# Efficient Firewall Designing using Ant Colony Optimization and Hierarchical Distribution

Tanveer Ahmed
University School of Information Technology
M.Tech CSE,GGSIPU, Dwarka
New Delhi, India
tanveerhmd88@gmail.com

Jyotsna Singh
University School of Information Technology
GGSIPU, Dwarka
New Delhi, India

*Abstract*— **A firewall is a security guard placed at the point of entry between a private network and the outside Internet such that all incoming and outgoing packets have to pass through it. The function of a firewall is to examine every incoming or outgoing packet. Here the paper aims to implement ACO (ant colony optimization) in the design process thereby allowing better filtering of the packets. The design of the firewall will be based on matching the rules and remembering the rules accessed by laying down pheromone. In practical situations multiple request for the same rule(same host and destination) arrives at the filtering router, if we are able to speed up the entire lookup process then a high access rate and throughput can be achieved. In other words, by comparing the packet's field values we first isolate the given rule in the rule set and then cache it in to speed up further lookup processes, also remembering the entire rule can increase the duration for which the rule will remain in the fast storage area. The idea is based on the fact that multiple requests for the same rule arrive at the filtering station. The rules here also contain a field indication the pheromone deposition value, whose rate of evaporation is governed by external factors like timers, environment variables, network congestion access frequency etc.**

***Keywords*— Network Security, Artificial Intelligence, Firewall .**

## I. Introduction

A firewall is a device or set of devices which allows or deny network transmissions based upon a set of rules and is frequently used to protect networks from unauthorized access while allowing legitimate transmissions to continue. A firewall is one of the most crucial part of the organization whether private or government. It is placed at the entry point between the internal and the external network, it filters out the data based on packet contents. A firewall can operate in at most 3 layers of the TCP/IP model.

**Circuit-Level Gateway:** They work at the session layer of the OSI model and monitor TCP handshaking between packets to determine whether a requested session is legitimate or not.

**Network layer:** These firewalls operate at a relatively low level of the TCP/IP protocol stack, not allowing packets to pass through the firewall unless a match is found.

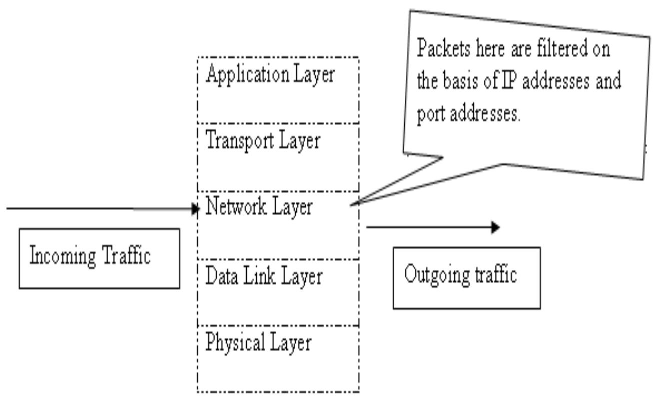
**Application-layer filter:** Application-layer firewalls work on the application level of the TCP/IP stack and intercept all packets traveling to or from an application.

The firewall at the top most layer i.e. application layer deals with a lot of traffic. Such traffic contain huge amount of information whether authentic or unauthentic, which sometimes can become real tricky to detect, so this part of the system is the one where the real risk lies. The content of such illegal traffic can therefore pass through a loophole and can cause serious damage. If such filtering can be assisted with the help of other mechanism then the overall security can be increased very effectively. Here in this paper we present a mechanism to deal with network layer packet filtering using ant colony optimization. In the paper we have used the terminology of rule set, which comprises of all the fields of an IP Packet that helps in correctly invoking the proper action against filtering of the packet. Sample rule set is shown below.

**Table 1**. Sample Rule Set

| Source IP | Destination IP | Source Port | Destination Port |
|---|---|---|---|
| 192.168.1.13 | 192.168.1.154 | 3306 | 3306 |

Packet filtering takes place at the network level, the following stack show the flows of IP Datagrams at the network layer.



