Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7922985

MODIFIED TREE RULE FIREWALL FOR REMOVING REDUNDANT AND SHADOWING RULES IN CLOUD FIREWALL POLICY

Dhwani Hakani[1], Palvinder Singh Mann[2*]
[1]Research Scholar, Gujarat Technological University, Ahmedabad, Gujarat, India, Adf_dhwani@gtu.edu.in
[2*]Associate Professor, Gujarat Technological University, Ahmedabad, Gujarat, India, * Corresponding Email ID - asso_psmaan@gtu.edu.in

Abstract

Most private networks are secured by firewalls, which are crucial for safety. A firewall aims to inspect every inward and outgoing traffic before deciding whether to allow it. The rule-based firewall is a frequently used conventional firewall. However, conventional Listed-Rule firewalls have limits when it comes to task performance and is ineffective when used with some networks that have very large firewall rule sets. This paper suggests a model firewall design, "Tree-Rule Firewall," which has advantages and works with expansive networks like "cloud". This paper proposes a modified tree rule firewall (MTRFcloud) for removing redundant and shadowing rules, improving cloud network security. This work first generates a tree rule firewall for the corresponding firewall policy. The suggested modified tree rule firewall does not produce redundant rules and efficiently finds the shadow rules. Then, a modified Tree-Rule firewall that manages firewall rules was tested in a cloud setting. It is shown that the updated Tree-Rule firewall provides faster processing and greater network security. With a big network, like a cloud network, the modified Tree-Rule firewall is simpler to construct and efficiently removes the redundant and shadow rules.

Keywords: Firewall, Tree rules, cloud security, redundant rule, shadowing rule

1. Introduction

Currently, firewalls are widely used on the Internet to safeguard network devices from unwanted or hostile traffic [1] that could risk the security, integrity, and accessibility of the services offered. Firewalls implement security policies as sequences of rules, each consisting of an action and a condition specified over a few packet header fields [2]. Incoming packets are compared progressively to rule conditions by firewalls using the first matching approach until a matching rule is discovered. At this point, the appropriate action is taken whether to allow or deny the packet. Network parameters specify firewall rules, such as the protocol type, source and destination IP addresses, and their respective port numbers [3]. A firewall rule typically has the following structure:

FR = (ruleId, protocol, srcIP, srcPort, destIP, destPort, action)

Where ruleId is a sequence order of a rule, the protocol includes {TCP, UDP, HTTP, *}, srcIP and destIP are IP address ranges from 0.0.0.0 to 255.255.255.255, srcPort and destPort are Port numbers ranging from 0 to 65536, and action includes allow or deny. The rule's action is carried out if the traffic meets the filtering rule's specifications; if not, the order sequence's next rule is carried out, and so on. Table 1 shows the sample firewall rules.

Table 1 Sample Firewall Rules

Rule –Id	Protocol	SrcIP	SrcPort	DestIP	DestPort	Action
R1	TCP	10.15.3.41	Any	3.5.12.8	25	Allow
R2	TCP	3.5.12.48	Any	192.108.1.16	80	Allow
R3	TCP	3.5.12.48	Any	192.108.1.17	25	Deny
R4	TCP	10.15.3.41	Any	3.5.12.8	25	Allow
R5	TCP	192.108.1.62	Any	3.5.12.46	8080	Deny
R6	TCP	3.5.12.48	Any	192.108.1.17	25	Allow

One of the most flexible, private, and easily available platforms is cloud computing, which offers robust services for information sharing over the Internet. With cloud computing, security is the most crucial concern [4]. The cloud firewall is required to safeguard the data centre from numerous threats. Using proper policies that a professional administrator has established will assure high levels of security. A cloud firewall and a normal firewall are identical, aside from the fact that a cloud firewall is situated within a cloud platform. Cloud firewalls are just traditional firewalls installed in the cloud and build a virtual wall to stop malicious network traffic.

A virtual firewall is a service that operates in a virtualized setting and offers the same packet filtering and monitoring functions as a physical firewall. In both virtual and physical contexts, virtual firewalls allow the deployment of network access controls among VMs and other sites. Under the framework of the virtualization environment, virtual firewalls are established [5]. A firewall can be installed as an appliance or service in a cloud environment. Figure 1 shows the firewall model in the cloud.

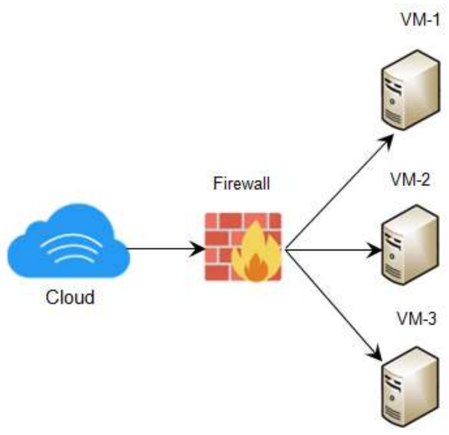


Figure 1 Firewall model in the cloud

In the domain of firewalls, rule disputes can be categorized into two groups: those that affect speed and security. These rule conflicts, caused by redundant and shadowed rules, significantly negatively affect how well classical firewalls perform. On a conventional firewall, shadowed rules specifically cause security issues [6]. Furthermore, a firewall's processing performance is slower when redundant rules are present. It is because they take up processing time on the firewall and are redundant with other rules. Hence, shadowing and unnecessary rules must be removed from the rule list to increase a firewall's operational speed.

This paper proposes a modified tree rule firewall for removing redundant and shadowing rules, improving cloud network security. This work first generates a tree rule firewall for the corresponding firewall policy. Then, the suggested modified tree rule firewall does not produce redundant rules and efficiently finds the shadow rules. The following is the key contribution of this research work:

- This work first constructs the tree rule firewall for cloud firewall policy. Then, it is a corresponding simplification of the original firewall policy.
- The redundant and shadowing rules are identified and removed based on the generated tree rule firewall.
- Significant numbers of rules are collected from the different numbers of the host. The tree rule firewall is implemented and evaluated using cloudsim.

The research paper is planned as follows: Section 2 describes the related work of tree-based rule firewalls and cloud firewalls. The definition of redundant and shadowing rule is explained in section 3. The suggested modified tree rule firewall is described in Section 4. Section 5 analyses the experimental findings, and Section 6 provides the paper's conclusion.

2. Related Works

2.1 Tree Rule Firewall

A Tree-Rule firewall developed by He et al. [7] does not produce duplicate or conflicting rules. However, the subsequent rule checking and ordering of larger rules after smaller rules cause the listed-rule firewalls to operate less efficiently. Therefore, the authors suggested the tree-rule firewall, which arranges rules in the structure of a tree rather than a list, to solve the mentioned issues. The Tree-Rule firewall uses rules in a tree-like data model, and it will follow the tree when deciding whether to transmit incoming traffic based on tree rules to make the decision more quickly.

According to Chomsiri et al. [8], the hashing algorithm used in a stateful tree rule firewall [9] takes far longer than validating packet header information with the proper conditions indicated in the associated rule. As a result, the author presented a hybrid approach [8] that included stateful and stateless rules. Unfortunately, the method is not suitable for security policies with many rules because the number of rules rapidly increases with the variety of rule field values, even though the resulting rule set is conflict-free.

The Tree rule firewall was developed by Suresh et al. [10] to improve firewall efficiency and remove the drawbacks of the listed rule firewall. The dynamic reordering of rules and the addition of temporary rules while maintaining the rule conditions further enhance the performance of tree rule firewalls by speeding up packet filtering. This approach illustrates a stateless firewall because it doesn't keep track of the packet's state and may be expanded for other applications by adding a state and doing a thorough inspection of each packet.

Trabelsi et al. [11] offer an analytical multilayer rapid firewall security method to increase firewall effectiveness. Using traffic parameters, analytical splay tree filters are used in this method to speed up packet filtering. Furthermore, it rearranged following the network traffic variance once reaching a particular threshold qualification. In other words, this technique can determine whether or not the dynamic splay tree filter's order needs to be updated to filter the upcoming network traffic session and forecast the ideal order pattern.

A stateful session table design for a splay tree firewall is presented by Trabelsi et al. in [12]. A splay tree firewall group hash tables by prefix length with a set of prefix length-designated splay trees to manage firewall rules. Each connection in the session table design uses a single hash slot to conserve memory, speed up firewall concurrency checks, and decrease hash computation.

2.2 Cloud Firewall

Jekese et al. [13] propose a virtual firewall enables strengthening the safety of the virtual environment, setting network traffic rules, and maintaining network security of the virtual infrastructure on a per-virtual machine basis. Open-source software is used to construct a private cloud, and a Tree-Rule firewall technique is used to manage the firewall rules. This technique filters packets tree-likely according to their attributes, such as IP addresses and protocols. Furthermore, to prevent the virtual firewall from becoming overloaded in this particular situation, the speed at which packets are filtered and processed has greatly increased.

Dezhabad et al. [14] suggest adaptive auto-scalability for the firewall in the cloud. This technique distributes the incoming traffic between various virtualized firewalls situated in one pool and has software deployed on them. Every virtual machine is given a queueing model to examine. The objective is to calculate the total amount of virtualized firewalls required in various time steps based on the traffic volume and the fraction of all requests that travel to each firewall.

In a cloud/cloudlets architecture, Bagheri et al. [15] outline a technique for transferring rules from a centralized firewall to decentralized micro firewalls. The solution necessitates rearranging traffic channels after rule migration to maintain the overall defence policy established in the network.

Praise et al. [16] created an inspection-based system to stop malevolent activity by verifying the message signature of incoming traffic. It concurrently integrates the potential of parallel fast pattern recognition and reinforcement learning, accumulating to an optimal solution as soon as possible. The RL technique processes the message signature in parallel while learning the environment. Furthermore, the RL technique incorporates pattern-matching, which checks the signature for rapid decision-making.

3. Definitions

This section explains the definitions of redundant and shadowing firewall rules with examples.

