A Real-Time Firewall Policy Rule Set Anomaly-Free Mechanism

Mohamed M.A. Elgazzar[1] , Mohamed E. Elhamahmy[2] , Abdel-Hamid M. Emara[3,4]

[1]Asso.Prof, ,Higher Institute of Computer Science and Information Systems, Fifth Settlement, New Cairo, Egypt.
[2]Dr. Information Technology,, Cairo, Egypt.
[3]Department of Computers and Systems Engineering ,, Faculty of Engineering, Al-Azhar University, Cairo, Egypt.
[4]Department of Computer Science, Taibah University, Medinah, Kingdom of Saudi Arabia.

Abstract *— a lot of work has been done on detecting firewall policy anomalies. There are tools have been proposed to help dependent on these studies, for example, Policy Advisor Tool. However, it dependent on the insertion of policy rules manually into the tool. There is a real need for a tool that acquire the firewall policy rules in real-time. There are also tools produced by firewall vendors developed for firewall systems to work on the management of their devices only and therefore does not work with the devices produced by other companies. Due to the growing network and its dependence on many of the firewall devices of different types and brands, it has become difficult to manage policies on various Firewall of different types and vendors and rely on many different tools as well. In this paper we proposed a method for investigating the firewall rule-set anomalies and suggest how to fix them. We also have built a practical tool that can obtain a copy of the policy rule-set in real time and dealing with several devices of different brands using the same tool. The proposed tool is considered as a way to help the network administrator and not an alternative him.*

Keywords - *Firewall; rule; policy; API; REST API; Anomaly detection and correction*

I. Introduction

Regularly, firewalls are deployed at the boundary of the network subsequently protecting the private network. In any case, there has been an emphasis on distributed firewalls. The essential thought of appropriated firewalls is to make each host in the system into a firewall that channels traffic to and from itself [1]. Recently, firewalls have developed from traditional packet filters to application entryways. The security policy implemented in these firewalls may consist of hundreds of rules and objects. The firewall rules included groups of servers, user machines and sub-networks. The firewalls rules and policies need to be frequently changed and modified because of the dynamic needs of its organization business [2]. In spite of the way that the progressions in firewall development is basic to secure private perimeters, the multifaceted nature and anomalies in firewall policy rules may compel the sufficiency of firewall security. These referenced challenges can be settled by an effective firewall inconsistency the practical tool which will naturally obtain the firewall rule set at that point recognize and potentially resolve the anomalies either automatically or manually.

In this paper, we propose an approach for obtaining a firewall policy rules from numerous continuous firewalls. At that point, we implemented a functional tool dependent on the proposed approach named firewall policy manager (FPM). The proposed FPM tool is utilized for ongoing obtaining a firewall policy rule set and identifying its anomalies. The redress activity for managing the distinguished anomalies is counseled by the FPM tool too. The expansion of another rule to the current firewall rule set may brought about new anomalies. So the algorithm for adding new rules to the current firewall rule set maintaining a strategic distance from anomalies is proposed as well.

A. Problem statement

Managing the firewall policy rule set is considered as one of the fundamental research subject for a years ago. A large portion of the current examines considered the firewall policy rule set is gotten and they concentrated on recognizing the standard set anomalies and suggested the amendment activity. However, the previously mentioned systems are unfit to get firewall policy rules from real-time firewalls with powerfully changing rule sets.

B. Scope of work

In this paper we are given two research issues. The first is proposing an approach for firewall policy rule-set acquisition in the real-time. Then, building a practical tool based on the proposed approach for real-time firewall policy rule set acquisition and make it displayed in graphical user interface (GUI). Second, is proposingand implementing algorithms,within the FPM tool, to detect and correct anomalies in the obtained firewall policy rules.

II.Related works

There are many studies focused on firewall management. Despite the fact that the advancements in firewall innovation is imperative to verify private networks, the intricacy and inconsistencies in firewall policy rules may restrict the viability of firewall security. In [2] authors formally characterized kinds of firewall anomalies as between firewall and intra-firewall anomalies. Intra-firewall anomalies are the inconsistencies present in the same firewall where the same packet may coordinate more than one filtering firewall policy rule. Inter-firewall anomalies arises between firewall anomalies emerge when singular firewalls in the same network perform diverse activities on a similar packet. Regularly, for expansive endeavor arranges, the access control lists (ACL's) can proceed to be a few several lines extensive. Accordingly, the trouble of the board of ACL's (for example including new rule or changing existing one) fundamentally increments and an anomaly may be presented particularly in circulated networks. Existing approach investigation tools, for example, Firewall Policy Advisor [2] and FIREMAN [3], with the objective of recognizing policy rules anomalies have been presented. Firewall Policy Advisor just has the ability of recognizing pairwise anomalies in firewall rules. FIREMAN can identify anomalies among numerous rules by analyzing the relationships between one rule and the collections of packet spaces derived from all preceding rules. In any case, FIREMAN likewise has constraints in identifying anomalies. For every firewall rule, FIREMAN just inspects every first rule however disregards every consequent rules when performing anomaly investigation. What's more, every examination result fromFIREMAN can just demonstrate that there is a misconfiguration between one rule and its previous rules, be that as it may, can't precisely show all rules associated with an anomaly [3].However, the method of acquiring policy rules in real-time was not presented.

III.The firewall policy management

A. Firewall policy rules

A firewall is a security system that expected to protect private network edges. It controls the approaching and active network traffic by sifting the information bundles as per requested principles to decide if they ought to be permitted or denied. A rule is described by attributes and an action to perform when a packet coordinates the attributes criteria. The characteristics of a standard packet involve the sequence rule number, protocol, source Internet protocol (IP), source port, destination IP and destination port. As such a complete rule may be portrayed by the organized tuple #order, protocol, source IP, source port, destination IP, destination port, action#. Each rule can be described when in doubt of attributes, which can be addressed and researched like sets. Table (I) displays example of a firewall rule set.

Table (I): An example of a firewall policy rule-set

Rule Order	Protocol	Src. IP	Src. Port	Dst. IP	Dst. Port	Action
Rule 1	UDP	10.5.*.*	53	60.117.38.5	80	Deny
Rule 2	UDP	10.5.15.*	25	172.33.1.2	53	Allow
Rule 3	TCP	10.1.1.2-10.1.1.20	25	60.117.38.4	25	Allow
Rule 4	TCP	10.1.1.15-10.1.1.30	25	60.117.38.4	25	Allow
Rule 6	UDP	10.5.15.15	53	172.33.1.2	53	Deny
Rule 7	TCP	10.5.15.10	80	172.33.*.*	80	Allow
Rule 8	UDP	10.5.25.*	53	60.117.38.5	53	Allow
Rule 10	TCP	10.5.15.10	8080	172.33.38.5	8080	Deny
Rule 15	TCP	10.5.15.*	80	172.33.38.5	80	Deny
Rule 18	UDP	10.5.15.*	53	172.33.38.5	53	Allow

As shown in Table (I), rule 1 has a source address range of IP of value (10.5.*.*). The source address range of the rule 2 is (10.5.15.*), which is considered as a subset of that of the rule 1. Thus the action field does not turn out to be perhaps the most imperative factor when contemplating the connection between two rules.

