Early packet rejection based on combining multiple fields using XOR operator with balanced tree

Vu Duy Nhat† and Nguyen Manh Hung††,

Information Technology Security MoD of Vietnam, Military Technical Academy, Hanoi, