2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA) | 978-1-6654-2529-2/20/$31.00 ©2021 IEEE | DOI: 10.1109/CyberSA52016.2021.9478256

An Anomaly Free Distributed Firewall System for SDN

Mitali Sinha
Indian Institute of Technology
Bhubaneswar, India
Email: a20ee09019@iitbbs.ac.in

Padmalochan Bera
Indian Institute of Technology
Bhubaneswar, India
Email: plb@iitbbs.ac.in

Manoranjan Satpathy
Indian Institute of Technology
Bhubaneswar, India
Email: manoranjan@iitbbs.ac.in

Abstract—**Firewall is a core element of a network security system which takes care of the availability, privacy, and integrity of network resources. However, managing a system with large scale, heterogeneous policies is complex and error prone. In multi-firewall systems, it is very important to configure the policy rules in the relevant firewall to prevent malicious flow into the network. Any modification to a firewall rule or insertion of a new rule needs intra and inter firewall conflict resolution to find correct mapping of rule to firewall. In SDN, the controller generates flow rules for different switches depending on application requirements and network topologies. Firewall can be used as a first line of defense against different attacks to the data plane and the control plane of SDN. The state-of-art work on firewall implementation in SDN shows various anomalies that may introduce functional failures and security violations. In this paper, we have proposed a novel approach for distributed anomaly-free firewall implementation on SDN controller. Here, the controller receives the firewall policies of different domains through north bound API and resolves the intra and inter firewall conflicts and derives a single anomaly-free firewall policy at the controller level. When the controller receives a packet_in message from a switch at run time, it selects a conflict free rule from global policy and sends it to appropriate switches. We have evaluated our proposed distributed firewall system for different network topologies under different attacking scenarios, e.g., DDoS attacks and experimental results are reported. The results show the efficacy of our solution in terms of reduced malicious traffic flows, improvements in CPU utilization of controller, packet loss and response time of legitimate packets.**

Index Terms—**SDN, Distributed firewall policy, intra and inter firewall anomalies, DDoS attack.**

I. Introduction

In traditional network, firewall is one of the essential defense component to protect the system from unauthorized access and suspicious traffic. Management of the firewall rules is a crucial task because of interdependency among the rules. There are research works to manage and analyze the security rules in a traditional network. FDD (Firewall Decision Diagram) [1] is designed to generate a complete and consistent firewall policy. A distributed firewall architecture [2] is designed to support centralized firewall policy. A tool called Firewall Policy Advisor [3] is implemented to detect anomalies in firewall rules in pairwise manner in a distributed firewall. FIREMAN [4] is introduced to check configuration errors in firewall policies using statistical analysis.

In SDN, all the firewall policies are defined at the controller and it installs the firewall rules as a configuration file, ofp_flow_mod to the switches of the system. The configuration or installation of firewall policy can be done in two ways: proactive and reactive. In a proactive firewall, the controller sends the firewall rules to the switches in advance before communication begins. In a reactive firewall, when a new packet_in comes to the controller, it selects one of the matched rules from the firewall policy and installs the rule as ofp_flow_mod to the switches. Now each switch acts as a firewall in distributed manner. A proactive firewall is useful in a scenario when there is a single firewall table or there is no inter firewall dependency with static firewall policy. But, a reactive firewall is useful in a multiple firewalls system with inter firewall dependency and dynamic firewall policy. The major contributions of this paper are as follows.
(1) We have introduced two different priority distribution policies: network, policy oriented and discussed different types of anomalies in single or distributed firewall system of SDN.
(2) We have proposed a novel technique to build a distributed firewall system in reactive mode that resolves these anomalies. In this approach, the controller collects the firewall rules for different domains of SDN system through north bound API (NBI) and derives a global anomaly free firewall table. When the controller receives a packet_in message from a switch, it installs suitable firewall rules in the appropriate switches.
(3) We have implemented our proposed distributed firewall system on POX controller and compare its performance with a proactive distributed firewall. In the first experiment, we compare the installation time of firewall policies in a proactive firewall and in our proposed distributed firewall. In the second experiment, the performance of two distributed firewall systems is measured under different DDoS attacking scenario. The paper is organized as: Related Work (part II), An overview of Access Control Rule (part III), ACL anomalies and a novel approach to remove anomalies in a distributed firewall SDN system(part IV), Experimental Implementation and Evaluation (part V) and finally conclusion and future work(part VI).

II. Related Work

In SDN, firewall is a software with access control rules (ACL) to filter the traffic flows and executed inside the controller [5,7]. The functions of a firewall are changed as per the user requirements directly in software without need of any specific hardware or individual changes in forwarding

TABLE I: ACL rule relations

Notation	Description
$Ri\ \phi\ Rj$	Condition filed of Ri and Rj are disjoint.
$Ri = Rj$	Condition field of Ri and Rj are equal
$Ri \subset Rj$	Condition field of Ri is subset of Rj
$Ri \supset Rj$	Condition field of Ri is superset to Rj
Ri correlated Rj	Condition field of Ri and Rj are correlated

devices [6,9]. A layer-2 [11] and a layer-3 [8,9] firewall is implemented on POX controller and ACL rules are installed as flow rules in all the switches of the network proactively. A topology aware packet filtering layer-2 firewall [10] is implemented on POX controller which consists of only 'deny' rules. It inserts the rule at a appropriate switch based on network topology. Similar to [10], a layer-3 firewall [12] is proposed. The applications introduced in [5-12], are packet filtering firewall which take the decision based on content flags of individual packet (such as IP address, MAC address or port number); they are proactive and distributed in nature. These studies [5-12] do not consider the interdependency between the firewall rules. Another firewall application which keeps trace of connections between client and server is known as circuit gateway firewall [13-16]. We have observed that in the existing literature, there is no implementation of distributed firewall in SDN that considers intra and inter rule dependency between firewall tables. In this paper, we proposed a reactive distributed firewall implementation while considering intra and inter firewall policy conflicts.

III. An Overview of Access Control Rule (ACL)

An ACL or firewall rule is defined as (condition, priority, action). The Condition in ACL consists of various fields such as protocol, source IP, destination IP, source port and destination port. The value of these fields are used to match with all the incoming packets. Priority is used to resolve the conflict when a packet matches with more than one ACL rule. The action of an ACL rule is of two types such as 'accept' and 'deny'. A set of firewall rules is defined as firewall policy.

A. ACL rule relations in firewall policy

In order to build a conflict free firewall model, we need to find out all possible relations which exist between ACL rules in firewall policy by comparing condition fields of ACL rules. The relation between two firewall rules Ri and Rj shown in Table.I is defined by E.Al-Shaer et al [3].

B. Priority assigned to ACL rules

Conflict resolution among ACL rules in a firewall policy can be solved by assigning different priority values to each conflicting rule. Some techniques for conflict resolution in firewall policy is introduced in [17]. The conflict resolution strategy for firewall policy can be of two types such as network oriented strategy and policy oriented strategy.

1) Network oriented Strategy: Based on the risk factor of a network, network administrator chooses a firewall rule when a packet matches with more than one rule. It can be of two types such as Deny-prioritized, Accept-prioritized.

2) Policy oriented Strategy: Based on the factors related to a firewall policy such as the time at which the rule is defined, who defines the rule and the purpose of rule, the priority is assigned to firewall rules. Various types of policy-oriented strategy are recent prioritized, specific prioritized, high-majority prioritized, first match prioritized and high authority prioritized.

IV. Our proposed Reactive distributed firewall for Anomaly resolution

We have designed a distributed firewall system in reactive mode. Before describing our proposed firewall system, we mathematically define anomalies in single and distributed firewall system.

A. Anomalies in a single firewall SDN system

Anomaly in a Firewall policy is defined as presence of more than one rule in an incorrect order which matches the same packet or presence of rules which are never be executed. The various anomalies present in single firewall policy is described as follows.

Shadowing anomaly: A rule Ri shadows rule Rj if:

$$Ri = Rj, Ri[priority] > Rj[priority], Ri[action] \neq Rj[action].$$
$$Ri \supset Rj, Ri[priority] > Rj[priority], Ri[action] \neq Rj[action].$$

In this situation, Rj is never executed and all the incoming matched packets of Ri and Rj get executed by Ri because of its higher priority.

Correlation anomaly: Two rules Ri and Rj have correlated anomalies if:
$Ri correlated Rj, Ri[priority] = Rj[priority], Ri[action] \neq Rj[action]$.
In this situation, all the incoming matched packets of Ri and Rj are executed according to first match prioritized policy.

Generalization Anomaly: Rule Ri is generalization of Rj if:

$$Ri \supset Rj, Ri[priority] < Rj[priority], Ri[action] \neq Rj[action].$$

In this situation, Rj will get more preference than Ri and all the incoming matched packets of (Ri - Rj) will be executed as rule Ri.

Redundancy Anomaly: A rule Rj is redundant to rule Ri if:

$$Ri = Rj, Ri[priority] > Rj[priority], Ri[action] = Rj[action].$$
$$Ri \supset Rj, Ri[priority] > Rj[priority], Ri[action] = Rj[action].$$

Similar to shadowing anomaly, in this situation, Rj is never executed and all the incoming matched packets of Ri and Rj get executed by Ri because of its higher priority.

Irrelevant anomaly: An ACL rule in firewall policy is irrelevant if this rule does not match with any traffic pass through that firewall. In other word, a packet with source and destination address of irrelevant rule does not pass through that firewall. A rule Ri is irrelevant if:

$$\forall packet : packet[source\_IP] \neq Ri[source\_IP],$$
$$packet[destination\_IP] \neq Ri[destination\_IP].$$

In this situation, Ri is an extra unused rule in the policy.

B. Anomalies in a distributed firewall SDN system

A single network can have multiple firewall policies with many network administrators. Each domain has its own security policy and applications. Some domain in a network may allow multicast traffic which may be denied by other domain. Due to presence of multiple firewalls in a single network, there is a high possibility of inter firewall policies anomalies even if absent of intra firewall policy anomalies. Figure 1 shows a SDN system with multiple firewalls. This system consists of two firewall policies such as FW1 for domain1 and FW2 for domain2. These firewall policies are sent to controller by their domain administrators through north bound API. After collecting the firewall rules, the controller sends the firewall policies to their respective firewall proactively as shown in Fig. 1(a) and Fig. 1(b). When a packet passes through a firewall, it will allow or deny according to the policy defined within that domain. In this section, we have discussed various types of anomalies present in a distributed firewall system of SDN with absent of anomalies in intra firewall policy as follows.

Shadowing Anomaly: Shadowing anomaly in a distributed firewall occurs if a traffic is denied by its preceding firewall (F_p) with lower, higher or equal priority whereas it could have been accepted by a following firewall (F_f) in its identified path.

A rule Rj in firewall table F_f is shadowed by rule Ri in F_p if:

$$case(1) : Ri = Rj, Ri[priority] <, >, = Rj[priority],$$
$$Ri[action] = deny, Rj[action] = accept.$$
$$case(2) : Ri \supset Rj, Ri[priority] <, >, = Rj[priority],$$
$$Ri[action] = deny, Rj[action] = accept.$$
$$case(3) : Ri \subset Rj, Ri[priority] <, >, = Rj[priority],$$
$$Ri[action] = deny, Rj[action] = accept.$$
$$case(4) : Ri \subset Rj, Ri[priority] <, >, = Rj[priority],$$
$$Ri[action] = accept, Rj[action] = accept.$$
$$case(5) : Ri \supset Rj, Ri[priority] <, >, = Rj[priority],$$
$$Ri[action] = deny, Rj[action] = deny.$$

In cases (1) and (2), a preceding firewall completely blocks a traffic which is accepted by its next firewall table. Rules (R1/FW1, R1/FW2) and rules (R2/FW1, R2/FW2) in Fig. 1 belong to case(1) and (2) respectively. Rules (R3/FW1, R3/FW2) in Fig. 1 belong to case(3). In cases (3) (4) (5), a preceding firewall partially blocks a traffic which is accepted by its next firewall. case(4) will be executed when the default firewall rule is 'deny' and case(5) will be executed when the default rule is 'accept'. In the Fig. 1, we have taken default rule as accepted, so rules (R4/FW1, R4/FW2) belong to case(5).

Fake traffic Anomaly: Fake traffic anomaly in a distributed firewall occurs if a traffic is accepted by its preceding firewall (F_p) with low priority whereas after reaching its following firewall it is blocked with higher priority.