B. Relations between two firewall policy rules

As the estimations of substitute properties of firewall policy rules can be addressed as sets, we can see a standard characteristics as it is made out of sets. In Figure (1), the conceivable relations between two sets might be disjoint, when their attribute values are distinct, or there are a partially overlapping between them, or one attribute is considered as a subset of another, or they are identical. In Figure (1-c), the set B is a subset of set A. It implies that the source address of one rule might be a scope of Internet protocol addresses IP (s) which are a subset of that identified with the compared rule source (IP), and the otherway around.

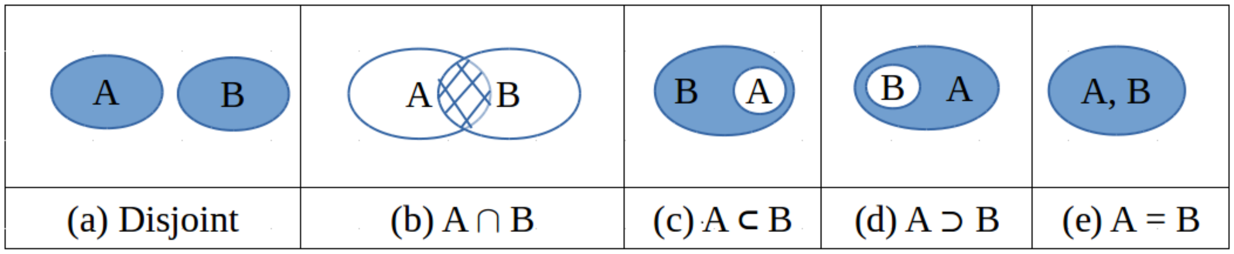


Figure (1): The Venn-diagram of two firewall policy rule attributes as sets

Disjoint (D): In science, two sets are said to be disjoint sets in the event that they share no component for all intents and purpose. Similarly, two disjoint rules can't avoid being rules whose intermingling is the empty set. By looking at two rules and in the event that there is no closeness in all fields between them, at that point they are disjoint.

Exactly Matching (EM): Two rules are exactly matched if each attribute of the rules match exactly.

Inclusively Matching (IM): A rule is inclusively matched of another rule if there exists at least one attribute of one rule value is a subset of the other's value and for the remainder of the attributes, the both are identical.

Correlation: Two rules are correlated, if they are not disjoint, but they have different filtering actions and there are same packets that are filtered by the both rules.

Intersection: Two rules are intersected, if they are not fully disjoint, but they are partially disjoint as there are a common shared attributes as well as there are disjoint space.

Firewall policy rule anomalies

The contradiction between rules is called anomaly. It means that more than one firewall policy rule can filter the same packet. The firewall policy rule anomaly may be classified as main four classes shadowing, redundancy, generalization, and correlation anomaly.

Shadowing Anomaly: A rule Rx is shadowed by another rule Ry if Ry precedes Rx in the policy, and Ry can match all the packets matched by Rx. The effect is that Rx is never activated. As shown in Table (1), let Ry is regarded by (Rule 1), and Rx is regarded by (Rule 8). The range of IP(s) of the Source_IP of Ry is (10.5.*.*.) which included the range of IP(s) of Rule 8 (10.5.25.*) and the reminder attributes are identical (the same protocol, source port, destination IP, and destination port). However the Action of (Rule 1) is (Deny). It means that (Rule 1) is denying all packets from source IP (10.5.*.*) included (10.5.25.*). So that, the (Rule 8) is shadowed by (Rule 1).

Redundancy Anomaly: A redundant rule Rx performs the same action on the same packets as another rule Ry such that if Rx is removed the security policy will not be affected. Both of (Rule 10) and (Rule 18) in Table (1) are similar as they match the same packets and perform the same action. Therefore, their attributes are EM. If one of these rules are removed from the rule set, the reminder rule will take place and the security policy should not be affected.

Generalization Anomaly: A rule is a generalization of another rule if it matches a specific rule that precedes it. Rule Ry is a generalization of rule Rx if Ry comes after Rx in the order, and Ry is a superset match of Rx , and the actions of an Ry d Rx are different. Both of (Rule 2) and (Rule 6) in Table (1) have the same protocol, source port, destination IP, destination port. However, the source address of the (Rule 2) is superset of that of (Rule6). As well as the action of both rules is not the same. So that, if order of the rules is reversed in Table (2), then action will change and the superset rule (Rule 2) will shadow the other rule (Rule 6). Rule 2 is superset rule relative to (Rule 6) and it is known as the General rule.

Correlation Anomaly: Two rules Rx and Ry are correlated if they have different filtering actions and the Rx matches some packets that match Ry and the Ry matches some packets Rx that matches. In Table (1), (Rule 7) is allowing all packets come from the source address (10.5.15.10) to destination at port 80, while (Rule 15) is contracted to this rule which wants to deny any packet comes from the range of IP(s) (10.5.15.*) at the same destination port 80.

IV. The proposed approach

In order to access a firewall for obtaining its configuration, Telnet/SSH (secure shell) may be used. However, many firewall devices do not support using Telnet/SSH client. This is because the use of client can be used to effectively bypass the firewall – that is, a user can SSH into the firewall, then SSH out from the firewall and then represents a security breach under most corporate security policies. The web services are more preferred for accessing firewalls [7]. Many firewall vendors such as Cisco, Palo Alto, Juniper and FortiGate are using REST Representational State Transfer) architecture on their recent firewalls.

In this paper, a layered framework based on REST architecture is proposed for obtaining the firewall rule-set in real-time. Alayered framework is a framework involved layers, with each layer having a particular usefulness and obligation. Each layer has its very own obligations, with the models containing how the information ought to be framed, the controller concentrating on the approaching activities and the view concentrating on the yield. Each layer is independent yet additionally collaborates with the other. Most of the tools mentioned in the previous work involved some versions of command line interface (CLI) scripting since nearly all firewall features and functions are available in this manner [7]. Perl and Expect scripts are the common scripting languages in use for managing traditional firewalls [8]. Historically the traditional firewalls have been a closed platform in the sense "Syslog" has been the only choice for event management. Recently, fully functional virtualized firewalls were introduced [9]. Combined with this virtualized offering and the increase in Software Defined Networking (SDN), and "Network Function Virtualization (NFV)", the next logical step was to develop an application programming interface (API) to manage firewalls [10]. This gives administrators a very flexible platform where common elements could be created and re-purposed to orchestrate and automate deployments [10]. So that we choose to acquire the firewall policy rule set by using one of the web services API.

