used with the Cortex-A9 processor are in like manner open for the Zynq-7000 family.

Chapter 4

Evaluation

CHAPTER 4. EVALUATION

This chapter first executes the recommended design on ISE design suit tools to assess the possibility of utilizing of resources and evaluate attainable throughput. Our results are then matched with the architectures presented in literature review.

4.1. System Setup

The Patterns for assessment are taken out from Snort (An open source intrusion and detection system). These rules contains keywords of IP, HTTP and TCP packets. Then the executions of snort rules is carried out with the help of goto, failure and output matrices. There are 75,306 keywords of TCP packet, 6059 of HTTP packet and 981 rules are present in IP data packet. In order to get sorting of these rules and to extract content of each rule an auto-generator code is implemented.

To get the goto, failure and output matrices a comprehensive implementation is performed, for example in case of IP snort keywords, the average length of keywords is 13. After successful execution of FPGA device the evaluation of utilization of LUTs, FFs is performed. Also calculate the total on chip power, signal power and static device power consumed. Accordingly after that calculate and analyze achieved throughput.

Implementation of snort rules in performed and then results are analyzed and compared with previous work explained in literature review.

4.2. Performance Metric:

Eq. (1), will be used to calculate the time required for one pattern matching. Throughput of the proposed architecture will be calculated using Eq. (2). Finally, performance of the proposed architecture will be evaluated using Eq. (3). For Example for a pattern to be matches 10 clock cycles are required at clock frequency of 100MHz.The throughput of the system will be 10Gbps. Which means that for one pattern matching the time required will be 0.1us.

$$time\ for\ one\ pattern\ matching = \frac{number\ of\ clock\ cycles}{operational\ clock\ frequency} \qquad \text{Eq. (1)}$$

$$throughput = \frac{1}{time\ for\ one\ pattern\ matching} \qquad \text{Eq. (2)}$$

$$\frac{throughput}{area} = \frac{\frac{1}{time\ for\ one\ pattern\ matching}}{FPGA\ Slices} \qquad \text{Eq. (3)}$$

4.3. Evaluation and Comparison

In this section first of all IP's keywords is performed on Aho-Corasick algorithm and hardware parameters will be summarized in table 3.

In table 3 Hardware parameters i.e. LUTs, FFs, Clock Frequency is presented. Also on the basis of number of clock cycles utilized in pattern matching throughout is evaluated. For 981 keywords pattern matching, the maximum length of keyword is 24. So by using equation (1). Time for one pattern matching will be 6.8us and throughput will be 1.8Gbps.To further optimize throughput the IP keywords are divided into chunks and then perform the pattern matching. Results are shown in Table.

Rule-sets	Max. Clock Cycles utilized/pattern	Hardware related parameters					
		Time for one pattern matching	Throughput (Gbps)	FPGA Area		Freq. (MHz)
				LUTs	FFs	
For 981 keywords	1500	6.8us	1.8	2110	451	220
For First 490 Keywords	1500	5.2us	2.3	1530	386	284
For remaining 490 Keywords	1500	5.3us	2.3	1250	315	279

Table 3 Evaluation with 981 IP Keywords

In table 4 Http snort keywords RTL implementation is performed and execute code with the help of goto, failure and output matrices. There are 6059 snort rules of http also the maximum keyword length is 78. To further optimize the throughput the http rules are equally divided and the perform the pattern matching procedure. Finally the achieved parameters are calculated and presented in Table 4.It has been seen that throughput achieved while processing 6059 keywords in one pass is 0.96Gbps while after equally dividing rules throughput will increase to 1.23Gbps and for next 3030 rules it will be 1.28Gbps.

In table 5 we summarized parameters of TCP snort rules. It can be seen that for complete 75,306 rules the achieved throughout is 0.43Gbps.

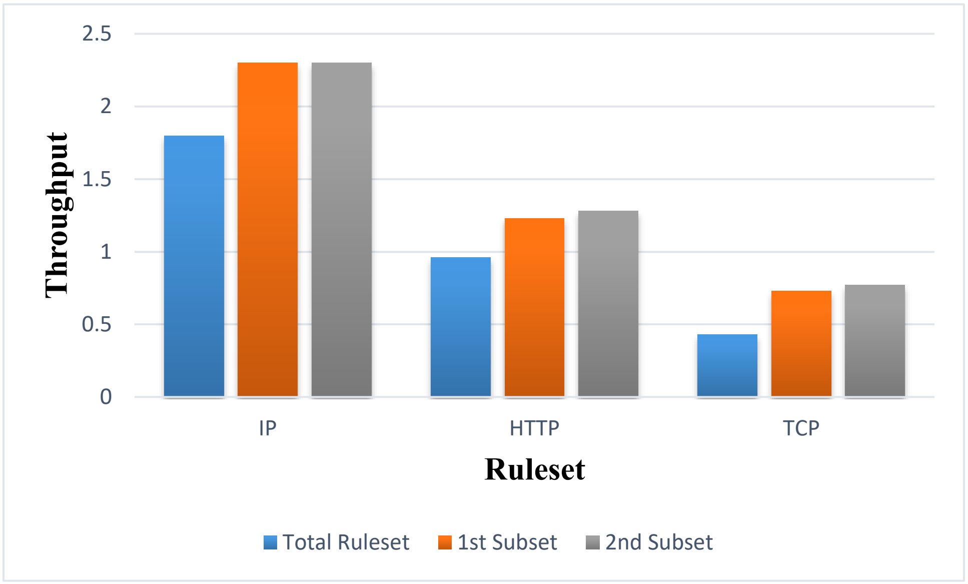
Rule-sets	Max. Clock Cycles utilized/pattern	Hardware related parameters				
		Time for one pattern matching	Throughput (Gbps)	FPGA Area		Freq. (MHz)
				LUTs	FFs	
For 6059 keywords	1500	12.8us	0.96	12,820	2,521	117
For First 3030 Keywords	1500	9.9us	1.23	8,021	1,910	151
For remaining Keywords	1500	9.6us	1.28	5,321	1,294	156

Table 4 Evaluation with 6059 HTTP Keywords

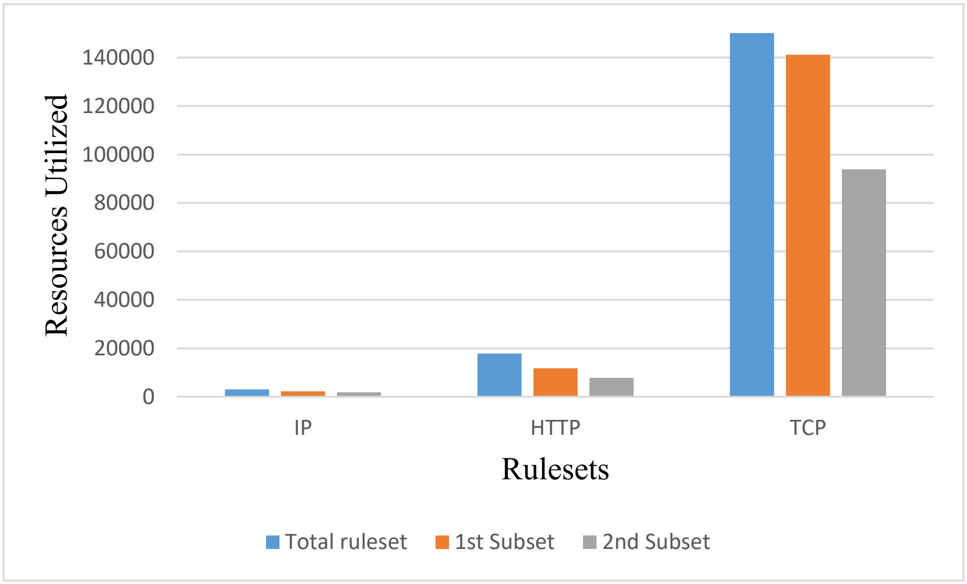
Rule-sets	Max. Clock Cycles utilized/pattern	Hardware related parameters				
		Time for one pattern matching	Throughput (Gbps)	FPGA Area		Freq. (MHz)
				LUTs	FFs	
For 75306 keywords	1500	28.3us	0.43	153,840	30,252	53
For First 37653 Keywords	1500	16.8us	0.73	96,252	22,920	89
For remaining Keywords	1500	15.9us	0.77	63,852	15,528	94

Table 5 Evaluation with 75306 TCP Keywords

4.3.1. Graph of Throughput vs Number of Rule-set:



4.3.2. Graph of Resources Utilization vs Number of Rule-sets:



Chapter 5

Conclusions and Future Work

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

This section will begin by to the point the results that have been achieved and the make suggestions as needs be for future work at a more significant level.