A rule Ri in firewall table F_p allows the fake traffic, which is blocked by Rj in F_f if:

$$case(6) : Ri = Rj,\ Ri[priority] < Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$
$$case(7) : Ri \supset Rj, Ri[priority] < Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$
$$case(8) : Ri \subset Rj, Ri[priority] < Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$
$$case(9) : Ri \subset Rj, Ri[priority] < Rj[priority],$$
$$Ri[action] = deny, Rj[action] = deny.$$

In cases (6) and (7), a preceding firewall completely allows a fake traffic which is blocked by its next firewall table. Rules (R5/FW1, R5/FW2) and rules (R6/FW1, R6/FW2) belong to case(6) and (7) respectively in Fig. 1. In cases (8),(9) a preceding firewall partially allows a fake traffic which is blocked by its next firewall. Rules (R7/FW1, R7/FW2) and rules (R8/FW1, R8/FW2) belong to case(8) and (9) respectively in Fig. 1.

Real traffic Anomaly: Real traffic anomaly in a distributed firewall occurs if a traffic is accepted by its preceding firewall (F_p) with higher priority whereas after reaching its following firewall (F_f), it is blocked with lower priority.

Rule Rj in firewall table (F_f) blocks the real traffic, which is allowed by Ri in F_p if:

$$case(10) : Ri = Rj, Ri[priority] > Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$
$$case(11) : Ri \supset Rj, Ri[priority] > Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$
$$case(12) : Ri \subset Rj, Ri[priority] > Rj[priority],$$
$$Ri[action] = accept, Rj[action] = deny.$$

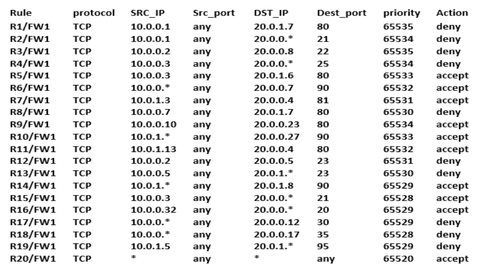
In cases (10) and (12), a following firewall completely blocks a genuine traffic which is allowed by its preceding firewall table. Rules (R9/FW1, R9/FW2) and rules (R11/FW1, R11/FW2) belong to case(10) and (12) respectively in Fig. 1. In case(11), a following firewall partially blocks a genuine traffic which is allowed by its preceding firewall table. Rules (R10/FW1, R10/FW2) belong to case(11) in Fig. 1.

Redundancy Anomaly: It occurs when more than one firewall table having same deny rule for a traffic in its identified path. A rule Rj in F_f is redundant to rule Ri in F_p if:

$$case(13) : Ri = Rj, Ri[priority] >, =, < Rj[priority],$$
$$Ri[action] = deny, Rj[action] = deny.$$
$$case(14) : Ri \supset Rj, Ri[priority] >, =, < Rj[priority],$$
$$Ri[action] = deny, Rj[action] = deny.$$

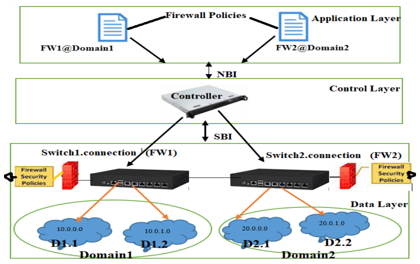
In both cases (13) and (14), Rj is an extra copy present in F_f which is already blocked by Ri rule in its preceding firewall F_p. Rules (R12/FW1, R12/FW2) and rules (R13/FW1, R13/FW2) belong to case(13), case(14) respectively in Fig.1.

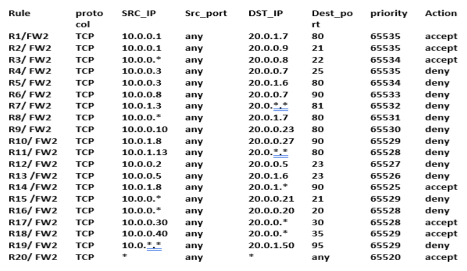
Correlated Anomaly: This anomaly occurs, if two rules are correlated to each other, and one is placed in a preceding



Rule	protocol	SRC_IP	Src_port	DST_IP	Dest_port	priority	Action
R1/FW1	TCP	10.0.0.1	any	20.0.1.7	80	65535	deny
R2/FW1	TCP	10.0.0.1	any	20.0.0.*	21	65534	deny
R3/FW1	TCP	10.0.0.2	any	20.0.0.8	22	65535	deny
R4/FW1	TCP	10.0.0.3	any	20.0.0.*	25	65534	deny
R5/FW1	TCP	10.0.0.3	any	20.0.1.6	80	65533	accept
R6/FW1	TCP	10.0.0.*	any	20.0.0.7	90	65532	accept
R7/FW1	TCP	10.0.1.3	any	20.0.0.4	81	65531	accept
R8/FW1	TCP	10.0.0.7	any	20.0.1.7	80	65530	deny
R9/FW1	TCP	10.0.0.10	any	20.0.0.23	80	65534	accept
R10/FW1	TCP	10.0.1.*	any	20.0.0.27	90	65533	accept
R11/FW1	TCP	10.0.1.13	any	20.0.0.4	80	65532	accept
R12/FW1	TCP	10.0.0.2	any	20.0.0.5	23	65531	deny
R13/FW1	TCP	10.0.0.5	any	20.0.1.*	23	65530	deny
R14/FW1	TCP	10.0.1.*	any	20.0.1.8	90	65529	accept
R15/FW1	TCP	10.0.0.3	any	20.0.0.*	21	65528	accept
R16/FW1	TCP	10.0.0.32	any	20.0.0.*	20	65529	accept
R17/FW1	TCP	10.0.0.*	any	20.0.0.12	30	65529	deny
R18/FW1	TCP	10.0.0.*	any	20.0.0.17	35	65528	deny
R19/FW1	TCP	10.0.1.5	any	20.0.1.*	95	65529	deny
R20/FW1	TCP	*	any	*	any	65520	accept

(a) Firewall policy for Domain1(FW1)