The web services may be either Simple Object Access Protocol (SOAP) or Representational State Transfer (REST). SOAP is an Extension Markup Language (XML) based protocol for accessing windows services (WS). It is a platform independent and language independent as well. It also allows interacting with other programming language applications. REST is an architectural style, not a protocol, as a web service, it is faster than SOAP because there is no strict specification like SOAP [11]. It consumes less bandwidth and resources as well. REST WS are both language and platform independent. They can be written in any programming language and executed in any platform. In comparing with SOAP, REST can use SOAP WS as the implementation. However, SOAP cannot use REST because it is a protocol. One more advantage of using REST API architecture, it permits different data formats such as Plain Text, Hyper Text Markup Language (HTML), XML, and Java Script Object Notation (JSON). Therefore, REST is more preferred than SOAP in implementation of the proposed tool because it is easier, faster and an architectural style.

In REST API design, a similar standard remains constant, with various layers cooperating to construct a chain of command that makes a progressively adaptable and secluded application. Based on the properties and parameters of a firewall design tool in [12] a multi-layers based model was proposed to achieve these determinants. The trend towards a multi-layered model is also consistent with REST API architecture. To connect with the firewall device using WS it should ask for the authentication first. The handling is maintained through secure web protocol (HTTPS). Then the configuration files should be issued by the authenticated user. The firewall then responds by sending the policy rules as a JSON file. Then the obtained JSON files should be prepared for presenting through a graphical user interface (GUI) standing for visualization and usability.

A. The proposed model

As shown in Figure (2), the proposed model for managing the firewall policy is composed of a four-layered conceptual model in addition to the anomaly detection and correction modules that are applied on the obtained rule set.The proposed model should stand for usability as well as visualization. Data acquisition is the preliminary step in visualization [12]. So, the data acquisition layer is that layer to interact with the firewall asking for its rule set. The firewall policy rules are the data of interest in this scope. The policy rules are obtained from the firewall by the administrator. He should use suitable credentials for firewall authentication access. Then the suitable REST API command should be issued for acquiring the policy rules. The second layer is the filtering. Data filtering is used to extract the portion of data of interest to be visualized. The third layer is the parsing layer. Data parsing is used to transform selected objects extracted from the returned files into readable data. The presentation layer is aimed to represent the obtained firewall policy rules in GUI. The presentation step is used to bring realism in data.Then the anomaly detection and correction algorithms are applied to the obtained rule set. Based on this, by examining the rule set of a real-time firewall policy, we propose in this paper that the outputs of the examination of the firewall rule set be four lists. The list of general rules of high risk, redundant rules list that contains the rules have been deleted, the anomaly rules list, and finally the optimized rules list as shown in Figure (2). The main idea behind this proposed model is to deal directly with a firewall in real-time.

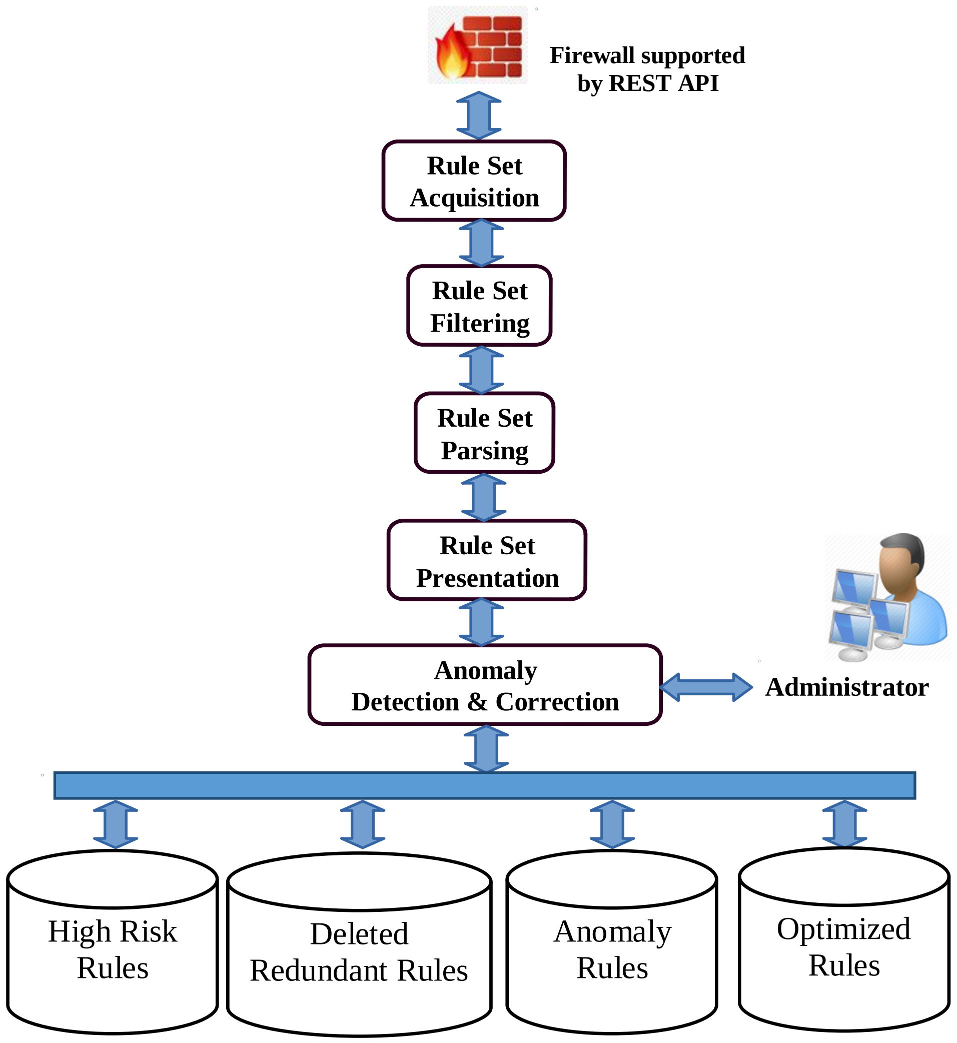


Figure (2): The proposed firewall management approach

Where there is a challenge to deal with dynamic policy rules for real-time firewall devices [3]. In addition to examining the firewall policy rule set and classification into four categories. The proposed model is managed by the network administrator just to assist him in managing the multiple firewalls inside the network. The model is planned to manage the firewall policy rule set safely. As it just retrieve a copy of the firewall rule set and investigate its anomaly then make the outcome lists available for the administrator provision without physically changing the rule set on the firewall device.

B. FPM tool based on the proposed approach

An implementation of a practical tool based on the proposed model is demonstrated. It is a tool to be used by the network administrator and it is named FPM. The (FPM) tool aimed to assist the administrator for the central management of multi-firewalls (may be of different vendors) in real-time. It stands for design principals of usability and visualization. It is safe as it just retrieves a copy of the firewall rule set without changing on the resources. It uses WS instead of CLI. The firewall policy rule set acquisition process is maintained by the administrator. The administrator first uses the appropriate credentials to authenticate the use of the determined firewall device. Then using the FPM tool to issue API scripts over the HTTP protocol to acquire the firewall policy. It returned with its policy rule set on a form of JSON file. JSON is an open-standard file format that uses human-readable text to transmit data objects consisting of attribute-value pairs and array data types. The obtained file is processed and parsed in order to get a visual firewall rule set. Then the returned rule set should be investigated. To evaluate the proposed tool named FPM, it is required to have firewall devices that supported the REST API architectures. The virtualization technology and cloud environment may be used for testing the firewall configuration [13]. The virtual machine (VM) of firewalls from different vendors such as "Cisco Adaptive Security Appliance (ASA) series v9.4" is added to the proposed tool FPM. In a word, Cisco ASA is a security device that joins firewall, antivirus, interruption aversion, and virtual private system (VPN). Another firewall device is added to FPM tool is Palo Alto device with its operating system (PAN-OS) version 8.0.0. Skillet operating system (PAN-OS) is the product that runs all Palo Alto Networks cutting edge firewalls. The Palo Alto firewall models of rendition 8.0.0 or more upheld REST API. One more firewall device is added to the FPM tool is Juniper Network Operating System (JUNOS) Series Services Gateways (SRX) version SRX-15.1X49-D154 is added to the FPM too. JUNOS is the FreeBSD-based working framework utilized in Juniper Networks equipment switches and firewalls. SRX (Series Service Gateways) depend on JUNOS demonstrated working framework which conveys security and propelled assurance administrations. The last firewall added to the FPM tool in this paper is "FortiGate v4.5.1". These virtual machines represented the most known firewall devices are used by FPM tool for testing purpose. Each one of them is studied well, to learn how to add the device to the proposed FPM tool and how to send and receive the required data based on the proposed approach. It has been found that each of them has a different form in the handling of policy objects. For example, all the security objects of the Juniper firewall are retrieved once. However, the other three devices (Cisco, Palo Alto and ForiGate) have multiple objects that returned by multiple JSON file queries.

After some trials, we have decided to upgrade all applications to ".NET 4.5". This upgrade selects the version of the Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocol to use for new connections; existing connections aren't changed. An example REST call that authenticate a firewall device that supports REST API architecture at IP address <firewall-IP> is:

RestClient (https://<firewall-IP> + <Logincheck>);

An example REST call that provides data to the proposed tool (FPM) at IP address <firewall-IP>is presented in Figure (3):

```
var client = new RestClient(https://+ IP+ "/logincheck");

Client.Authenticator = new SimpleAuthenticator
("username", User, "secretkey", PW);

var request = new RestRequest () { Method = Method.Post } ;

var v = client.Post (request) ;
```

Figure (3): The REST calls that provide data to the FPM tool.

An example REST call that provides data of a firewall device by its dedicated IP-address <firewall-IP> in JSON to the proposed tool (FPM).

```
"http_method": "GET",
"results": [
{           "policyid":1,
            "q_orrigin_key":"1",
            "name":"Ahmed",
"uuid":"bdd275c8_6218_51e7_e5d3_3c458e379603",
"srcaddr": [
{           "name":"192.111.1.4",
            "q_origin_key":"192.111.1.4"
}
],
"action":"accept",
}].
```

Figure (4): The REST calls that provides data of firewall device by its IP-address.

UUID (Universal Unique Identifier) is a 128-bit number used to uniquely identify some objects. The objects are units of code that are eventually derived from the process or entity on the network. Depending on the specific mechanisms used, a UUID is either guaranteed to be different or is, at least, extremely likely to be different from any other UUID generated until certain predefined value.

The output from FPM REST API is in JSON can be used to parse the JSON string returned:

RestClient (https://<firewall-IP>:8080/~path~/json)

For each of the above examples, <firewall-IP> should be replaced with the IP address or hostname of the targeted firewall. Each one of the four firewall devices mentioned above is studied well, to learn how to add the device to the proposed FPM tool and how to send and receive the required data using the proper API based on the proposed approach. It has been found that each of them has a different form in the handling of policy objects. For example, all the security objects of the Juniper firewall are retrieved once. However, the other three devices (Cisco, Palo Alto and FortiGate) have multiple objects that returned by multiple JSON file queries.

C. The experimental results

To test the response time of each firewall device, the experiment is run on the available personal computer. The implementation of the proposed work is maintained by a laptop system type x64-based PC. The operating system's name is Microsoft Windows 10 home edition – version: 10.0.17134-build 17134. Processor Intel(r) core(tm) i5-7300hq central processing unit is 2.50 GHz, 2501 MHz, four core(s), four logical processor(s). Installed physical memory (ram) is eight GB. Total physical memory is 7.87 GB. Available physical memory: 1.64 GB. Total virtual memory: 19.4 GB. Available virtual memory: 3.08 GB. Page file space: 11.5 GB. As shown in Table (II), the virtual machine represents FortiGate firewall v5.4.1 has a good time response to deliver its JSON files that contain its configurations. Cisco ASA v9.4 virtual machine consumed more time (about 890ms) to respond with its Network Address Translation (NAT) configuration

Table II: the experimental results

	Palo Alto		Cisco		FortiGate		Juniper	
	Time(ms)	Size(bytes)	Time(ms)	Size(bytes)	Time(ms)	Size(bytes)	Time(ms)	Size(bytes)
GetPolicy	512	696	67	3497	9	1392	1976	34373
GetNat	518	202	890	4011	39	210	1976	34373
GetObject	516	359	338	1791	40	360	1976	34373
GetRouting	507	685	134	3397	41	359	1976	34373
GetObjService	1171	499	105	120	38	1500	1976	34373
GetApplication	-	-	-	-	-	-	1976	34373

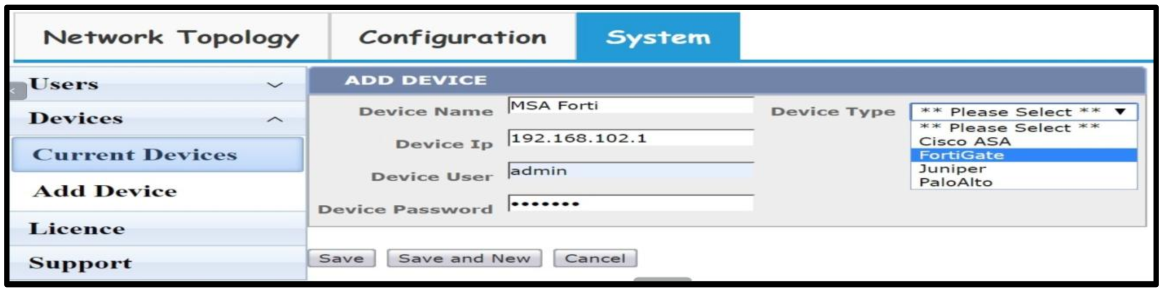
The virtual machine of Palo Alto firewall v8.0.0 has the longest latency in replying with its configuration objects relative to the others. It took about (1171ms) to deliver its service configuration. The virtualmachine of Juniper version SRX 15.1x49 has a different configuration object named (Application) instead of (Service) as found on the rest devices, and it replied with a fixed time about 1976 milliseconds for each configuration file response.

D. The real environment results

Subsequent to adding firewall equipment to the proposed tool and testing on four firewall gadgets

through VM, we went to attempt the proposed tool in a genuine situation.

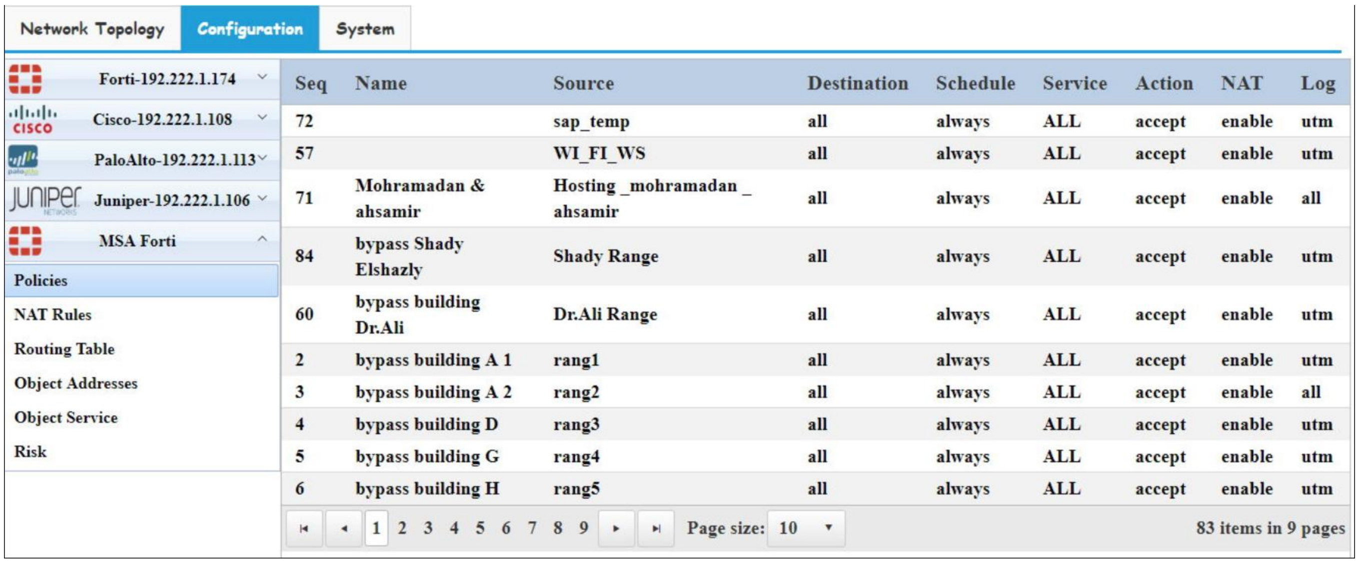
A server farm identified with one Egyptian college was chosen and we acquainted the proposed instrument FPM with the system administrator and requested a firewall gadget to be evaluated. It was FortiGate version 4.5.1 managed security service (MSA). It is named (MSA Forti).



Device Name: MSA Forti
Device Ip: 192.168.102.1
Device User: admin
Device Password: •••••••
Device Type: ** Please Select **
** Please Select **
Cisco ASA
FortiGate
Juniper
PaloAlto

FIGURE (5): ADD NEW FIREWALL DEVICE TO THE FPM TOOL

The required username, password, and IP addresses are controlled by the administrator so as to associate the "MSA Forti" firewall utilizing the FPM tool as shown in Figure (5). It was perceived specifically in light of the fact that a similar kind of firewall was predefined to the FPM instrument previously. After passing the authentication phase, the HTTP request was initiated, and then the firewall gadget reacted with sending back its configurations in type of JSON files.

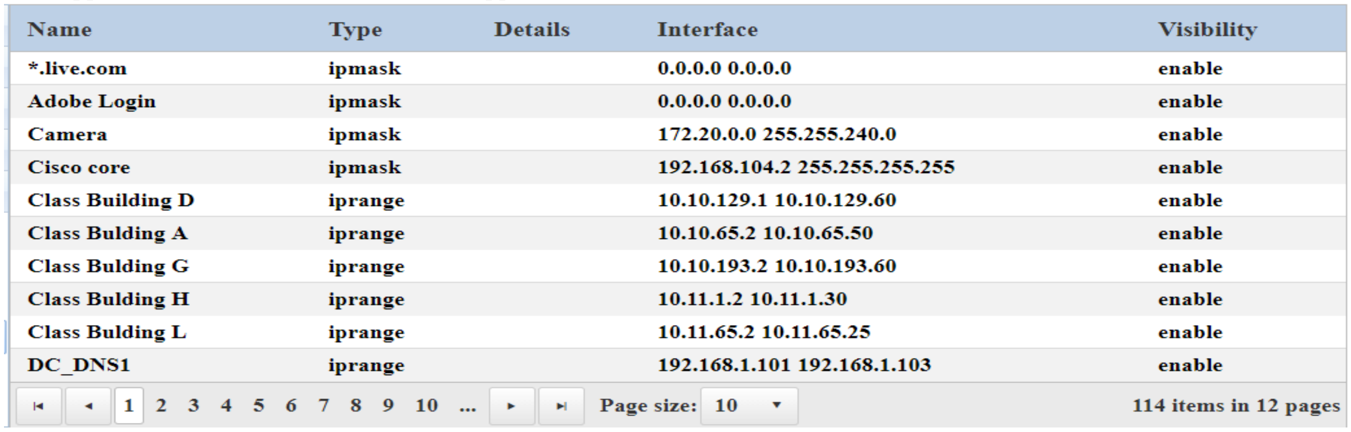


Seq	Name	Source	Destination	Schedule	Service	Action	NAT	Log
72		sap_temp	all	always	ALL	accept	enable	utm
57		WI_FI_WS	all	always	ALL	accept	enable	utm
71	Mohramadan & ahsamir	Hosting _mohramadan _ ahsamir	all	always	ALL	accept	enable	all
84	bypass Shady Elshazly	Shady Range	all	always	ALL	accept	enable	utm
60	bypass building Dr.Ali	Dr.Ali Range	all	always	ALL	accept	enable	utm
2	bypass building A 1	rang1	all	always	ALL	accept	enable	utm
3	bypass building A 2	rang2	all	always	ALL	accept	enable	all
4	bypass building D	rang3	all	always	ALL	accept	enable	utm
5	bypass building G	rang4	all	always	ALL	accept	enable	utm
6	bypass building H	rang5	all	always	ALL	accept	enable	utm

1 2 3 4 5 6 7 8 9 Page size: 10 83 items in 9 pages

Figure(6): The returned real-time Fortigate firewall policy rule-set

The proposed tool (FPM) processed files and extracted the data of interest (DOI). These DOI includes the policy rules, NAT, the routing Table, object addresses and object service. These objectswere processed for viewing through a visual interface and in a way that helps the network administrator review the policy rules as shown in Figure (6).As appeared in Figure (6), in the left pane of the screen are the firewall devices that previously included. By choosing the firewall device name, you will find that the arrangement objects have been appeared at the base of the device. As appeared on the Figure (6), there were 83 rules that can be perused in nine pages. The displayed page (1) contains the policy rules which displayed in a tabular form. The first row on the Table includes the column names. The FortiGate machine is named likewise Unified Threat Management (UTM). Empowering log security occasions will just show up traffic log coordinate UTM profile characterized. Notwithstanding, (ALL) or (log all sessions) will incorporate traffic log both match and non-coordinate UTM profile characterized.