3.1 Redundant Rule

If each field in rule R_a equals its equivalent field in rule R_b, then the two rules match exactly.

$$\forall x: \quad R_a(x) \cap R_b(x) \neq \emptyset$$

$$Where\ x \in \{protocol, SrcIP, DestIP, SrcPort, DestIP, action\}$$

$$If\ R_a(proto) \cap R_b(proto) \neq \emptyset\ and\ R_a(IPaddr) \cap R_b(IPaddr) \neq \emptyset\ and\ \ R_a(port) \cap R_b(port) \neq \emptyset\ and\ R_a(action) \cap R_b(action) \neq \emptyset\ then\ R_a\ is\ redundent\ to\ R_b$$

A redundancy occurs when both rules have identical behaviours, and the matching element of rule R_a is a subset of rule R_b. The rule's outcome will remain unchanged if the pointless limitations are removed. Redundancy is thought to be an error. A duplicate rule raises the dimensions of the filtering rule table, which could increase searching times and take up more file storage even though it might not be used in the filtering decision. Finding duplicate policies is essential so the administrator can modify or do away with them. For instance, they are not required because R1 and R4 in Table 1 share the same matching fields and behaviours.

3.2 Shadowing Rule

If at least one field in Ra is not equivalent to the associated field in R_b, then the rules R_a and R_b are only partially matching.

$$\forall x: \quad R_a(x) \cap R_b(x) \neq \emptyset\ and\ R_a(action) \cap R_b(action) = \emptyset$$

$$Where\ \ x \in \{protocol, SrcIP, DestIP, SrcPort, DestIP\}$$

$$If\ R_a(proto) \cap R_b(proto) \neq \emptyset\ and\ R_a(IPaddr) \cap R_b(IPaddr) \neq \emptyset\ and\ \ R_a(port) \cap R_b(port) \neq \emptyset\ and\ R_a(action) \cap R_b(action) = \emptyset\ then\ R_a\ is\ shadowed\ by\ R_b$$

A shadowing occurs when the same element of rule R_a is a subset of rule R_b, yet their behaviours differ. Shadowing is a severe issue in the rule since it prohibits the shadowed rule from always taking consequences. As a result, approved traffic may be prohibited, and vice versa. Therefore, it is essential to identify shadow rules so that the admin can resolve the issue by moving or removing the shadowing rule. As an illustration, rules R3 and R6 in Table 1 have similar matching fields but carry out different actions; they shadow one another.

4. Modified Tree Rule Firewall

The proposed modified tree rule firewall is explained in this section. Figure 2 shows the workflow of the MTRFcloud.

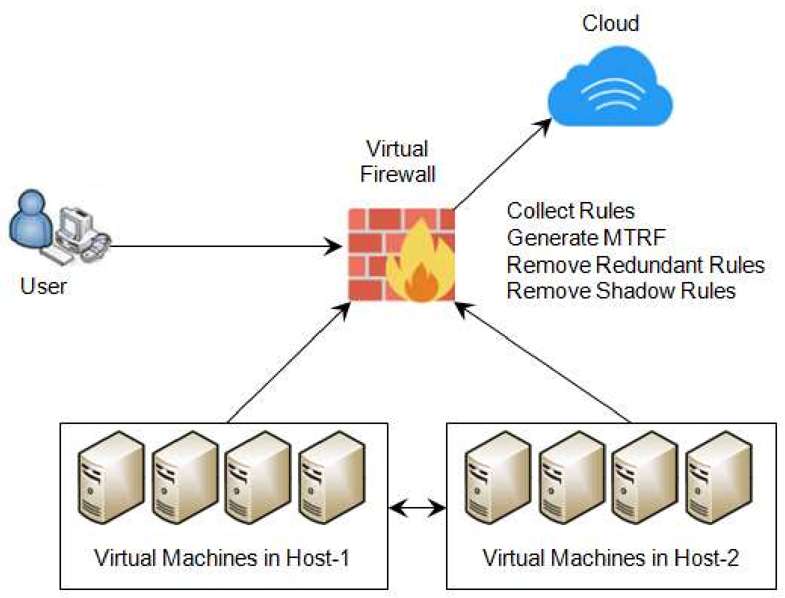


Figure 2 MTRFcloud Workflow

A new virtual firewall presents the rules dynamically rather than statically or manually. The firewall filters network traffic using IP addresses, protocols, and ports; it can also keep track of the flow's connection status and is therefore referred to as a stateful inspection firewall. First, this firewall will read the packet's attribute information and compare it to the information in its root nodes. The firewall will next examine the packet's additional properties sequentially by limiting its search to pertinent nodes at the appropriate levels. Thus, the traffic will be promptly determined using a certain action.

The modified tree rule firewall is generated using algorithm-1. The algorithm takes a list of the firewall as input and generates tree rules. Initially, the first rule is added to the firewall tree rule. Then, the remaining rules are added based on the node index. Here, the node indicates the fields in firewall rules (protocol type, source and destination IP, and Port).