5.1. Contributions

A lot of work has been done in propelling the viability of security frameworks. Before parallel IDS were talked about, assaults that are part into various stages have consistently been exceptionally hard to investigate, recognize and alleviate. With the move toward equal IDS, multistage assaults would be considerably increasingly hard to distinguish. The trouble lives in the way that there is no connection between the various centres that play out the examination. The IDS will surely improve in wording of speed for example the quantity of payload prepared every second and at the equivalent time, however when assaults are part into various stream, most current frameworks don't associate streams. In this exploration, the creator adds that measurement to the current framework. Notwithstanding, this would make the IDS be upgraded and that is the thing that this exploration is about.

An advanced auto-trie generator is designed which takes rulesets of data packet and irrespective of type of packet it generates AC-NFA trie which helps us to get goto, failure and output matrics. Also in this thesis transition state reduction of snort IP, HTTP and TCP packet rules is achieved which automatically reduces number of resources.

5.2. Future Works

The idea of this work in itself is a test which has produced numerous other challenges which can fill in as full research assignments. These assignments could be:

- Plan and build up an assault tree equipped for a full incorporation for a multicore atmosphere.
- Plan and execution of a pipelined design, the default packet utilized for packet capturing.
- Design and implementation of mutli-character pattern matching using Aho-Corasick algorithm up to 24 characters in one clock cycles.
- To design architecture of ICMP, UDP snort rules and to develop advance solution of IDS.

REFERENCES

[1] Jiajia Yang, Lei Jiang, Xu Bai and Qiong Dai, “High Performance Regular Expression Matching on FPGA.ICSR pp.541-553,2018

[2] Kubilay Atasu,” Feature-rich Regular Expression Matching Accelerator for Text Analytics” Journal 2016.

[3] Tran Ngoc Thinh, Tran Trung Hieu, Hiroshi Ishii and Shigenori Tomiyama, “Memory efficient signature matching for ClamAV on FPGA”. Conference 2014

[4] Junsik Kim and Jaehyun Park, “FPGA-based network intrusion detection for IEC 61850-based industrial network”. Journal February 2018

[5] Derek Pao, Nga Lam Or, and Ray C.C Cheung, “ A memory-based NFA regular expression match engine for signature -based intrusion detection” Computer Communications Journal March 2013.

[6] P. M. K. Tharaka, D. M. D. Wijerathne, N. Perera, D. Vishwajith and A. Pasqual, "Runtime rule-reconfigurable high throughput NIPS on FPGA," 2017 International Conference on Field Programmable Technology (ICFPT), Melbourne, VIC, 2017, pp. 251-254.

[7] D. Pao and X. Wang, "Multi-stride string searching for high-speed content inspection," Comput. J., vol. 55, no. 10, pp. 1216–1231, 2012.

[8] D. Pao, W. Lin, B. Liu, "A memory-efficient pipelined implementation of the aho-corasick string-matching algorithm", ACM Trans. Archit. Code Optim., vol. 7, no. 2, pp. 1-27, 2010.

[9] X. Wang and D. Pao, "Memory-Based Architecture for Multicharacter Aho–Corasick String Matching," in IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 26, no. 1, pp. 143-154, Jan. 2018.

[10] Kim H.J. (2015) A Failureless Pipelined Aho-Corasick Algorithm for FPGA-based Parallel String-Matching Engine. In: Kim K. (eds) Information Science and Applications. Lecture Notes in Electrical Engineering, vol 339. Springer, Berlin, Heidelberg

[11] C.-C. Chen, S.-D. Wang, "An Efficient Multicharacter Transition String-matching Engine Based on the Aho-corasick Algorithm", ACM Transactions on Architecture and Code Optimization, 2013.

[12] C.-C. Chen and S.-D. Wang, “A hybrid multiple-character transition finite-automaton for string matching engine,” Microprocess. Microsyst., vol. 39, no. 2, pp. 122–134, Mar. 2015.

[13] João Silva , Valery Sklyarov and Iouliia Skliarova,”Comparison of On-chip Communications in Zynq-7000 All Programmable Systems-on-Chip” pp:31-34 March 2015