Rule	protocol	SRC_IP	Src_port	DST_IP	Dest_port	priority	Action
R1/FW2	TCP	10.0.0.1	any	20.0.1.7	80	65535	accept
R2/ FW2	TCP	10.0.0.1	any	20.0.0.9	21	65535	accept
R3/ FW2	TCP	10.0.0.*	any	20.0.0.8	22	65534	accept
R4/ FW2	TCP	10.0.0.3	any	20.0.0.7	25	65535	deny
R5/ FW2	TCP	10.0.0.3	any	20.0.1.6	80	65534	deny
R6/ FW2	TCP	10.0.0.8	any	20.0.0.7	90	65533	deny
R7/ FW2	TCP	10.0.1.3	any	20.0.*.*	81	65532	deny
R8/ FW2	TCP	10.0.0.*	any	20.0.1.7	80	65531	deny
R9/ FW2	TCP	10.0.0.10	any	20.0.0.23	80	65530	deny
R10/ FW2	TCP	10.0.1.8	any	20.0.0.27	90	65529	deny
R11/ FW2	TCP	10.0.1.13	any	20.0.*.*	80	65528	deny
R12/ FW2	TCP	10.0.0.2	any	20.0.0.5	23	65527	deny
R13 /FW2	TCP	10.0.0.5	any	20.0.1.6	23	65526	deny
R14 /FW2	TCP	10.0.1.8	any	20.0.1.*	90	65525	accept
R15 /FW2	TCP	10.0.0.*	any	20.0.0.21	21	65529	deny
R16/ FW2	TCP	10.0.0.*	any	20.0.0.20	20	65528	deny
R17/ FW2	TCP	10.0.0.30	any	20.0.0.*	30	65528	accept
R18/ FW2	TCP	10.0.0.40	any	20.0.0.*	35	65529	accept
R19/ FW2	TCP	10.0.*.*	any	20.0.1.50	95	65529	deny
R20/ FW2	TCP	*	any	*	any	65520	accept

(b) Firewall policy for Domain2(FW2)

Fig. 1: Proactive distributed firewall in SDN

firewall (F_p) and other is placed in a following firewall (F_f). It occurs not only when the actions of two correlated rules are different but also when the actions of two correlated rules are same in a distributed firewall . Two correlated rules Ri in F_p and Rj in F_f generates anomaly if:

$$case(15) : Ri\ correlated\ Rj, Ri[priority] <, >, = Rj[priority], Ri[action] = accept, Rj[action] = accept.$$

No anomaly for $(Ri \cap Rj)$, but it may generate $(Ri - Rj)$ or $(Rj - Ri)$ anomaly traffic. Rules (R14/FW1, R14/FW2) belong to case(15) as shown in Fig. 1.

$$case(16) : Ri\ correlated\ Rj, Ri[priority] < Rj[priority], Ri[action] = accept, Rj[action] = deny.$$

$(Ri \cap Rj)$ is the fake traffic which is allowed to reach at F_f and it may generate $(Ri - Rj)$ or $(Rj - Ri)$ anomaly traffic. Rules (R15/FW1, R15/FW2) belong to case(16) as in Fig. 1.

$$case(17) : Ri\ correlated\ Rj, Ri[priority] > Rj[priority], Ri[action] = accept, Rj[action] = deny.$$

$(Ri \cap Rj)$ is the genuine traffic which is denied by F_f and it may generate $(Ri - Rj)$ or $(Rj - Ri)$ anomaly traffic. Rules (R16/FW1, R16/FW2) belong to case(17) as shown in Fig. 1.

$$case(18) : Ri\ correlated\ Rj, Ri[priority] < Rj[priority], Ri[action] = deny, Rj[action] = accept.$$

$(Ri \cap Rj)$ is the genuine traffic which is denied by F_p. It may generate $(Ri - Rj)$ or $(Rj - Ri)$ anomaly traffic. Rules (R17/FW1, R17/FW2) belong to case(18) as shown in Fig. 1.

$$case(19) : Ri\ correlated\ Rj, Ri[priority] > Rj[priority], Ri[action] = deny, Rj[action] = accept.$$

No anomaly for $(Ri \cap Rj)$, but there is the possibility of $(Ri - Rj)$ or $(Rj - Ri)$ anomaly traffic. Rules (R18/FW1, R18/FW2) belong to case(19) as shown in Fig. 1.

$$case(20) : Ri\ correlated\ Rj, Ri[priority] <, >, = Rj[priority], Ri[action] = deny, Rj[action] = deny.$$

It only blocks $(Ri \cap Rj)$ fake traffic at F_p but it may generate $(Ri - Rj)$ or $(Rj - Ri)$ traffic anomaly. Rules (R19/FW1, R19/FW2) belong to case(20) as shown in Fig. 1.

C. *Our proposed anomaly detection scheme*

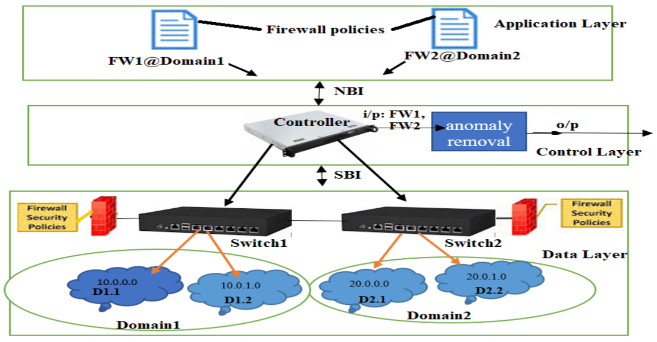
Our proposed distributed firewall system in SDN is shown in Fig. 2. The controller collects the firewall policy for different domains from the respective network administrators. After collecting the firewall policies, it sends these rules to the anomaly removal module and outputs a global anomaly free firewall table. The pseudo code for anomaly removal module is presented in Algorithm 1.

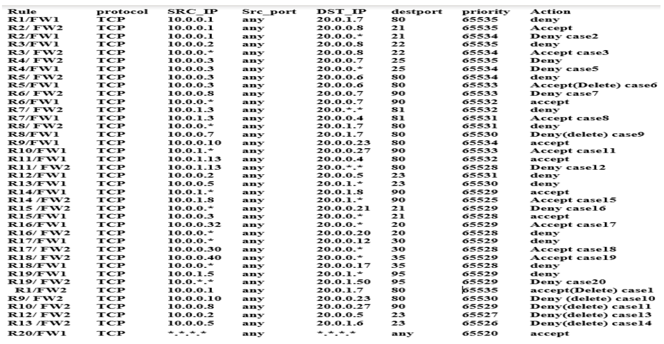
Algorithm 1 Anomaly removal module

```
Input: $(FW = FW1 \cup FW2 \cup FW3, \dots, FWn)$
Output: A Global Anomalies free policy.
1: $N \leftarrow length(FW)$
2: for $i \leftarrow 2$ to $N$ do
       $k \leftarrow i - 1$
3:     for $j \leftarrow 1$ to $k$ do
           if $(\sim (Rj \Phi Ri)$ then
               if $(Rj = Ri)$ then
4:                 Go to Algorithm2
               if $(Rj \subset Ri)$ then
5:                 Go to Algorithm3
               if $(Rj \supset Ri)$ then
6:                 Go to Algorithm4
               if $(Rj\ correlated\ Ri)$ then
7:                 Go to Algorithm5
```

