"Heap Based Rule Set for Packet Filters"

Mukesh kumar yadav[1], Shiv Kumar[2]
Arya college of Engineering and Information technology, Jaipur, Rajasthan
mukeshyadav.doc@gmail.com , shiv@aryacollege.in

ABSTRACT

Firewalls play the crucial role in the computer security. The overall performance of a firewall depends upon the Time required in rule matching. As the number of the entries increase in the list, the speed of firewall becomes slow. Time required in rule matching is directly proportion to the size of the list. To improve the overall performance of a List based firewall, new multi level Heap Based rule set for the packet filter is presented. In this scheme system maintains two different rule set for incoming & outgoing packets, so average rule set size is reduced to n/2. Rule set for incoming and outgoing packet contains only three parameters to evaluate that packet's action type (i.e. – either forward or discard the packet). Using the multilevel heaping the performance of a packet filter improves up to level O ($n_1+n_2+n_3$). This scheme also removes the anomalies of the rule set.

Keywords**: SIP, SPORT, DIP, DPORT, Heap based rule set**

1. Introduction to Packet Filters

Speed of the packet filter primarily depends upon the searching and matching time of the rule in the rule set. Therefore the focus is on reduction in searching & matching time in the rule set using the new heap based rule set. Normally home users use the optimistic policy and an enterprise uses the pessimistic policy [1].

1.1 Packet Filtering by the Packet Filter

Packet filters take the decision for packet forwarding on the basis of policies defined by the rule set. The rule set is stored in a list, in the form of a table that contain values for the following attributes (<action type>, < source IP address (SIP) >, < source port (SPORT) >, < destination IP address (DIP) >, < destination port (DPORT) >) as shown in the table 1. Suppose a packet arrives on a packet filter, its SIP is 190.15.62.1, SPORT 8080, DIP is 195.20.10.1, and DPORT is 8080. After receiving the packet the packet filter ensures the policies defined by the rule set, by matching the SIP , SPORT, DIP, DPORT with its own rules defined in the rule set table containing 'n' entries. Packet filter will match these values with the first entry in list, if match do not occur then, packet filter will match these values with next entry. This process will be repeated until a match is found or end of list is encountered. This process will take at most n comparisons for a successful or unsuccessful match.

ACTION	SIP	SPORT	DIP	DPORT
Block	192.10.10.1	6060	192.20.20.2	6060
Block	192.10.10.1	6060	192.20.20.2	6080
Block	192.10.10.1	6060	192.20.20.2	8080
Block	25.15.6.1	8080	195.20.10.1	6080
Accept	*	*	*	*

TABLE 1: A sample of rule set used by packet filter to identify that packet should be forwarded or discarded. Comment field is only for explanation and may not be present in real rule set table[1][2][3].

In worst case, searching will take time of order O (n) to take the default action and each entry is composed of four variables.

Commonly ssimulation study is done monitor the performance[4] . Common scheme to improve the performance is Dynamic rule ordering [4] and Traffic-Aware Optimization [5] , but they are not enough to increase the performance over the few percent , because both are working on local search heuristics.

2. Heap based rule set to improve the searching performance of a packet filter

Today's Packet filter uses the List Based Rule set for defining & searching the rules for packet filter. Performance of packet filter decreases as the size of rule set grows. To overcome this problem a new way of defining the rule set is presented here. i.e. – Heap Based Rule Set.

2.1 Proposed improvement in List based Packet Filter

Whole matching process takes sequential approach for rule matching, resulting in the time complexity of order O (n). Here 'n' depicts the number of entries existing in the rule set. The searching time can be reduced by dividing the single list into the two different rules sets one for incoming packets and second for outgoing packets. If list based rule set contain "N" entries then this scheme creates two rule sets with "N/2" entries on an average. System can determine in a single match that, packet is either outgoing or incoming.

Sample Rule set for Incoming Packets

SIP	SPORT	DPORT
189.78.9.1	6060	8080

Table 2: sample rule for incoming packets after creating two different rules set for incoming and outgoing packets.

Sample rule set for Outgoing Packets

DIP	DPORT	SPORT
189.78.9.1	6060	8080

Table 3: sample rule for outgoing packets after creating two different rules set for incoming and outgoing packets.

This searching time can be further reduced if system divides the whole list into small subsets according to some common features, like same SIP, SPORT or same DIP . If the packet's information does not matching with the common properties of the subgroup, then the whole sub-group is skipped, and next subgroup is matched. This idea is elaborated with the help of multilevel heaping. So performance is again improved by the multilevel heaping scheme

2.2 Introduction to Multilevel Heaping

Basic idea behind this is that a heap file is used to store the information about the records position, where they are stored in file. For this heap contains the record name and key associated with the file, generally key denotes the position of the record[7].

2.3 Implementation of the Multilevel Heaping for the Rule set

This is an optimistic approach. In this approach list based rule set is converted into the multilevel heap. Idea behind this is that unwanted packets (i.e. the packets going to or coming from an unauthorized / malicious end point are required to be discarded) are sent from the intruder's site to different hosts to breach the security. These packets may have the common source IP address or destination IP, this is the criteria for grouping the rule set for the primary heap.

ACTION	SIP	SPORT	DIP	DPORT
Block	192.10.10.1	6060	192.20.20.2	6060
Block	192.10.10.1	6060	192.20.20.2	6080
Block	192.10.10.1	6060	195.25.25.2	6080
. .	. .	. .	. .	. .
Block	190.15.62.1	8080	195.20.10.1	8080

TABLE 4: A sample list based rule set used by packet filter to identify packet should be discarded.

2.4 Converting the sequential rule set into the heaped rule set

For the large rule set after the conversion performance of searching is likely to be improved considerably. This chapter deals with the conversion process of list based rule set into the heap based rule set and searching process of entries in heap based rule set. A sample conversion of list based rule set table 4 is shown in figure 1. Here the algorithms presented for conversion of list based rule set into heap based rule set, searching record of packet into heap based rule set.

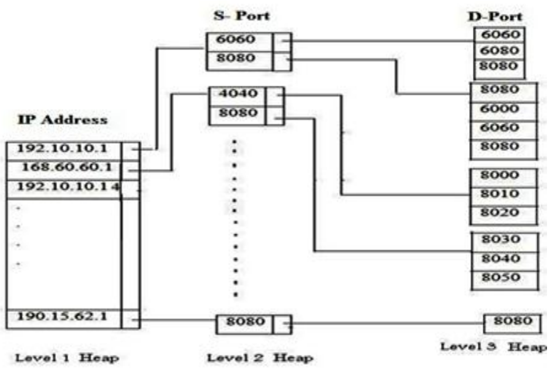

Fig 1: A sample multilevel heaped rule set for the Packet filter to improve the search efficiency of the packet filter. the basic process conversion of list based rule set into heap based rule set for packets rule set().

2.5 Algorithms for heap based rule set

1. Algorithm for converting list based rule set into heap based rule set

Input: SIP: = source IP address, SPORT: = source port number, DIP: = destination IP address , DPORT:= destination port number

```
START
1.  if SIP!= system own IP address, then // add rule in incoming packet's rule set R1
2.      if SIP found in first level heap of R1 then,
3.          Search SPORT in second level Heap with name "R1.SIP", if found then
4.              Search DPORT in third level Heap with name "R1.SIP.SPORT", if found then
5.                  Declare : "entry for SIP, SPORT , DPORT already in R1(incoming packet's rule set)"
6.              else add DPORT entry in "R1.SIP.SPORT"
7.              endif
8.          else
                i.   Add SPORT in "R1.SIP"
                ii.  Create third level heap with name "R1.SIP.SPORT" and add DPORT in this
9.      endif
10.     else
                i.   Add SIP in R1
                ii.  Add SPORT in "R1.SIP"
                iii. Create third level heap with name "R1.SIP.SPORT" and add DPORT in this
11. endif
12. Else // add new rule in outgoing packet's rule set R2
13. if DIP found in first level heap of R2 then,
14.     Search DPORT in second level Heap with name "R2.DIP", if found then
15.         Search SPORT in third level Heap with name "R2.DIP.DPORT", if found then
```

```
16.                 Declare : "entry
    for  DIP, DPORT , DPORT already in
    R2(Outgoing  packet's rule set)"
17.                 else   add SPORT
    entry in "R2.DIP.DPORT"
18.               endif
19.           else
               i.  Add DPORT in
                   "R2.DIP"
               ii. Create third level
                   heap with name
                   "R2.DIP.DPORT"
                   and add SPORT in
                   this
20.    endif
21.       else
               i.  Add DIP in R1
               ii. CREATE  second
                   level heap with name
                   "R2.DIP"Add
                   DPORT in "R2.DIP"
               iii. Create third level
                   heap with name
                   "R1.DIP.DPORT"
                   and add SPORT in
                   this
22.   endif
23.  STOP
```