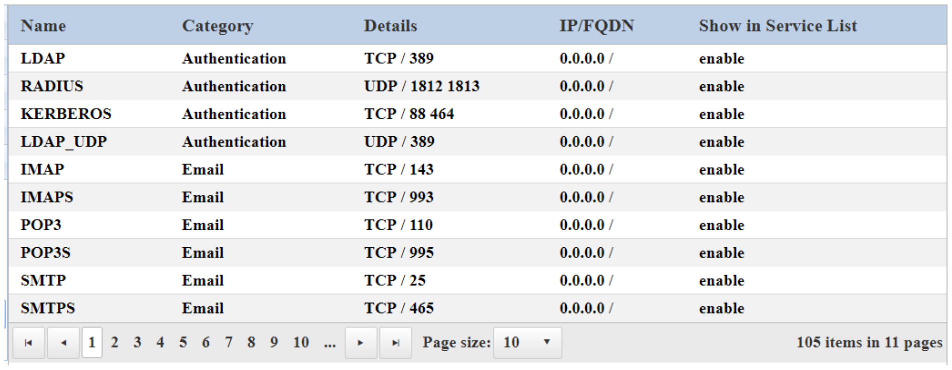


Name	Type	Details	Interface	Visibility
*.live.com	ipmask		0.0.0.0 0.0.0.0	enable
Adobe Login	ipmask		0.0.0.0 0.0.0.0	enable
Camera	ipmask		172.20.0.0 255.255.240.0	enable
Cisco core	ipmask		192.168.104.2 255.255.255.255	enable
Class Building D	iprange		10.10.129.1 10.10.129.60	enable
Class Bulding A	iprange		10.10.65.2 10.10.65.50	enable
Class Bulding G	iprange		10.10.193.2 10.10.193.60	enable
Class Bulding H	iprange		10.11.1.2 10.11.1.30	enable
Class Bulding L	iprange		10.11.65.2 10.11.65.25	enable
DC_DNS1	iprange		192.168.1.101 192.168.1.103	enable

1 2 3 4 5 6 7 8 9 10 ... Page size: 10 114 items in 12 pages

FIGURE (7): THE OBJECT ADDRESSES OF MSA FORTI FIREWALL

The system get to interpretation (NAT) controls likewise were come back from the (MSA Forti) firewall. It has three segments, and 30 things showed in three pages. It included routed IP addresses so it isn't like to distribute the Figure. The "Routing Table" was discovered unfilled, as the administrator clarified that it isn't should have been designed yet. So, the next item was (object addresses). It was found 114 items displayed in 12 pages as shown in Figure (7). The (Object Service) Table was found in the JSON file and displayed as shown on the next Figure (8). It contains 21 items displayed in three pages. The system administrator was happy with the execution of the FPM tool. The administrator clarified that he has 20 distinctive firewall devices models. Each firewall model has a tool that exclusively manages it. So having one tool that can deal with numerous models of firewall is great. Likewise, the manner in which the approach rules are shown and kept up is critical for the system administrator. He also recommended with adding filters in order to display the columns is important to make it less demanding to survey numerous strategy rules.

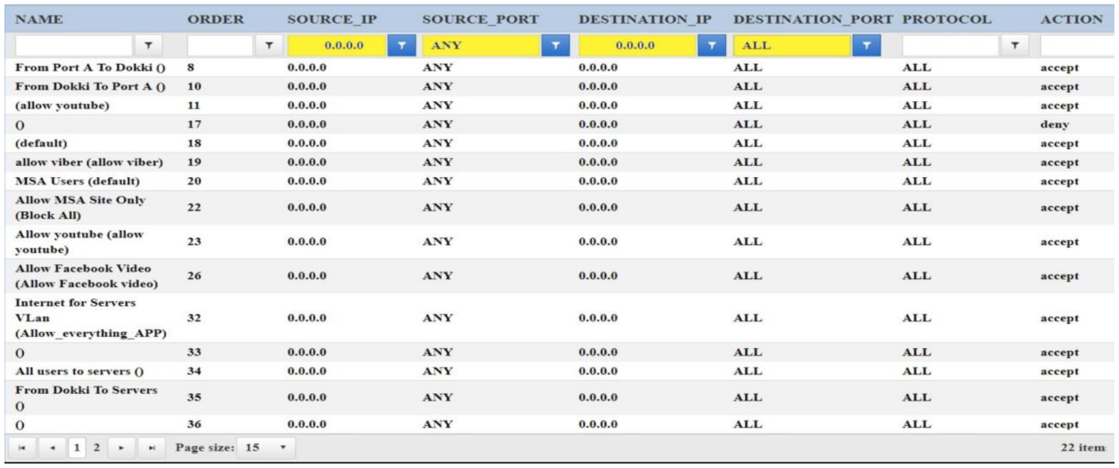


Name	Category	Details	IP/FQDN	Show in Service List
LDAP	Authentication	TCP / 389	0.0.0.0 /	enable
RADIUS	Authentication	UDP / 1812 1813	0.0.0.0 /	enable
KERBEROS	Authentication	TCP / 88 464	0.0.0.0 /	enable
LDAP_UDP	Authentication	UDP / 389	0.0.0.0 /	enable
IMAP	Email	TCP / 143	0.0.0.0 /	enable
IMAPS	Email	TCP / 993	0.0.0.0 /	enable
POP3	Email	TCP / 110	0.0.0.0 /	enable
POP3S	Email	TCP / 995	0.0.0.0 /	enable
SMTP	Email	TCP / 25	0.0.0.0 /	enable
SMTPS	Email	TCP / 465	0.0.0.0 /	enable

1 2 3 4 5 6 7 8 9 10 ... Page size: 10 105 items in 11 pages

FIGURE (8): THE (OBJECT SERVICE) OF (MSA FORTI) FIREWALL

The required filters have just been included as appeared in Figure (9). The policy rules are filtered to detect the risk rules. It is a risk to use generalization rules, by allowing to (all) source addresses and (all) destination addresses (all) services, for example. The specified values should be identified by the source or destination addresses to avoid one of the most rule anomalies. These filters also make it easier to go directly to certain rules. As it displayed there are many pages of policy rules so it is hard to browse them one by one for searching certain rule.