## II. Ant Colony Optimization

Ant colony optimization can be considered to be flock of ants trying to find their route to the food and back to the colony, in the process first an ant agent traverses the path based on the likelihood of finding food, after that it lays down the pheromone in order to remember the path traveled. In simple terms a scout here interacts with the external system, based on the observations of the scout future operations are executed. Now a days computer science is trying to map every single opportunity to map the real world into computer model which would help us solve our problem in a rather optimistic way by the process of selection and evolution. Flocks of ants try to find their route to the source of food, in computer world there are rules which direct these kinds of searches. Since flocking arise from such rules therefore the possibility that an unknown rule can be accepted by the agents is unlikely. These rules comprises of the rule set. The ant agent in this case first moves through the rule set and lays down pheromone. This pheromone information will direct the search of the future ants. Furthermore, an ACO algorithm uses the mechanism of trail evaporation. Trail evaporation decreases all trail values over time, in order to avoid infinite accumulation of trail for a particular component. To accomplish this daemon processes can be used, these daemon procedures can be used to update of global information responsible for biasing the search process. The algorithm for ACO can be summarized below
Initialize parameters which determines the pheromone trail

```
While (solution found)
Do
Generate Solutions
Apply Local Search
Update Pheromone Trail
End
```

## III. Related work

The simplest approach to designing the firewall would be to match the rules sequentially, the resulting complexity of the system would be O(n), In [1], a firewall was proposed which matches the k fields in the rule set, thought the complexity of the system increased to $O(n^2)$. In [3] a binary search based rule filter is proposed the complexity of the system gradually decreased to O(log n), but with an additional overhead of maintaining the rule set in sorted form. In [4], an efficient filtering algorithm was proposed whose complexity is O(log n) but again the filtering rule set has to be present in sorted manner.

## IV. Proposed Firewall Model

Packet filters are based on the fact that multiple attributes contribute is the decision of whether dropping or accepting the packet at that moment. Routers operate at network layer providing functionality for such filtering. The attributes that contribute for the filtering process are Source IP address, Destination IP address, Source Port Number, Destination Port Number. The firewall compares these particular fields of the incoming packets and makes it's decision whether to accept or drop the packet.

Mathematically, a firewall is set of rules which are defined as:
<Action>→<Decision>
where a rule is defined as
$R(i)=a(1) \cup a(2) \cup a(3) \cup a(4) \cup a(5)\ldots\ldots\ldots\ldots a(n)$
$R(i) = i^{th}$ rule
$a(i) = i^{th}$ attribute
Also, decision $\in$ {accept, discard}

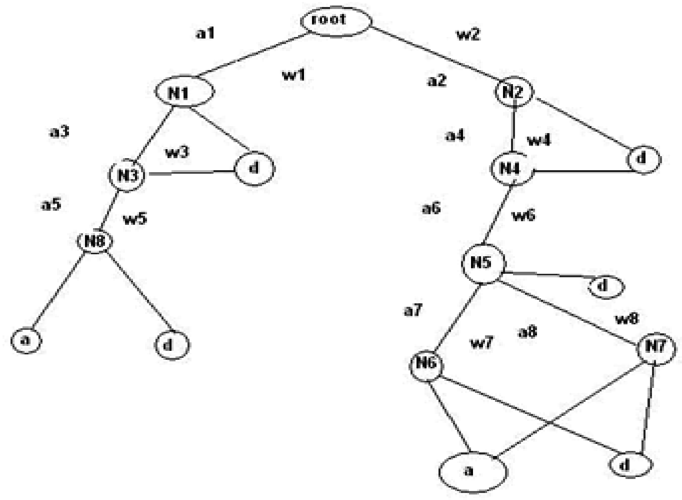


**Fig 2.** Decision Tree comprising of rules.

In the algorithm proposed first the IP address is converted to its numerical format. Sample format is shown in the table below
Table 2. Numerical Format Conversion

| IP Address | Numeric equivalent |
|---|---|
| 192.168.1.13 | 192168001013 |
| 10.10.0.5 | 010010000005 |

ACO based firewall filters the rules present in the rule set. The rule checker first checks the rules in the rule set then based on the various detection units in the firewall, packets are filtered. First the filtering starts with the source and destination IP address then the algorithm checks for the source and destination port numbers in the port checking unit. The combined result of the IP and Port checker together constitute the rule being selected and packet being filtered.