Algorithm for searching in heap based rule set:

2. Algorithm for Search in Heap based rule set

```
Input: SIP: = source IP address, SPORT:=
source port number, DIP:= destination IP
address , DPORT:= destination port number
START
1 .      if SIP!= system own IP address, then //
Search  rule in incoming packet's rule set R1
2.       if SIP found in first level heap of R1
         then,
3.            Search SPORT in second level
         Heap with name "R1.SIP", if found then
4.                 Search DPORT in third
         level Heap with name "R1.SIP.SPORT", if
         found then
5.                      Declare : "entry for
         SIP, SPORT , DPORT found in
         R1(incoming packet's rule set)"
6.                    else  Declare: " Rule
         not found for incoming packet at heap level
         III"
7.                   endif
8.              else   Declare: " Rule not
         found for incoming packet at heap level II"
9.            endif
10.        else  Declare: " Rule not found for
         incoming packet at heap level I"
11.  endif
12.  Else  // Search  new rule in outgoing
     packet's rule set R2
13.   if DIP found in first level heap of R2 then,
14.           Search DPORT in second level
     Heap with name "R2.DIP", if found then
15.                Search SPORT in third
     level Heap with name "R2.DIP.DPORT", if
     found then
16.                     Declare : "entry for
     DIP, DPORT , DPORT Found in
     R2(Outgoing  packet's rule set)"
17.                   else  Declare : "Rule
     not found for outgoing  packet at heap level
     III"
18.                  endif
```

```
19.        else  Declare: “Rule not found
    for outgoing  packet at heap level II”
20.      endif
21.       else   Declare: “Rule not found for
    outgoing  packet at heap level II”
22.  endif
23.  STOP
```

3 Performance analysis

The generic and induction based analysis for rule sets are presented in following section.

3.1 Performance analysis of List based rule set

Suppose that a packet is to be blocked with source IP address 127.5.90.214, source port number 8080 and destination IP address 127.25.0.133 and destination port is 6000. To define rule set of packet filter as quadruple SIP; SPORT; DIP; DPORT; where quadruple is defined as SIP=127.5.90.214; SPORT= 8080; DIP=127.25.0.133; DPORT= 6000;

. Then rule is defined as follows (shown in table 5)

Rule No.	SIP	SPORT	DIP	DPORT
R1	127.5.90.214	8080	127.25.0.133	6000

Table 5: defines the rule to block arriving packet from SIP=127.5.90.214; SPORT= 8080; DIP=127.25.0.133; DPORT= 6000; for a packet filter.

Now suppose that attacker changes his source port to 5000 to attack on the same application on the target application of destination. If system wants to prevent this kind of attack, the system has to add one more rule in the packet filters rule set with quadruple SIP=127.5.90.214; SPORT= 5000; DIP=127.25.0.133; DPORT= 6000; Now one additional entry for the same source . Same scenario occurs when attacker does not change either his source IP or source port number to attack on the same application on the target application of destination; rather he / she attacks on new application of the target with destination port say 6060. To prevent this kind of attack, add one more rule in the packet filters rule set with quadruple SIP=127.5.90.214; SPORT= 5000; DIP=127.25.0.133; DPORT= 6060; Now one additional entry for the same source as shown in table 6.