Algorithm-1 Firewall Rule Tree Generation

```
Input: Firewall Rule List FR={$fr_1$,$fr_2$,$fr_{,3}$…,$fr_n$}
Output: Tree Rule TR
Step01: Insert rule $fr_1$ into TR
Step02: for i = 2 to n
Step03:   for each field ($f_j$) on $fr_i$
Step04:     Node_Index = checkIndex($f_j$)
Step05:     if( Node_Index < 0)
Step06:       insert $f_j$ into TR
Step07:     endif
Step08:   endfor
Step09: endfor
Step10: n_index=checkIndex(Field $f_j$, Node($f_j$))
Step11:   edges=$f_j$.getEdges()
Step12:   n_index=-1
```

```
Step13:   while(edges!=null and n_index < 0)
Step14:     ed=edges.next()
Step15:     if ed != null and ed = $f_j$
Step16:       n_index= Node.getIndex(ed)
Step17:     endfor
Step18:   endwhile
Step19:   return n_index
```

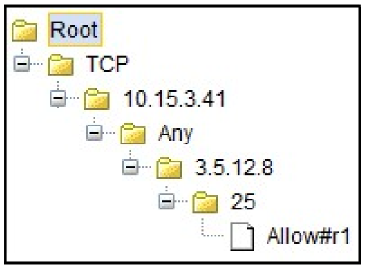
Working example for tree rule generation

Consider the list of firewall rules

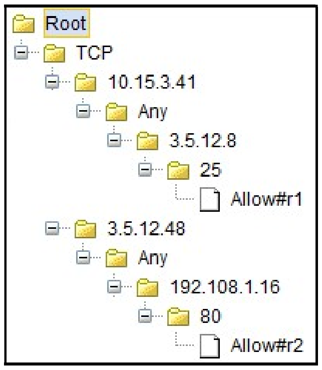
1. {TCP,10.15.3.41,Any,3.5.12.8,25,Allow}
2. {TCP,3.5.12.48,Any,192.108.1.16,80,Allow}
3. {TCP,3.5.12.48,Any,192.108.1.17,25,Deny}
4. {TCP,10.15.3.41,Any,3.5.12.8,25,Allow}
5. {TCP,192.108.1.62,Any,3.5.12.46,8080,Deny}
6. {TCP,3.5.12.48,Any,192.108.1.17,25,Allow}
7. {TCP, 10.15.3.86, Any, 2.2.2.91, 43, Allow }
8. {TCP, 12.20.25.20, Any, 10.15.1.20, 83, Allow }
9. {TCP, 10.15.3.37, Any, 12.20.25.84, 64, Deny }
10. {TCP, 10.15.3.40, Any, 21.20.25.12, 61, Allow }

The first rule {TCP,10.15.3.41, Any,3.5.12.8,25, Allow} is inserted into TR as shown in Figure 3 (a). Then, the for-loop is started for adding the remaining rules. Figure 3 (b) shows the insertion of rule-2 {TCP,3.5.12.48, Any,192.108.1.16,80, Allow}. For rule-3, {TCP,3.5.12.48, Any,192.108.1.17,25, Deny}, the source IP address 3.5.12.48 already exists in TR. It will check the index and add rule-3 (Figure 3 (c) shows rule-3 insertion). Figure 3 (d) the generation of tree rule firewall.

(a) After inserting rule-1

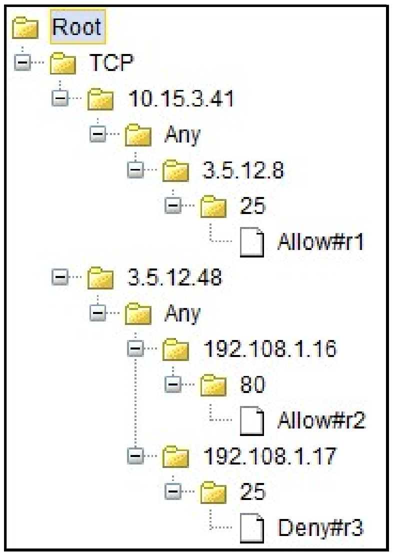


(b) After inserting rule-2



(c) After inserting rule-3

(d) Generated Tree Rule



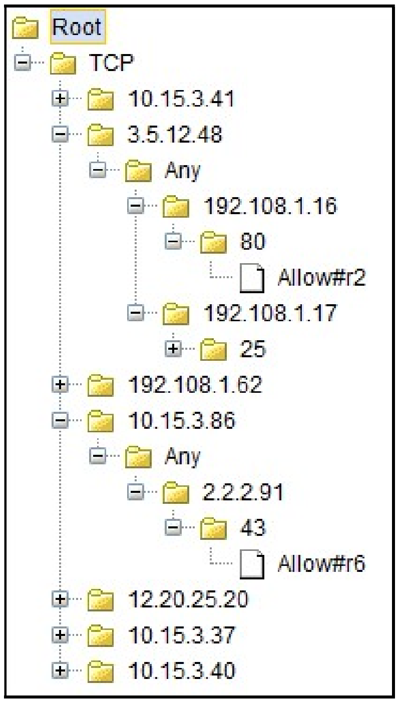


Figure 3 Tree Rule Generation

A redundant rule performs the same operation as another rule. For instance, Rule-1 in Table 1 is redundant to Rule-4. Removing a redundant rule shouldn't cause a firewall rule to vary. However, redundant rules slow things down since too many can take up too much processing time on the firewall. Therefore, the tree rule firewall automatically removes the redundant rules. The MTRFcloud does not add duplicate rules. In this example, rule 4 is automatically removed from the tree. The remaining rules are taken for further process.