## A. *Packet Filtering*

In this following example we have used only 3 rule set although the number of rule sets can be increased as per requirements. Though the number of rule sets can vary from organization to organization but we strongly recommend to limit the rule sets to 3-5 as more rule set means more tuples to scan for finding a particular rule. The following example shows the rule and their access for 3 rule sets. Master rule set here is one contains all the rule sets. Sample rules of the master rule set can be summarized in the following tables.

**Table 3.** Sample Master Rule Set

| Source IP | Destination IP | Source Port | Dest Port | Phval | Decision |
|---|---|---|---|---|---|
| 192168001013 | 192168010010 | 80 | 80 | 0 | Accept |
| 010010000001 | 010010000005 | * | 3306 | 0 | Accept |
| 194027251021 | 172016112100 | 1169 | 6859 | 0 | Deny |
| 202154125203 | 203201145165 | 80 | 28 | 0 | Deny |
| 154155158201 | 200012057125 | * | * | 0 | Deny |
| 198157045023 | 198001157124 | 3306 | * | 0 | Accept |

After some of the rules have been accessed from a particular rule, they are propagated to the next level, the propagated rules are summarized in the following table.

**Table 4.** Rule Set in the next hierarchical Rule Set

| Source IP | Destination IP | Source Port | Dest Port | Phval | Decision |
|---|---|---|---|---|---|
| 192168001013 | 192168010010 | 80 | 80 | 1 | Accept |
| 194027251021 | 172016112100 | 1169 | 6859 | 1 | Deny |
| 154155158201 | 200012057125 | * | * | 1 | Deny |

Again if the same rule is accessed then the rule in the next higher hierarchy is accessed not the master rule set.

**Table 5.** Rule Set in the next hierarchical Rule Set

| Source IP | Destination IP | Source Port | Dest Port | Phval | Decision |
|---|---|---|---|---|---|
| 192168001013 | 192168010010 | 80 | 80 | 2 | Accept |
| 154155158201 | 200012057125 | * | * | 2 | Deny |

If the rules in the 3rd rule set is accessed the amount of pheromone deposition is increased but only to a certain level due to the environment constraints. In table 5 it is considered that the same rules are accessed 4 more times. The max pheromone deposition in this case is limited to 6.(though it can be changed depending on the requirements) only.

**Table 6.** Rules accessed multiple times

| Source IP | Destination IP | Source Port | Dest Port | Phval | Decision |
|---|---|---|---|---|---|
| 192168001013 | 192168010010 | 80 | 80 | 6 | Accept |
| 154155158201 | 200012057125 | * | * | 6 | Deny |

The importance here is that as we go up in the hierarchy of the tables the access time is considerably reduced. As soon as a rule is identified in a particular rule set at any level,, it is propagated up in the hierarchy thereby the access time of the rule is decreased.

The following figure shows the hierarchy of the rule sets used in the methodology.

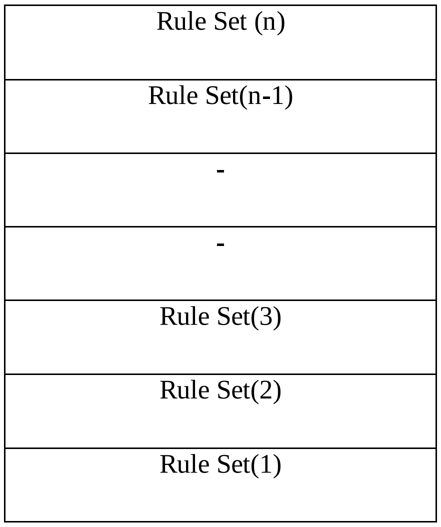


Figure 3. **Hierarchy of Rule Sets**

It can be thought of an analogous to the memory divided into n level. The rule accessed at any level is propagated 1 level up in the hierarchy. The highest level is fastest as it contains only a limited number of rules and is present in the fast access storage (cache memory). A particular level is governed by an associated ph value. For e.g. for rule set at level n the phval can be n-1, for that at level n-1 it is n-2, for level 1 it is always zero as this level is the one where all the rule for filtering are present i.e. it is the master rule set (analogous to the hard disk in a cache memory scheme).

## B. *Rule Matching*

As shown in the above tables the rules in the rule set consist of various fields which are Source IP address, Destination IP address, Source Port, Destination Port. All the fields are checked in their separate units the units operate in parallel; these units in turn determine the likeliness of the rule to be selected.