NAME	ORDER	SOURCE_IP	SOURCE_PORT	DESTINATION_IP	DESTINATION_PORT	PROTOCOL	ACTION
		0.0.0.0	ANY	0.0.0.0	ALL		
From Port A To Dokki ()	8	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
From Dokki To Port A ()	10	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
(allow youtube)	11	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
()	17	0.0.0.0	ANY	0.0.0.0	ALL	ALL	deny
(default)	18	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
allow viber (allow viber)	19	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
MSA Users (default)	20	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
Allow MSA Site Only (Block All)	22	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
Allow youtube (allow youtube)	23	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
Allow Facebook Video (Allow Facebook video)	26	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
Internet for Servers VLan (Allow_everything_APP)	32	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
()	33	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
All users to servers ()	34	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
From Dokki To Servers ()	35	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept
()	36	0.0.0.0	ANY	0.0.0.0	ALL	ALL	accept

1 2 Page size: 15 22 item

FIGURE (9) : THE (RISK) DETECTED IN POLICY RULES OF (MSA FORTI) FIREWALL

In Figure (9) it is noticed that the estimation of (0.0.0.0) that showed any IP address has been doled out to numerous guidelines in both if their source and destination addresses. The esteem (ANY) is doled out to the source port. The esteem (ALL) is allocated to the destination port. At that point, the policy rules are sifted and brought about 22 rules. These came about standards speaking to various activities for the system traffic that originate from any source address originates from all source ports and targeted on any destination address through any destination port. It is fundamental for the system administrator to survey these rules as it is considered as high risk rules. They are positively fair or should be changed so as to show signs of improvement execution.After excluding high risk rules, the remaining rules are examined to detect anomalies with the splitting of these rules if there is an overlap between them as shown previously.

TABLE (III).THE RESULTS OF USING THE PROPOSED FPM TOOL FOR INVESTIGATING THE OBTAINED FIREWALL RULE SET

Total Number of Rules	High-Risk Rules	Deleted Redundant Rules	Anomaly Rules	Disjointed (Optimized Rules)
83	22	20	16	25

A. *Resolution of the detected anomaly rules*

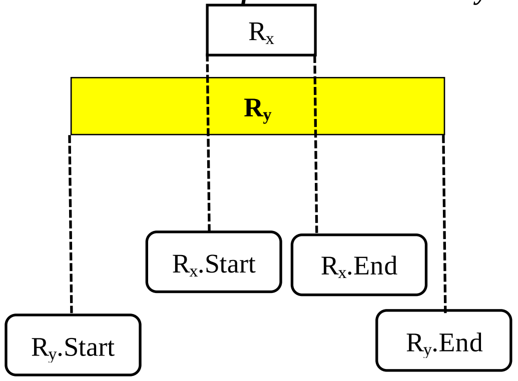


Figure (10): Intersection between source IP-address of two anomaly rules

As shown in Figure (10), the two anomaly rules R_x and R_y are intersected or overlapped in their source IP-address range. So that R_x may be shadowed or generalized or correlated by R_y. In [6] the authors stated the overlapping problem in general and they believed in three types only of rule anomalies. They considered "generalization" is not one of the rule anomalies. However, some generalized rules may be intersected with many other rules causing anomalies. As shown in Figure (9) the high risk rules detected by using our proposed tool are an example of this mentioned problem. So that, it will be a challenge to try to resolve the detected anomalies before put these high risk rules away from the rule-set. The detected redundant rules also should be deleted from the rule-set as they will not affect the firewall policy. The deletion of the redundant rules from the rule set is the correction action that mentioned in many previous work such as [3-5]. As shown in Table (III), there were (20) deleted redundant rules and (16) anomaly rules have been detected detected in the real-time obtained rule-set. There also (22) rules considered as high-risk rules so that they have been excluded from the rule-set untill reviewd by the administrator. To resolve the detected (16) anomaly rules we extended the overlapping algorithm proposed by [6] as shown in Figure (11).

Algorithm. Solving shadowing, generalization, and correlation anomaly between two rules

```
1. Function Detecting_Disjoint_Range ($R_x$.Start, $R_x$.End, $R_y$.Start, $R_y$.End)
2.  {
3.         Function get-new-range ($R_x$.Start, $R_x$.End, $R_y$.Start, $R_y$.End)
4.          {
5.         Left ← min ($R_x$.Start, $R_y$.Start);
6.         Right ← max ($R_x$.End, $R_y$.End);
7.            If Left=$R_y$.Start and Right=$R_y$.End then
8.         Superset = Ry and Subset = Rx;
9.         Else
10.        Superset = Rx and Subset = Ry;
11.        End
12.          }
13.        Return $R_{xd}$ (Subset.Start, Subset.End);
14.        Return $R_{yd1}$(Superset.Start, Subset.Start-1);
15.        Return $R_{yd2}$(Subset.End+1, Superset.End);
16.        OptimizedList= $R_{xd}$;
17.        OptimizedList=$R_{yd1}$;
18.        OptimizedList=$R_{yd2}$;
19.    }
```

Figure (11): The proposed algorithm for solving rule-set anomaly

Table (IV): Sample of the anomaly rules shown in Table (I) after correction

Rule Order	Protocol	Src. IP	Src. Port	Dst. IP	Dst. Port	Action
Rule 1a	UDP	10.5.0.1-	53	60.117.38.5	80	Deny

		10.5.24.254				
Rule 1b	UDP	10.5.26.1-10.5.255.254	53	60.117.38.5	80	Deny
Rule 2a	UDP	10.5.15.1-10.5.15.14	25	172.33.1.2	53	Allow
Rule 2b	UDP	10.5.15.16-10.5.15.254	25	172.33.1.2	53	Allow
Rule 6	UDP	10.5.15.15	25	172.33.1.2	53	Deny
Rule 7	TCP	10.5.15.10	80	172.33.*.*	80	Allow
Rule 8	UDP	10.5.25.*	53	60.117.38.5	80	Allow
Rule 15a	TCP	10.5.15.1-10.5.15.9	80	172.33.38.5	80	Deny
Rule 15b	TCP	10.5.15.11-10.5.15.254	80	172.33.38.5	80	Deny

Table (IV) shows the rule-set after disjointing the overlap of the source IP-address range between two rules.

Table (V): The obtained rule-set after correction of the anomaly

Total Number of Rules Became	High-Risk Rules	Deleted Redundant Rules	Anomaly Rules	Disjointed (Optimized Rules)
91	22	20	-	49

Table (V) shows the total number of rules after solving the anomaly. The total number of rules was 83 rules. It became 91 rules. The proposed solution aimed to disjoint the intersected range of the source IP-address. The proposed approach resulted in 4 outcome lists to be reviewed by the administrator. The high-risk rules list, the deleted redundant rules list, the anomaly rules list and finally the optimized rules list.

V. Discussion

The results ofthe real-time firewall (MSA Forti) were maintained using FPM tool, and presented using filters as shown in Figure (10).A total of 22 rules were detected and identified with our knowledge under a high risk rules. This result alone was appreciated by the network administrator at the data center where we run FPM. He confirmed that few system administrators have been changed in the course of recent years and each is relied upon to include a lot of policy rules of this sort. At that point the 20 redundant rules were recognized utilizing the FPM device and so, the correction action is to delete these redundant rules as they didn’t affect the policy. The restorative activity of these excess rules is the erasure of the higher order rules and push the most minimal order into the optimized rules list. As the lower order rules were of higher need while filtering the traffics using the firewall device. At that point, the recognized 15 anomaly rules are pushed into the anomaly rules list. The FPM tool is considered as a safe tool. It didn't change the firewall rule set. It just acquired a copy of its policy rule set under the supervision and control of the system administrator. At that point it produces four lists of the acquired firewall policy rule set subsequent to examining it. These lists separate between the disjointed (optimized) rules and the anomaly rules. In this manner the FPM is an associate device for the administrators not supplanting them.

VI. Conclusion and Future Work

This paper presents a new approach to retrieving a copy of the policy rule set for multiple firewall devices operating in real-time. An algorithm was also proposed to detect conflict in policy rules. It also sheds light on the high risk of some policy rules that actually intersect or overlapped with the rest of the rules, which may be due to leaving the default rule that comes bundled with the firewall device and allows to pass all the packets coming from any source and target any destination through all protocols. An algorithm for the detection of anomaly was also proposed based on the examination of the address space and the extent of the overlap, if any, between the two rules being examined. When this overlap is discovered, the two rules are divided to extract two non-overlapping (disjointed) rules and other intersected. Then the intersected rules are examined by the anomaly detection algorithm. The outcome of the proposed algorithms is classified the obtained firewall rule set into four class lists. The disjointed rules, the anomaly rules, the deleted redundant rules, and the high risk rules. The FPM proposed tool based on the proposed approach is tested in a data center and acquired a firewall policy rule set in real-time. It is safe tool as it does not make any physically changes on the running firewall devices. It is considered as an assistant tool for the network

administrators. It also satisfies the visualization, usability as well as the efficient detection of firewall policy rules anomaly.
On the future, it is planned to run the FPM tool in many data centers and test another firewall models of multiple vendors.

Acknowledgment
We are gratefully thanking Mr. George Ezzat, Eng. Amjad Mohamed and KhaledSoliman from IP Protocol Company for their technical advice. An appreciation for Eng. Adaham Mohamed is a must for his great support. Special thanks for Eng. Ahmed Mandor, MostafaHanafy, M. Zakaria. M. Hassan, A. Aly and Amr Hassan from IRA, for their technical effort and support.

References

[1] Mayer, A. Wool and E. Ziskind, "Offline firewall analysis," International Journal of Information Security 5 (3), 2005, pp. 125–144.
[2] E. Al-Shaer and H. Hamedl," Firewall policy advisor for anomaly detection and rule editing", In Proceedings of Data and Application Security (LNCS4127), March 2006.
[3] L. Yuan, H. Chen, J. Mai, C. Chuah, Z. Su, P. Mohapatra, and C.Davis, "Fireman: A Toolkit for Firewall Modeling and Analysis" , Proc. IEEE Symp. Security and Privacy, p. 15, 2006
[4] A. Hanamsagar, N. Jane, B. Borate, A. Wasvand, and S. A. Darade, "Firewall Anomaly Management : A survey," vol. 105, no. 18, pp. 1–5, 2014.
[5] E. Al-Shaer and H. Hamed, "Discovery of Policy Anomalies in Distributed Firewalls," IEEE INFOCOM '04, vol. 4, 2004. pp. 2605-2616.
[6] M. Abedin, S. Nessa, L. Khan, and B. Thuraisingham, "Detection and resolution of anomalies in firewall policy rule ", Data and Applications Security XX, pages 15-29, 2006.
[7] Y. Bartal, A.J. Mayer, K. Nissim, A. Wool, "Firmato: A novel firewall management toolkit," ACM Transactions on Computer Systems 22, 2004, pp. 381-420.
[8] Martínez A. Yannuzzi M. López J. Serral-Gracià R. Ramirez W. (2015). Applying information extraction for abstracting and automating the CLI-based configuration of network devices in heterogeneous environments.
[9] Kim H., Ko S., Kim D. S. and Kim H. K. (2017). Firewall ruleset visualization analysis tool based on segmentation. IEEE Symposium on Visualization for Cyber Security (VizSec), Phoenix, AZ, pp. 1-8. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8062196&isnumber=8062166
[10] Tran, T., Al-Shaer, E., &Boutaba, R. (2007). PolicyVis: Firewall Security Policy Visualization and Inspection. LISA.
[11] X. Wang et al. (2018). PNPL: Simplifying programming for protocol-oblivious SDN networks. Computer Networks. 147, 64–80.
[12] Antonio JesúsFernández-García, Luis Iribarne, Antonio Corral, Javier Criado, James Z. Wang. (2018). A flexible data acquisition system for storing the interactions on mashup user interfaces. Computer Standards & Interfaces, Volume 59, Pages 10-34, ISSN 0920-5489. International Journal of Scientific & Engineering Research, Volume 6, Issue 2, February-2015. ISSN 2229-5518.
[13] Voronkov, A., Iwaya, L.H., Martucci, L.A., &Lindskog, S. (2017). Systematic Literature Review on Usability of Firewall Configuration. ACM Comput. Surv, 50, 87:1-87:35.

FIRST AUTHOR: Ass.Prof. Dr . Mohamed.M.A.Elgazzar, He received the B.Sc.degreefrom MTC, Cairo, Egypt, in 1984 and M.Sc.degree in Computer NetworksfromFaculty of Engineering,Cairo University,Egypt, in 1991 and the Ph.D. degree in Computer Networks at Faculty of Engineering, from Cairo University, Egypt, in 1995.He has experience of 33 years which includes both academic and research.He is currently an Associate Professor in Computer Science Department at Higher institute of Computer Science and information systems fifth Settlement, New Cairo, Egypt. His research interests Computer Networks, Machine Learning, and Network Security. He published several research papers.

SECOND AUTHOR: Dr. Eng. Mohamed EzzatElhamahmy is a cyber-technology expert. He is from Egypt. He has received his Ph. D. in computer networks security from Faculty of Computers and information, Cairo University, 2011. He has received his B.Sc. in Computer Engineering from MTC, Cairo in 1989. He received his M. Sc. in Computer Engineering, Faculty of Engineering, Al-Azhar University, Cairo in 2001. His past research was in building text to speech synthesis system in Arabic. His current research interests are Network Security and Machine Learning.

Third AUTHOR: Abdel-Hamid Emarareceived the B. S., M. S., and Ph.D. degree in computers engineering from Al-Azhar University in 1992, 2000, 2006, respectively. He works in Computers and Systems Engineering Department at Faculty of Engineering, Al-Azhar University,Cairo, Egypt. He has experience of 12 years which includes both academic and research. He is currently an Assistant Professor in Computer Science Department at the Taibah University, Al Madinah Al Monawarah, KSA. His research interests educational data mining, Machine Learning, Arabic text mining, and intelligent, and adaptive systems. He published several research papers.