1. {TCP,10.15.3.41,Any,3.5.12.8,25,Allow}
2. {TCP,3.5.12.48,Any,192.108.1.16,80,Allow}
3. {TCP,3.5.12.48,Any,192.108.1.17,25,Deny}
4. {TCP,192.108.1.62,Any,3.5.12.46,8080,Deny}
5. {TCP,3.5.12.48,Any,192.108.1.17,25,Allow}
6. {TCP, 10.15.3.86, Any, 2.2.2.91, 43, Allow }
7. {TCP, 12.20.25.20, Any, 10.15.1.20, 83, Allow }
8. {TCP, 10.15.3.37, Any, 12.20.25.84, 64, Deny }
9. {TCP, 10.15.3.40, Any, 21.20.25.12, 61, Allow }

Security issues will likely arise, particularly in a corporate network with many firewall rules. For example, consider a scenario in which a new worm attacks the network by sending packets. The firewall admin will add a new rule once this threat is discovered to provide a defence. If the existing rules above that permit attacker packets to pass through are obscured by this new rule, then a security issue has unquestionably arisen.

Several shadowing policies can spend the firewall's computation time on these pointless rules, which can cause speed issues. Furthermore, the shadowed rules must be executed to match the packets before the last rule because most packets will be matched with the last rule. This may result in a low firewall throughput. This paper will identify the shadow rule based on the tree rule firewall.

Algorithm-2 explains the identification of shadow rules from generated tree rule firewall. The input to this algorithm is the root node of the firewall tree.

Algorithm-2 Find Shadow Rules

```
Input: Tree Rule Root Node (TRN)
Output: Shadow Rule (SR)
Step01:  If  TRN == LeafNode
Step02:      parent=getParent(TRN)
Step03:      childCount=getChildCount(parent)
Step04:      if(childCount > 1)
Step05:         add TRN into SR
Step06:      endif
Step07:  endif
Step08:  childs = getChildren(TRN)
Step09:  if (childs ≠ NULL)
Step10:     FindShadowRule(childs.next)
Step11:  endif
Step12:  return SR
```

Figure 4 shows the shadow rules. The shadow rules are identified based on the node (field) child count. For example, in figure 4, rule-3 and rule-5 are considered shadowed rules.

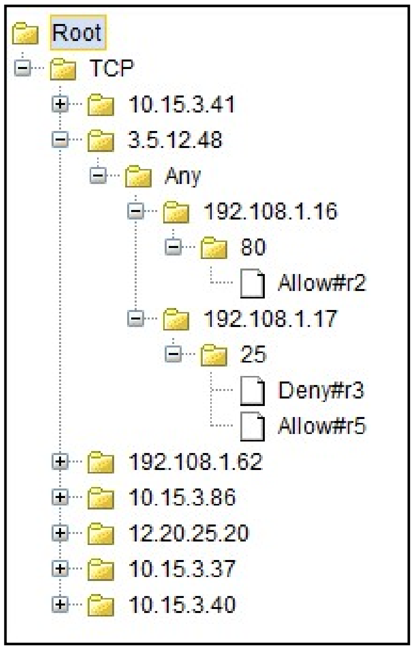


Figure 4 Shadow Rules

5. Implementation and Results

This section explains the implementation details and analyzes the performance of the suggested MTRFcloud. It is implemented with the Java platform, and the number of hosts and VMs are created and managed by cloudsim (cloud simulator). The system configuration is Windows 10, 64-bit OS, Intel Pentium 2.30 GHz and 4 GB of RAM. Figure 5 shows the sample Host and VM configuration used for this implementation.