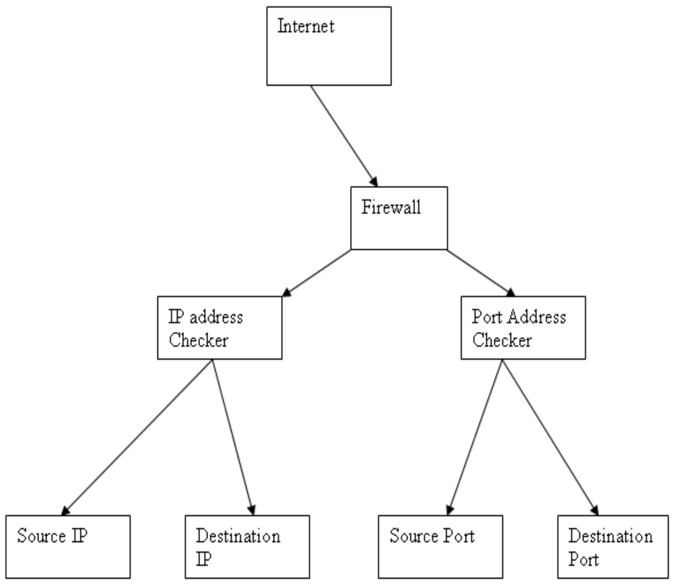


**Figure 4.** Scenario for parallel checking units

The behavior here loosely resembles the likeliness concept of ACO. The following flowchart shows the functionality of this behavior. Likeliness is probability of a rule found most suitable for the particular Datagram. The likeliness here is determined by the variable L(i,j), where L(i) means the likeliness of the i th rule of j th rule set. For every unit if the field of the incoming packet matches that of the rule in the rule set then the likeliness of that rule can be thought to increase. For every unit the increment of .25 in the likeness takes place only if the rule`s field matches that of datagram's field. Only the rule/ rules whose likeliness is equal to 1 are allowed to take part in the selection process. Since there are 4 fields therefore, value of 1 is considered appropriate. For packets with likeliness value less than 1, are dropped by the firewall. In simple terms as soon the packet has passed through the IP checker the overall likeliness of the rule is increased by a factor of ½ only both the destination and source IP address matches. Similarly likeliness is increased to 1 if the port checker also conforms the presence of a rule for the particular datagram.