Rule No.	SIP	SPORT	DIP	DPORT
R1	127.5.90.214	8080	127.25.0.133	6000
R2	127.5.90.214	5000	127.25.0.133	6000
R3	127.5.90.214	5000	127.25.0.133	6060

Table 6: To prevent attack from same source IP and same port number but on different application, system defines the rule to block arriving packet from SIP=127.5.90.214; SPORT=5000; DIP= 127.25.0.133; DPORT= 6060; for a packet filter.

So the above process will gradually increase the number of entries in rule set by only changing the port number (either source or destination port). The system will see multiple entries in the rule set for the same Source IP (SIP) in the list based rule set (Source IP redundancy in the list based rule set). So system can conclude that time complexity of a packet filter depends on the number of entries in rule set

$$T(n) = C^* (n_{SIP} * n_{SPORT} * n_{DIP} * n_{DPORT}) \qquad (1)$$

Where C is a Constant value , n_{SIP} is the number of source IP Addresses blocked ; n_{SPORT} is the average number of source ports associated with each SIP to be blocked.; n_{DIP} is the average number destination IP Addresses blocked with each SIP; n_{DPORT} is the average number of destination ports associated with each SIP to be blocked

3.2 Performance analysis of Heap based rule set

This approach contains two different rule sets R_{IN} and R_{OUT} for incoming packets & outgoing packets. On an average each rule set R_{IN} (for incoming packets) and R_{OUT} (outgoing packets) contain "n/2" entries only. Again each rule set R_{IN} and R_{OUT} is converted in heap based rule set to further improve the performance. System can decide in a single comparison the either rule R_{IN} will be used or R_{OUT} will be used , for this system will compare packets source IP address with own address. If both are equal the packet is outgoing packet else packet is incoming packet.

Performance analysis for R_{IN} rule set:

To evaluate the performance of heap based rule set R_{IN}, This heap based rule set contain first level heap file contain only source IP address and do not contain any duplicate entries or any other level of heaping (in heap file). So to search any entry in first level heap will take at most O (n_1) time for a successful or an unsuccessful search, where n1 is the number of entries in first level heap.

Same as above second level heap associated with specific source port SPORT address, it contains only source port entries associated with that SIP address. Second level heap do not contain duplicate entries for SPORT. If second level heap contain n_2 entries, then any search for source port entry takes at most O (n_2) time. This second level heap is checked only if the SIP address entry was found in first level heap.

Third level heap file contain only destination port DPORT address. So to search any entry in third level heap will take at most O(n_3) time for successful or unsuccessful search ,where n_3 is the No. of entries in third level heap. This third level heap is checked only if the source port entry is found in second level heap.

So for any search total time taken by heap based rule set is

Case 1:

$$T(n) = O(n_1+n_2+n_3) \qquad (2)$$

For a worst case successful search.

Case2:

$$T(n) = O(n_1+n_2+n_3) \qquad (3)$$

For an unsuccessful search, in which SIP, SPORT was matched at heap level I, and heap level II.

Case3:

$$T(n) = O(n_1+n_2) \qquad (4)$$

For any unsuccessful search, but source IP SIP was matched during search in first level heap.

Case4:

$$T(n) = O(n_1) \qquad (5)$$

For any unsuccessful search, which source IP SIP was not matched during search in first level heap. Performance analysis of heap based rule set for R_{OUT} is similar to the $R_{in.}$

Advantages of the heap based approach over the list based & indexed based rule set

1. There are no duplicate entries for same rule, this removes the redundancy anomaly [6].
2. Uses the optimistic approach for packet filtering i.e. – discard only those packets whose entries are found in the rule set i.e. there is no shadow anomaly [6].
3. Heap based rule set can be optimized by the dynamic rule set optimizer on the basis of local search heuristic. But Index based rule set can not be optimized on this criteria.
4. It breaks the rule set in small sub parts, which reduces the matching time [7].
5. Easy to implement & configure [7].

4 EXPERIMENTAL RESULTS

Results are obtained using simulation, and tested with data set collected from own firewall. To obtain the experimental results two simulator programs are built in "C- language" and random input are applied on the simulator. For complexity analysis the step count method is used to find complexity. Both the rule sets checked for same input set..