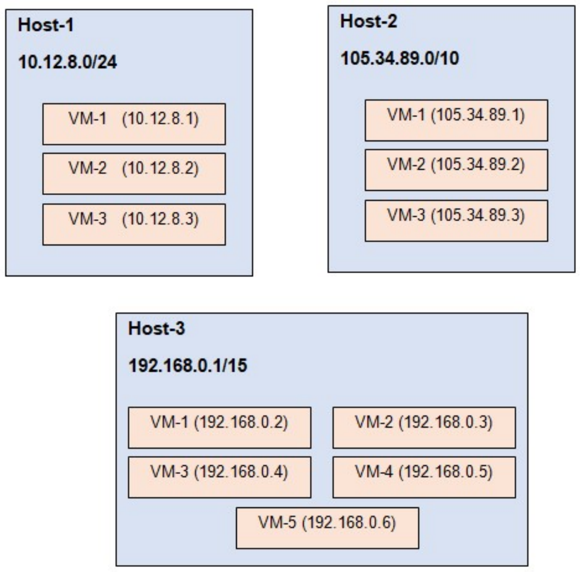


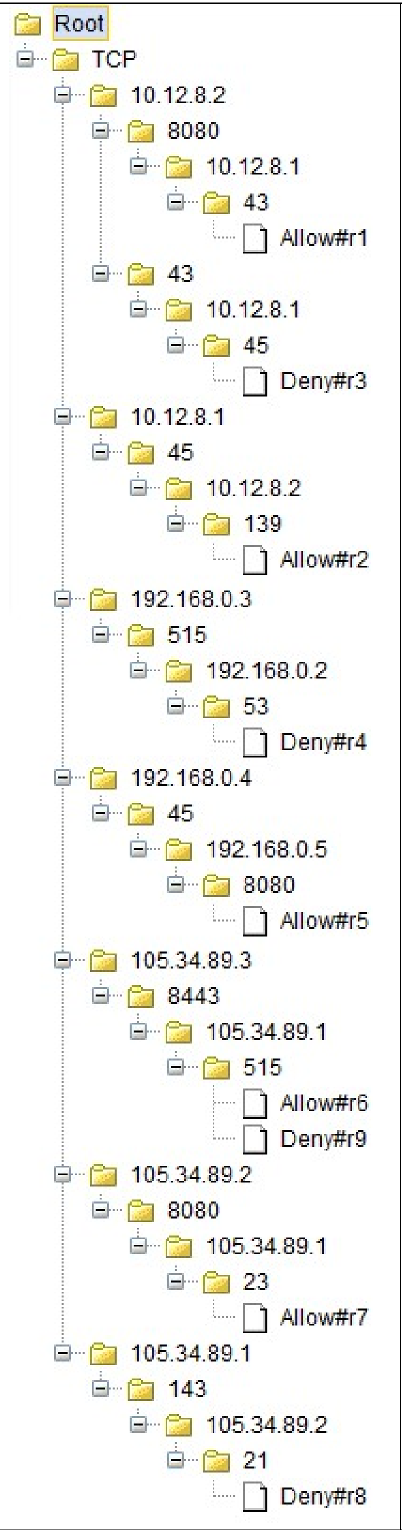
Figure 5 Sample Host and VM Configuration

Table 2 shows the sample rules collected from VMs.

Table 2 Sample Rules From VMs

Rule –Id	Protocol	SrcIP	SrcPort	DestIP	DestPort	Action
R1	TCP	10.12.8.2	8080	10.12.8.1	43	Allow
R2	TCP	10.12.8.1	45	10.12.8.2	139	Allow
R3	TCP	10.12.8.2	43	10.12.8.1	45	Deny
R4	TCP	192.168.0.3	515	192.168.0.2	53	Deny
R5	TCP	192.168.0.4	45	192.168.0.5	8080	Allow
R6	TCP	192.168.0.4	45	192.168.0.5	8080	Allow
R7	TCP	105.34.89.3	8443	105.34.89.1	515	Allow
R8	TCP	105.34.89.2	8080	105.34.89.1	23	Allow
R9	TCP	105.34.89.1	143	105.34.89.2	21	Deny
R10	TCP	105.34.89.3	8443	105.34.89.1	515	Deny

Figure 6 shows the Tree Rule and results after removing redundant and shadowing rules.



```
Root
  TCP
    10.12.8.2
      8080
        10.12.8.1
          43
            Allow#r1
      43
        10.12.8.1
          45
            Deny#r3
    10.12.8.1
      45
        10.12.8.2
          139
            Allow#r2
    192.168.0.3
      515
        192.168.0.2
          53
            Deny#r4
    192.168.0.4
      45
        192.168.0.5
          8080
            Allow#r5
    105.34.89.3
      8443
        105.34.89.1
          515
            Allow#r6
            Deny#r9
    105.34.89.2
      8080
        105.34.89.1
          23
            Allow#r7
    105.34.89.1
      143
        105.34.89.2
          21
            Deny#r8
```

After Removing Rule-6 (Redundant Rule) from Table-2

R1, TCP, 10.12.8.2, 8080, 10.12.8.1, 43, Allow
R2, TCP, 10.12.8.1, 45, 10.12.8.2, 139, Allow
R3, TCP, 10.12.8.2, 43 , 10.12.8.1, 45, Deny
R4, TCP, 192.168.0.3, 515, 192.168.0.2, 53, Deny
R5, TCP, 192.168.0.4, 45, 192.168.0.5, 8080, Allow
R6, TCP, 105.34.89.3, 8443, 105.34.89.1, 515, Allow
R7, TCP, 105.34.89.2, 8080, 105.34.89.1, 23, Allow
R8, TCP, 105.34.89.1, 143, 105.34.89.2, 21, Deny
R9, TCP, 105.34.89.3, 8443, 105.34.89.1, 515, Deny

Rule-6 and Rule-9 are the shadowed rules

Rules after removing redundant and shadow Rules from Table-2

R1, TCP, 10.12.8.2, 8080, 10.12.8.1, 43, Allow
R2, TCP, 10.12.8.1, 45, 10.12.8.2, 139, Allow
R3, TCP, 10.12.8.2, 43 , 10.12.8.1, 45, Deny
R4, TCP, 192.168.0.3, 515, 192.168.0.2, 53, Deny
R5, TCP, 192.168.0.4, 45, 192.168.0.5, 8080, Allow
R6, TCP, 105.34.89.2, 8080, 105.34.89.1, 23, Allow
R7, TCP, 105.34.89.1, 143, 105.34.89.2, 21, Deny

Figure 6 Tree Rules and Results

Table 3 shows the number of redundant and shadow rules

Table 3 Number of Redundant and Shadow Rule

Rules #	**Redundant Rules #**	**Shadow Rules #**
100	2	6
200	4	14
400	99	194
600	150	284
800	193	396
1000	240	482
1200	282	590