Algorithm 2 Rj=Ri

```
if $(Rj = Ri)$ then
    if $(Ri[priority] > Rj[priority])$ then
        Remove Rj and place Ri in index of Rj;
    if $(Ri[priority] < Rj[priority])$ then
        Remove Ri; break;
    if $(Ri[priority] = Rj[priority])$ then
        if network risk factor is high then
            Remove rule with action accept
        else
            Remove rule with action deny
```





Rule	protocol	SRC_IP	Src_port	DST_IP	destport	priority	Action
R1/FW1	TCP	10.0.0.1	any	20.0.1.7	80	65535	deny
R2/ FW2	TCP	10.0.0.1	any	20.0.0.8	21	65535	Accept
R2/FW1	TCP	10.0.0.1	any	20.0.0.*	21	65534	Deny case2
R3/FW1	TCP	10.0.0.2	any	20.0.0.8	22	65535	deny
R3/ FW2	TCP	10.0.0.*	any	20.0.0.8	22	65534	Accept case3
R4/ FW2	TCP	10.0.0.3	any	20.0.0.7	25	65535	Deny
R4/FW1	TCP	10.0.0.3	any	20.0.0.*	25	65534	Deny case5
R5/ FW2	TCP	10.0.0.3	any	20.0.0.6	80	65534	deny
R5/FW1	TCP	10.0.0.3	any	20.0.0.6	80	65533	Accept(Delete) case6
R6/ FW2	TCP	10.0.0.8	any	20.0.0.7	90	65533	Deny case7
R6/FW1	TCP	10.0.0.*	any	20.0.0.7	90	65532	accept
R7/ FW2	TCP	10.0.1.3	any	20.0.*.*	81	65532	deny
R7/FW1	TCP	10.0.1.3	any	20.0.0.4	81	65531	Accept case8
R8/ FW2	TCP	10.0.0.*	any	20.0.1.7	80	65531	deny
R8/FW1	TCP	10.0.0.7	any	20.0.1.7	80	65530	Deny(delete) case9
R9/FW1	TCP	10.0.0.10	any	20.0.0.23	80	65534	accept
R10/FW1	TCP	10.0.1.*	any	20.0.0.27	90	65533	Accept case11
R11/FW1	TCP	10.0.1.13	any	20.0.0.4	80	65532	accept
R11/ FW2	TCP	10.0.1.13	any	20.0.*.*	80	65528	Deny case12
R12/FW1	TCP	10.0.0.2	any	20.0.0.5	23	65531	deny
R13/FW1	TCP	10.0.0.5	any	20.0.1.*	23	65530	deny
R14/FW1	TCP	10.0.1.*	any	20.0.1.8	90	65529	accept
R14 /FW2	TCP	10.0.1.8	any	20.0.1.*	90	65525	Accept case15
R15 /FW2	TCP	10.0.0.*	any	20.0.0.21	21	65529	Deny case16
R15/FW1	TCP	10.0.0.3	any	20.0.0.*	21	65528	accept
R16/FW1	TCP	10.0.0.32	any	20.0.0.*	20	65529	Accept case17
R16/ FW2	TCP	10.0.0.*	any	20.0.0.20	20	65528	deny
R17/FW1	TCP	10.0.0.*	any	20.0.0.12	30	65529	deny
R17/ FW2	TCP	10.0.0.30	any	20.0.0.*	30	65528	Accept case18
R18/ FW2	TCP	10.0.0.40	any	20.0.0.*	35	65529	Accept case19
R18/FW1	TCP	10.0.0.*	any	20.0.0.17	35	65528	deny
R19/FW1	TCP	10.0.1.5	any	20.0.1.*	95	65529	deny
R19/ FW2	TCP	10.0.*.*	any	20.0.1.50	95	65529	Deny case20
R1/FW2	TCP	10.0.0.1	any	20.0.1.7	80	65535	accept(Delete) case1
R9/ FW2	TCP	10.0.0.10	any	20.0.0.23	80	65530	Deny (delete) case10
R10/ FW2	TCP	10.0.0.8	any	20.0.0.27	90	65529	Deny(delete) case11
R12/ FW2	TCP	10.0.0.2	any	20.0.0.5	23	65527	Deny(delete) case13
R13 /FW2	TCP	10.0.0.5	any	20.0.1.6	23	65526	Deny(delete) case14
R20/FW1	TCP	*.*.*.*	any	*.*.*.*	any	65520	accept

(a) Global anomaly free Firewall policy

Fig. 2: Our Proposed Reactive Distributed firewall in SDN

Algorithm 3 Rj $\subset Ri$

```
if ($Rj \subset Ri$) then
    if ($Rj[priority] > Ri[priority]$) then
        Insert Ri after Rj
    if ($Rj[priority] < Ri[priority]$) then
        Remove Rj and place Ri in index of Rj
    if ($Rj[priority] = Ri[priority]$) then
        if Ri[action]=Rj[action] then
            Remove Rj; Break;
        else
            if ((Rj[action]=="deny" and network risk factor is high) or (Rj[action]=="accept" and network risk factor is low)) then
                Insert Ri after Rj
            else
                Remove Rj;
```

Algorithm 4 Rj $\supset Ri$

```
if ($Rj \supset Ri$) then
    if ($Rj[priority] > Ri[priority]$) then
        Remove Ri; Break;
    if ($Rj[priority] < Ri[priority]$) then
        Insert Ri before Rj ;
    if ($Ri[priority] = Rj[priority]$) then
        if Ri[action]=Rj[action] then
            Remove Ri ; Break;
        else
            if ((Ri[action]=="deny" and network risk factor is high) or (Ri[action]=="accept" and network risk factor is low)) then
                Insert Ri before Rj
            else
                Remove Ri; Break;
```

Algorithm 5 Rj correlated Ri

```
if ($Rj$ correlated $Ri$) then
    if ($Ri[priority] > Rj[priority]$) then
        Place Ri before Rj
    if ($Ri[priority] = Ri[priority]$) then
        if Ri[action]=Rj[action] then
            Place Ri before or after Rj.
        else
            if ((network risk factor is high and Ri[action]="deny") or (network risk factor is low and Ri[action]="accept")) then
                Place Ri before Rj
            else
                place Ri after Rj.
```

Algorithm 6 Rules Distribution in Appropriate Switches

```
if ($Packet\_in[header] \in (R1, R2, R3, ..., Rn)$) then
    if ($R1[priority] > R2[priority] > R3[priority] > ... > Rn[priority]$) then
        $Rf = Packet\_in[header] \cap R1[conditionfield]$
        Rf[ideal timeout]: 30
        Rf[hard timeout]: "PERMANENT"
        if ($R1[action] ==$ "accept") then
            Send Rf to switches present in the identified paths between Rf[src_ip] to Rf[dst_ip]
        else
            Send Rf to most preceding switch contains Rf[src_ip] host
```

When a new packet arrives at a switch, it sends a packet_in message to the controller and the controller selects the most appropriate firewall rule from the anomaly free global firewall table. Then it takes a conjunction between the selected rule, the header of packet_in and sends the resultant rule to the appropriate switches. If the action of firewall rule is 'deny', then the controller sends the rule to its most preceding firewall at the data layer. If the action is 'accept', then it sends the rules to all the firewalls along the identified paths at the data layer. The distribution of firewall rules in its appropriate firewalls is described in Algorithm 6. The *timeout* of each firewall rule depends on two values such as *idle timeout* and *hard timeout*. We set the *ideal timeout* as any integer and *hard timeout* as permanent. So, if the rule is in used by incoming traffic, it will be present at the respective switch after first request. Otherwise, the rule will be dropped from the switch after *ideal timeout* period and the search time for matched rule doesn't increase at switch level firewall. Here, we have explained our proposed anomaly removal algorithm to remove intra and inter-firewall anomalies with the example shown in Fig. 1.

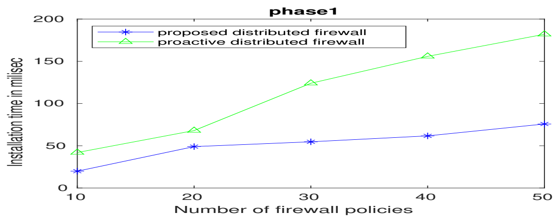
The controller collects the two firewall policies: FW1 for domain1 and FW2 for domain2 as shown in Fig. 2. After collecting the firewall policies, the controller sends it to the anomaly removal module and the output comes as a global anomaly free firewall policy as shown in Fig. 2(a). R1/FW2 is shadowed by R1/FW1 of case(1) and removed from firewall policy by applying Algorithm 2. The anomaly (R2/FW1, R2/FW2) of case(2) is resolved by making R2/FW1 as generalization of R2/FW2 (Algorithm 4). The anomaly (R3/FW1, R3/FW2) of case(3) is resolved by making R3/FW2 as generalization of R3/FW1 (Algorithm 3). The anomaly (R4/FW1, R4/FW2) of case(5) is resolved by making R4/FW1 as generalization of R4/FW2 (Algorithm 4). The rules R5/FW1, R7/ FW1, R8/FW1 of case(6), case(8), case(9) respectively are removed from firewall policy due to fake traffic anomaly (Algorithm 2 (case(6)), (Algorithm 3 (case(8) and (9))). The anomaly (R6/FW1, R6/FW2) of case(7) is resolved by making R6/FW1 as generalization of R6/FW2 (Algorithm 4). R9/FW2 is removed from firewall policy due to real traffic anomaly of case(10)(Algorithm 2). Similarly, R10/FW2 is removed from firewall policy due to real traffic anomaly of case(11) (Algorithm 4). R11/FW2 is the generalization of R11/FW1 of case(12) (Algorithm 3). R12/FW2 is removed from firewall policy due to redundancy anomaly of case(13) (Algorithm 2). Similarly, R13/FW2 is removed from firewall policy due to redundancy anomaly of case(14) (Algorithm 4). The correlated conflict between firewall rules (R14/FW1, R14/FW2) of case(15), (R15/FW1, R15/FW2) of case(16), (R16/FW1, R16/FW2) of case(17), (R17/FW1, R17/FW2) of case(18), (R18/FW1, R18/FW2) of case(19), (R19/FW1, R19/FW2) of case(20) as discussed in previous section is resolved by placing them in proper order (Algorithm 5). When a new packet with header (protocol:tcp, Src_ip:10.0.0.9, Src_port:80, Dst_ip:20.0.0.8, Dst_port:22) arrives at switch, the switch creates packet_in message of this packet and sends it to the controller. The controller selects the most appropriate rule from global firewall table based on first matched prioritized i.e R3/FW2 (protocol:tcp, Src_ip: 10.0.0.*, Src_port:any, Dst_ip:20.0.0.8, Dst_port:22) . It makes the conjunction as:
Header[packet]: tcp, 10.0.0.9, 80, 20.0.0.8, 22
R3/FW2: tcp, 10.0.0.*, any, 20.0.0.8, 22, accept
R_f:tcp, 10.0.0.9, 80, 20.0.0.8, 22, accept
It sends rule R_f to switch1 and switch2. Here the action of firewall rule is 'accept', so it sends the rule to both switch1 and switch2. Let us take another example when the action of the rule is 'deny'. A new packet with header (protocol:tcp, Src_ip:10.0.0.3, Src_port:80, Dst_ip:20.0.0.8, Dst_port:25) comes to switch, it sends the packet_in message to controller. The controller selects R4/FW1(tcp, 10.0.0.3, any, 20.0.0.*, 25, deny) as rule from global firewall table. It makes a conjunction and sends R_f (tcp, 10.0.0.3, 80, 20.0.0.8, 25, deny) to only switch1 because of deny action. By sending the rule with deny action to one switch only, it reduces the redundancy anomaly at the switch.

V. Experimental Analysis and Evaluation:

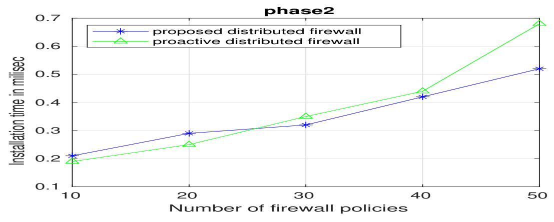
Our proposed distributed firewall system in reactive mode is implemented on POX controller and the network topologies are created using mininet emulator. We evaluate the performance of our proposed distributed firewall in terms of installation time in two phases and compare it with the installation time of a proactive distributed firewall. In proposed distributed firewall system, for phase1, the controller collects firewall policies of different domains through northbound API and make an anomaly free global firewall table and for phase2, when a packet_in message comes to the controller, it installs the correct firewall rule to its appropriate switches. However, in the proactive distributed firewall, for phase1, the controller collects firewall policies of different domains and proactively sends it to its respective switches. For phase2, when a new packet matches with a rule of 'deny' action, it immediately drops the packet at the switch, but when it matches with a rule of 'accept' action, the packet_in message is sent to the controller. In the next experiment, the performance of our proposed firewall is compared with the proactive distributed firewall in terms of traffic redundancy control under DDoS attack.

A. *Experiment 1 (Timing Analysis)*

We have calculated the installation time for proactive distributed firewall and our proposed reactive distributed firewall in two different scenarios. In the first scenario, number of rules for each switch remains fixed (10) and we take four linear topologies with switches varying from 10 to 50. In the second scenario, we have taken a linear topology with 50 switches and increased the number of rules of each switch from 100 to 500. Each switch consists of ten hosts. Fig. 3(a) and 3(b) show the installation time of our proposed distributed firewall and proactive distributed firewall for phase1 and phase2 under scenario-1 respectively. Similarly, Fig. 4(a) and 4(b) show the installation time of our proposed distributed firewall and proactive distributed firewall for phase1 and phase2 under scenario-2 respectively. From experimental result, we can see that if number of security policy of the system increases, the installation time for phase1 and phase2 increases in both the schemes, but our proposed firewall system requires less installation time as compare to proactive distributed firewall.

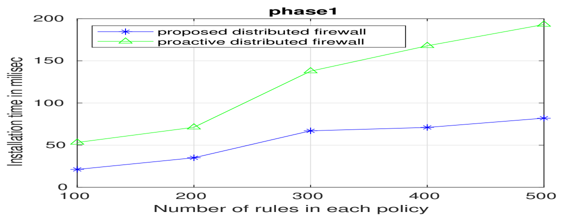


(a) Installation Time for phase1

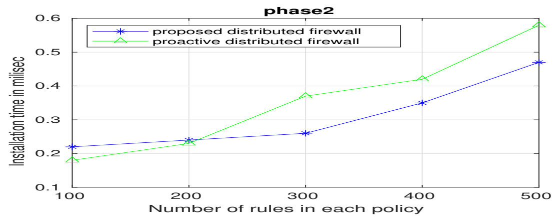


(b) Installation Time for phase2

Fig. 3: Scenario-1



(a) Installation Time for phase1



(b) Installation Time for phase2

Fig. 4: Scenario-2

The average response time of an user under benign scenarios depends upon installation time for phase-2. Faster installation of routing rules leads to lesser response time of packets.

In order to know the variation in number of inter firewall anomalies without intra firewall anomaly with respect to number of firewalls polices, we have conducted an experiment with scenario-1 and used our proposed anomaly removal module. From Fig. 5, one can infer that if the number of firewall policy increases, inter-firewall anomaly increases. However, the rate of inter-firewall anomalies depends on the firewall application context and the dependencies between firewall rules.

B. Experiment 2 (Security Analysis)

In this experiment, we have taken a linear topology with 50 switches and 500 hosts. Each switch has its own firewall policy with 100 rules. We have taken four different cases of DDoS attacks. The four cases are described below.
Case-1: DDoS attack from any two hosts of 5 switches.

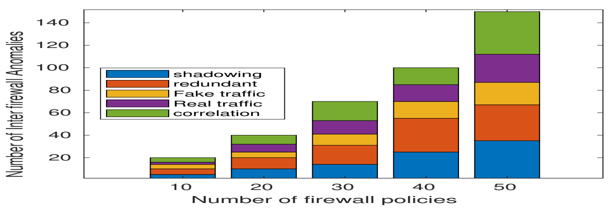


Fig. 5: Inter-firewall Anomalies detection using proposed reactive firewall

Case-2: DDoS attack from any two hosts of 10 switches.
Case-3: DDoS attack from any two hosts of 50 switches.
Case-4: DDoS attack from any two hosts of 100 switches.
At the same time a legitimate user sends 100 different ping requests to a server of the network. The deny rules for attacker hosts and accept rule for legitimate hosts are present in distributed firewall policies. We have compared the performance of our proposed distributed firewall system with proactive distributed firewall and SDN without firewall security in terms of reduction in data traffic ratio (%), average response time of legitimate packets, packet loss of legitimate user (%) and controller CPU load (%) under four DDoS attack cases.

Reduction in data traffic ratio: We measure a reduced data traffic ratio, which is defined as the ratio of total data traffic volume before and after the deployment of firewall system. The reduced data traffic ratio under four cases of DDoS attack after deployment of proactive and our proposed distributed firewall systems is shown in Fig. 6. It shows that, our proposed distributed firewall system could reduce the data traffic approximately 50% more than that of the proactive distributed firewall. This is because our scheme deploys the 'deny' rules to its most preceding firewall in order to avoid fake traffic anomaly.

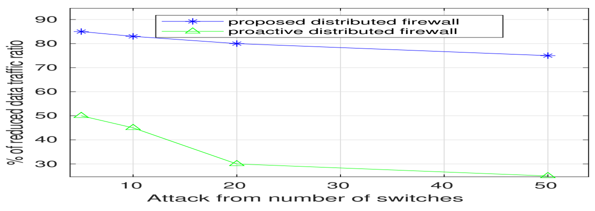


Fig. 6: Reduced Data traffic ratio under DDoS attack

Controller's CPU load (%): Figure 7 shows the CPU load of controller with proposed distributed firewall, proactive distributed firewall and SDN without security under four scenarios of DDoS. The CPU load of controller with proposed distributed firewall is approximately 40% less than that of the proactive distributed firewall. This is because our proposed scheme can block the fake packets as early as possible in the preceding firewall without forwarding the packets to following firewall.