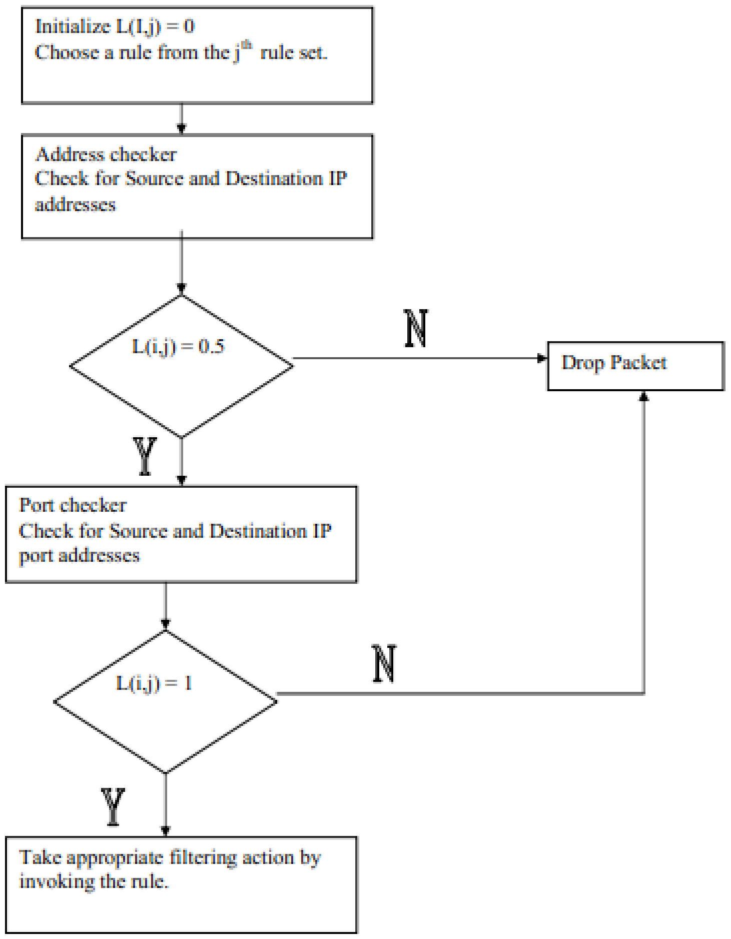

Initialize L(I,j) = 0
Choose a rule from the j th rule set.
Address checker
Check for Source and Destination IP addresses
L(i,j) = 0.5
N
Drop Packet
Y
Port checker
Check for Source and Destination IP port addresses
L(i,j) = 1
N
Y
Take appropriate filtering action by invoking the rule.

**Figure 5**. Flowchart for likeliness calculation

## V. Algorithm for packet filtering

In the following algorithm we have considered only three rule sets - ant1, ant2, ant3 (low to high manner). To simulate evaporation of pheromone, timers are set to t1 and t2 seconds for ant3 and ant2, no timer was utilized for ant1 as in this case it is the master rule set and will be accessed only if the rule are not found in any of the higher rule sets. The time periods of the timers here are flexible and can be set on the fly. As soon as the phval of the rules is zero that is as soon as the pheromone deposition of the particular rule has been destroyed that rule is removed from the higher rule but it is still present in the lower rule set. Here pseudo code of the algorithm is shown, the algorithm first starts with converting the IP address to it`s numeric format then comparing all the respective fields with that of the rules in the rule set. If a match if found then proper filtering is done else the packet is dropped. Here, the packet could be dropped or even logged for further analysis.

```
Start_timer()
{
 /*start filtering the rule with the rule set ant3*/
 access the ant3 to find the rule
 if (found)
 filter packet
 return
else if
/*if the rule is not found then access ant2*/
access the ant2 to find the rule
if (found)
add rule to ant3
increase PHval by 1
filter packet
return

/* access the table ant1 that is the master table which contains all the rules */

else if
access the ant1 to find the rule
if (found)
add rule to ant2
increase PHval by 1
filter packet
return
/*if the rules are not found then drop the packet*/

 else
 add the rule to the log file for analysis
 return

/*here the timer is started to update the pheromone values  first the table ant3 updated and then the table ant2 if the pheromone has completely evaporated then the rule is removed from the rule set.*/

 if (timer_time>t1)
 {
 decrease PHval in ant3 by 1
if(phval==0)
remove rule
 }
for each rule removed from ant3
do{
 if (timer_time>t2)
 {
 decrease PHval in ant2 by 1
if(phval==0)
remove rule
 }
}
}
```

## VI. Summary

In designing of the algorithm it can be seen that the complexity of the system in best case is π(n) in the worst case it is O(n).The following table summarizes the results

**Table 7.** Comparision with ACO-PF[4]

| Name | ACO-PF | Firewall Proposed |
|---|---|---|
| Time Complexity | O(ln n) | π(n) |
| Space Complexity | O(n) | O(n*j) |
| Maintenance | O(ln n + n) | O(1) |
| Support for Cache | No | Yes |

Here j is the number of rule sets utilized by the system.

## *References*

[1]. E. Bonabeau, M. Dorigo and G. Theraulaz. Swarm Intelligence: from natural to artificial

[2]. T.V. Lakshman and D. Stidiaslis, High speed policy-based packer forwarding using efficient multi- dimensional range matching *Proceedings of SIGCOMM'98* (1998).

[3]. M. Waldvogel, G. Varghese, J. Turner and B. Plattner, Scalable high speed ip routing lookups.

[4]. N.K. Sreelaja, G.A. Vijayalakshmi PaiC:\Users\GATEXPERTS\AppData\Local\Temp\desktop\S156849461000075X.htm - fn1 Ant Colony Optimization based approach for efficient packet filtering in firewall Applied Soft Computing Volume 10, Issue 4, September 2010, Pages 1222-1236.

[5]. A. Hari, S. Suri, G.M. Parulkar, Detecting and resolving packet filter conflicts, in: Proceedings of IEEE INFOCOM, 2000, pp. 1203–1212.

[6]. Mohamed G. Gouda, Alex X. Liu, Structured firewall design, Computer Networks: The International Journal of Computer and Telecommunications Networking, Volume 51 Issue 4, March 2007