Experimental Results for list based and heap based rule set

Experimental Complexity for List Based Rule Set and Heap based rule set is as follows

No. of rules	**C_L**	**C_H**	**% improvement in performance**
1	21	11	47.61905
5	57	15	73.68421
10	102	21	79.41176
15	147	30	79.59184
20	192	48	75
25	237	55	76.79325
30	282	70	75.1773
35	327	78	76.14679
40	372	88	76.34409
45	417	98	76.4988
50	462	108	76.62338
100	909	208	77.11771
200	1822	418	77.05818
Average performance improvement in %			74.38972

Table 7: Shows the complexity of list based rule set C_L , Complexity of heap based rule set C_H and % improvement in performance (100-(C_H/C_L*100)) (i.e. - total steps taken to perform action) .

Performance/ Comparison Graphs for List Based and Heap Based Rule set

Performance graph figure 2 shows the comparative performance in terms of search comparisons required to find the entry of rule in the rule set.

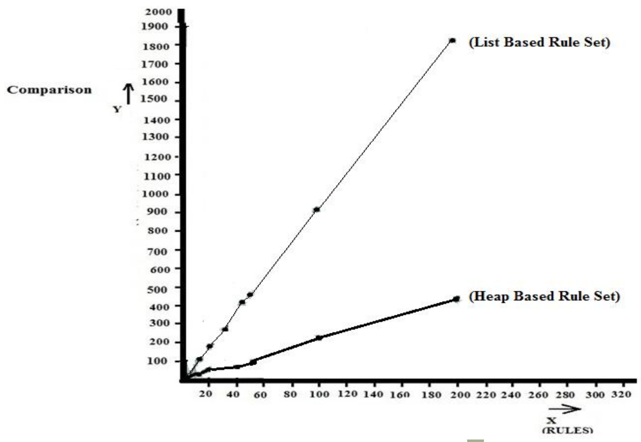

Figure 2 : Graph shows the number of comparisons required in List Based Rule Set & Heap Based Rule set (worst case).

6 CONCLUSION

Heap based rule set takes less or equal number of steps for searching as the size of rule set is increasing, and especially in case when there are multiple copies of rules having the same source IP address. One more advantage of Heap based rule set is that there are no anomalies in the rule set.

Searching in Heap based takes time of order T (n) =O (n_1+n_2+n_3), where n_1 is number of entries in first level heap file, n_2 is number of entries in second level heap file associated with source IP address stored in first level heap, n_3 is number of entries in third level heap file associated with Source Port address stored in second level heap. Average percentage improvement (observed) in the presented work is 74.39 %

7 REFERENCES

[1]. William Stallings, “Cryptography and Network Security”, ISBN 81-7758-774-9, Pearson Education, Inc Pvt. Ltd. 2006.

[2]. Christoph L. Schuba and Eugene H. Spafford , “A Reference Model for Firewall Technology” Proceedings of the 39th Annual

Simulation Symposium. (ANSS'06) –page no - 1213-1222 IEEE2006.

[3]. Tarek Abbes, Adel Bouhoula, Michale Russinowith, " An Inference system for detecting firewall filtering anomalies ", SAC 08 March 16-20, (page no-2122-2128) ACM9781-59593-753-7/08/2003.

[4]. Subrata Acharya, Jia Wang, Zihui Ge, Taieb Znati, "Simulation Study of Firewalls to Aid Improved Proceedings of the 39th Annual Simulation Symposium (ANSS'06) 0-7695-2559-8/06 - 2006 IEEE.

[5]. Hazem Hamed ,Ehab Al-Shaer , "Dynamic Rule-ordering Optimization for High-speed Firewall Filtering" ASIACCS'06, March 21-24, 2006, Taipei, Taiwan (page no.332 no.332) - 342)ACM 1-59593-272-0/06/2000.

[6]. S. Acharya, J. Wang, Z. Ge, T. Znati, and A. Greenberg. "A Traffic-Aware Framework and Optimization Strategies for Large Scale Enterprise Networks". Technical Report, page no. 199-210 IEEE computer 2005.

[7]. Thomas H. Core, Charles e., Ronald L. Rivest, Clifford Stein, "Introduction to Algorithms", ISBN -81-203-2141-3, Prentice –Hall of India Private Limited-2001.