Average response time of legitimate packets: Figure 8 shows the average response time of legitimate packets under

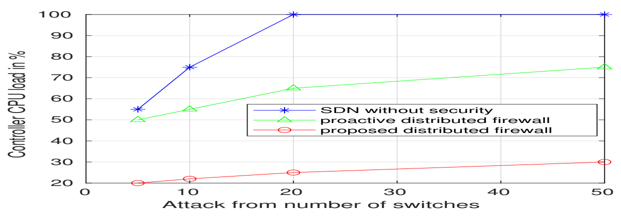


Fig. 7: Controller's CPU load(%) under DDoS attack

four scenarios of DDoS attack with proposed distributed firewall, proactive distributed firewall and SDN without security. The average response time of legitimate packets with proposed distributed firewall is less than that of the proactive distributed firewall and SDN without security.

Packet loss of legitimate user (%): Figure 9 shows packets

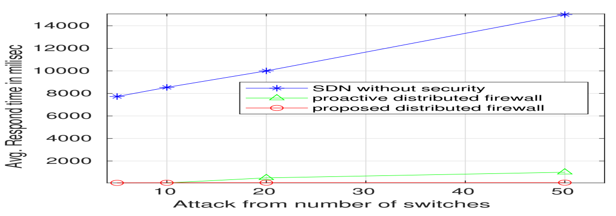


Fig. 8: Avg. response time of legitimate packets under DDoS attack

loss of legitimate user in percentages under four scenarios of DDoS attack with proposed distributed firewall, proactive distributed firewall and SDN without security. The packet loss of legitimate user with proposed distributed system is zero percent due to less congestion in control layer.

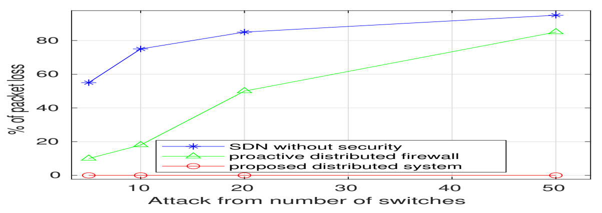


Fig. 9: Packet loss of Legitimate user under DDoS(%) attack

VI. Conclusion and Future work

A Firewall security system requires a controlled administration of intra and inter firewall anomalies to provide real security assistance. Thus, a network having single or distributed firewall cannot provide proper security to the system without anomaly resolution. In this paper, We have not only discussed various anomalies in single or distributed SDN firewall system but also discussed various algorithms to overcome these anomalies. A reactive distributed firewall system is implemented on POX controller in order to resolve these anomalies. The experimental studies have been conducted in order to know the performance of our proposed distributed firewall system with a proactive firewall system in terms of timing analysis and security analysis. From experimental results, we can infer that our proposed distributed firewall system is better than proactive distributed firewall system, as in case of installation time of firewall rules. Our proposed distributed firewall system also provides better security under DDoS attack.

Firewall policy administration is a very interesting and vast area to work. There are a lot of things to do in this area. so the future study will be conducted on auto-configurable firewalls and the rule editing in the firewall policies based on this study.

References

[1] M. Gouda, and X. Liu "Firewall Design: Consistency, Completeness, and Compactness," Proc. 24th Int'l Conf. Distributed Computing Systems (ICDCS '04), p. 327, 2004.
[2] S. Ioannidis, A. Keromytis, S. Bellovin, and J. Smith "Implementing a Distributed Firewall," Proc. Seventh ACM Conf. Computer and Comm. Security, p. 199, 2000.
[3] E. Al-Shaer, and H. Hamed "Discovery of Policy Anomalies in Distributed Firewalls,"IEEE INFOCOM '04, vol. 4, pp. 2605-2616, 2004.
[4] 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.
[5] J. G. V. Pena and W. E. Yu "Development of a distributed firewall using software defined networking technology," in Proceedings of the 4th IEEE International Conference on Information Science and Technology, ICIST 2014, pp. 449–452, IEEE, China, April 2014.
[6] K. Kaur, J. Singh, K. Kumar, and N. S. Ghumman "Programmable firewall using Software Defined Networking," in Proceedings of the 2nd International Conference on Computing for Sustainable Global Development, INDIACom 2015, pp. 2125–2129, IEEE, India, March 2015.
[7] S. Morzhov, I. Alekseev, and M. Nikitinskiy "Firewall application for FloodLight SDN controller," in Proceedings of the International Siberian Conference on Control and Communications, SIBCON 2016, Russia, May 2016.
[8] A. Kumar and N. K. Srinath "Implementing a firewall functionality for mesh networks using SDN controller," in Proceedings of the 1st IEEE International Conference on Computational Systems and Information Technology for Sustainable Solutions, CSITSS2016, pp. 168–173, IEEE, India, October 2016.
[9] M. Suh, S. H. Park, B. Lee, and S. Yang "Building firewall over the software-defined network controller," in Proceedings of the 16th International Conference on Advanced Communication Technology: Content Centric Network Innovation!, ICACT 2014,pp. 744–748, Republic of Korea, February 2014.
[10] T. V. Tran and H. Ahn "A network topology-aware selectively distributed firewall control in sdn," in Proceedings of the International Conference on Information and Communication Technology Convergence (ICTC), pp. 89–94, IEEE, Oct 2015
[11] T.javid, T. Riaz, and A. Rasheed "A Layer2 Firewall for Software Defined Network," in Information Assurance and Cyber Security (CIACS), 2014 conference on. IEEE, 2014, pp. 39-42.
[12] S. Kim et al "Secure Collecting, Optimizing, and Deploying of Firewall Rules in Software-Defined Networks," IEEE Access, vol. 8, pp. 15166-15177, 2020.
[13] V. Gupta, S. Kaur, and K. Kaur "Implementation of stateful firewall using POX controller," in Proceedings of the 3rd International Conference on Computing for Sustainable Global Development, INDIACom 2016, pp. 1093–1096, India, March 2016.
[14] A. chowdhary, D.Hauang, A.Alshamrani etal. "SDFW: SDN-based stateful distributed firewall," http://arxiv.org/abs/1811.00634
[15] P.Krongbaramee, Y.Somchit "Implementation of SDN stateful firewall on data plane using open vswitch," in proceeding of the 15th international joint conference on Computer Science and Engineering (JCSSE), pp.1-5, Nakhonpathom, July 2018.
[16] M. Caprolu, S. Raponi and R. Di Pietro "FORTRESS: An Efficient and Distributed Firewall for Stateful Data Plane SDN," Security and Communication Networks, vol. 2019, 2019.
[17] T. Moses "Extensible Access Control Markup Language (XACML), Version 2.0, Oasis Standard, Internet," http://docs.oasis-open.org/xacml/2.0/accesscontrol-xacml-2.0-core-spec-os.pdf, 2005.