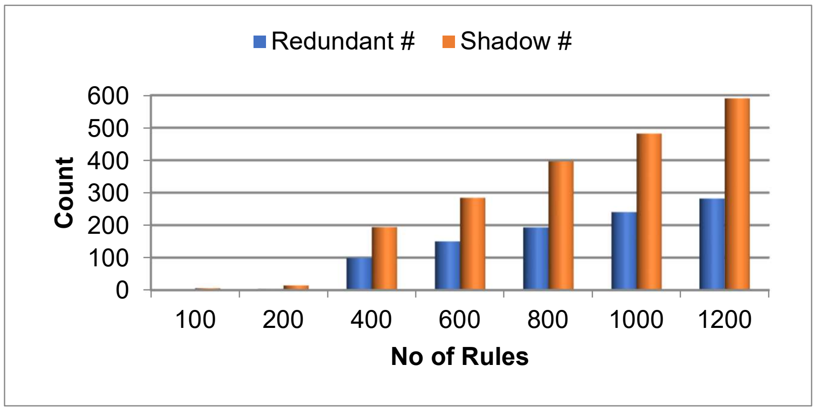


Figure 7 No of Rules Vs Redundant and Shadow Rules

Table 4 shows the time comparison for different numbers of rules.

Table 4 Time Comparison

Rules #	**Tree Rule Generation Time(ms)**	**Processing Time (ms)**
100	114	181
200	214	375
400	402	549
600	615	750
800	819	1080
1000	1019	1592
1200	1258	3932

Figure 8 shows the execution time of tree generation and processing time.

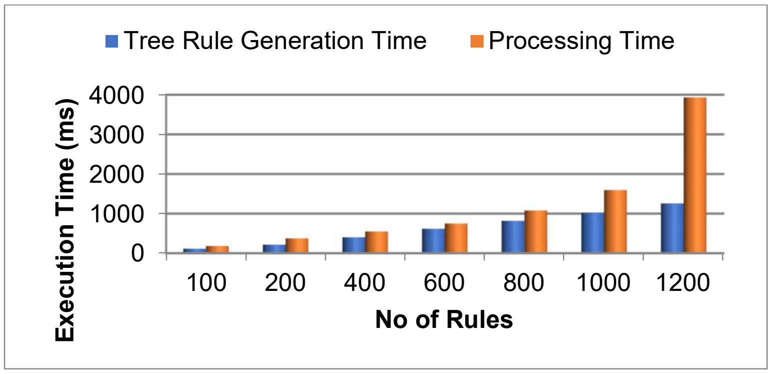


Figure 8 Execution Time for different numbers of rules

The total processing time of MTRFcloud is compared with an adaptive cross-domain firewall (ACD) [17] and a double decision tree (DDT) [18]. Figure 9 shows the total processing time comparison of different rules. From that results, the time percentage between ACD and proposed is 32.78% and DDT and proposed is 14.75%.

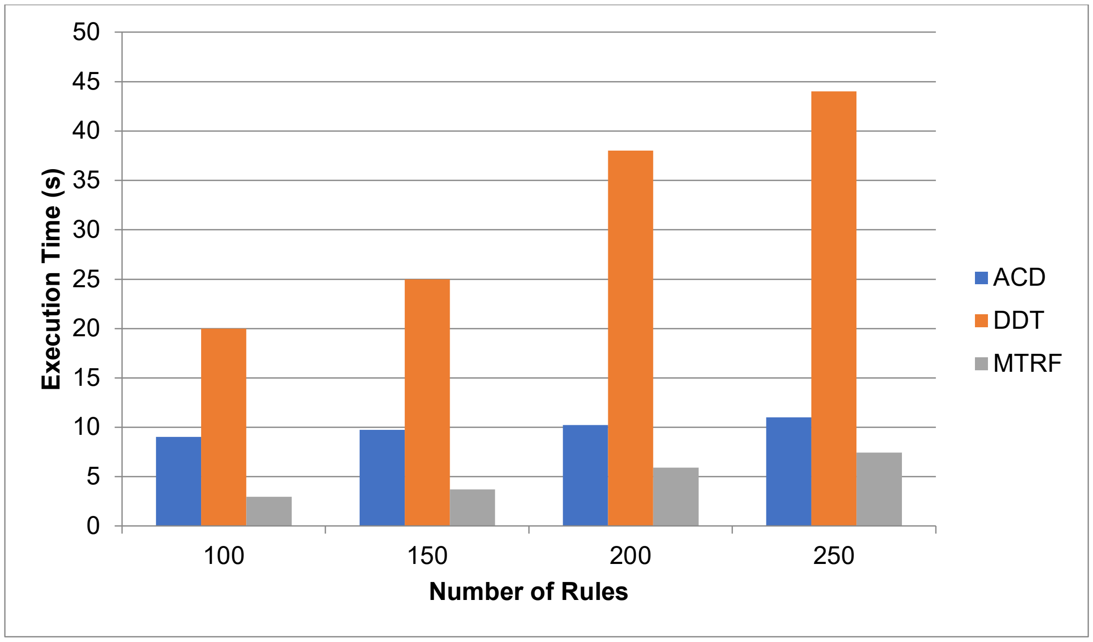


Figure 9 Total Processing Time Comparison

6. Conclusion and Future Work

The modified Tree-Rule firewall was suggested in this study to eliminate redundant and shadow rules. The MTRFcloud uses rules in a tree-based format, and the transmitting selection of incoming traffic based on tree policy will adhere to the tree formation to decide on the traffic sooner. MTRFcloud has been tested in a cloud setting and is better suited for a cloud environment. It was also observed to be quick to decide whether to forward packets. Nonetheless, the results of the experiments indicate that in a realistic setting, the processing time for tree rule generation and the identification of redundant and shadowing rules in a firewall are acceptable. From that results, the time percentage between ACD and proposed is

32.78% and DDT and proposed is 14.75%. Future works include the detection of intra and inter-firewall anomalies.

Declarations

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.The authors declare the following financial interests/personal relationships which may be considered as potential competing interests.

Conflicts of Interests

No Conflicts of Interest present.

Funding

No funding received from anywhere. This research work is under No Funding Zone.Authors and Co Authors have completely bared the research costs.

Availability of data and materials

Datasets are made by own.

Ethics approval and consent to participate

This paper can be participated in this journal with full consent of Authors and Co-Auhtors

Consent for publication

This paper can be published in this journal with full consent of Authors and Co-Auhtors

Competing interests

There are no Conflicts of Interests with any one

Funding

Authors and Coauthors have solely funded in research work

Authors' contributions

Coauthor have written paper

Acknowledgements

First of all I would like to thanks Almighty for all His kind blessings without which my work has been knock-off. Also I would like to give the credit to the people who remember me in their prayer for my achievements.

References

[1] Liu, A. X., Khakpour, A. R., Hulst, J. W., Ge, Z., Pei, D., & Wang, J. (2017). Firewall fingerprinting and denial of firewalling attacks. IEEE Transactions on information forensics and security, 12(7), 1699-1712.

[2] Cheminod, M., Durante, L., Seno, L., & Valenzano, A. (2021). An Algorithm for Security Policy Migration in Multiple Firewall Networks. In ITASEC (pp. 344-359).

[3] Jabal, A. A., Davari, M., Bertino, E., Makaya, C., Calo, S., Verma, D., et al. (2019). Methods and tools for policy analysis. ACM Computing Surveys (CSUR), 51(6), 1-35.

[4] Ullrich, J., Cropper, J., Frühwirt, P., & Weippl, E. (2016). The role and security of firewalls in cyber-physical cloud computing. EURASIP Journal on Information Security, 2016(1), 1-20

[5] Toumi, H., Fagroud, F. Z., Zakouni, A., & Talea, M. (2019). Implementing Hy-IDS, mobiles agents and virtual firewall to enhance the security in IaaS Cloud. Procedia Computer Science, 160, 819-824.

[6] Voronkov, A., Iwaya, L. H., Martucci, L. A., & Lindskog, S. (2017). Systematic literature review on usability of firewall configuration. ACM Computing Surveys (CSUR), 50(6), 1-35.

[7] He, X., Chomsiri, T., Nanda, P., & Tan, Z. (2014). Improving cloud network security using the Tree-Rule firewall. Future generation computer systems, 30, 116-126.

[8] Chomsiri, T., He, X., Nanda, P., & Tan, Z. (2016). Hybrid tree-rule firewall for high speed data transmission. IEEE transactions on cloud computing, 8(4), 1237-1249.

[9] Chomsiri, T., He, X., Nanda, P., & Tan, Z. (2014, September). A stateful mechanism for the tree-rule firewall. In 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications (pp. 122-129). IEEE.

[10] Suresh, N., & Bai, B. M. (2016). Predictive Modelling of Tree Rule Firewall for the Efficient Packet Filtering. International Journal of Computer Science and Information Security, 14(10), 189.

[11] Trabelsi, Z., Masud, M. M., & Ghoudi, K. (2015). Statistical dynamic splay tree filters towards multilevel firewall packet filtering enhancement. Computers & Security, 53, 109-131.

[12] Trabelsi, Z., Zeidan, S., Shuaib, K., & Salah, K. (2018). Improved session table architecture for denial of stateful firewall attacks. IEEE Access, 6, 35528-35543.

[13] Jekese, G., & Hwata, C., "Virtual Firewall Security on Virtual Machines in Cloud Environmen", International Journal of Scientific and Engineering Research, 6(2), 2015

[14] Dezhabad, N., & Sharifian, S. (2018). Learning-based dynamic scalable load-balanced firewall as a service in network function-virtualized cloud computing environments. The Journal of Supercomputing, 74, 3329-3358.

[15] Bagheri, S., & Shameli-Sendi, A. (2020). Dynamic firewall decomposition and composition in the cloud. IEEE Transactions on Information Forensics and Security, 15, 3526-3539.

[16] Praise, J. J., Raj, R. J. S., & Benifa, J. B. (2020). Development of Reinforcement Learning and Pattern Matching (RLPM) Based Firewall for Secured Cloud Infrastructure. Wireless Personal Communications, 115, 993-1018.

[17] Kadam, P. R., & Bhusari, V. K. (2014) Redundancy removal of rules with reordering them to increase the firewall optimization. International Journal of Research in Engineering and Technology, 3(10), 317-321.

[18] Lin, Z., & Yao, Z. (2022). Firewall Anomaly Detection Based on Double Decision Tree. Symmetry, 14(12), 2668.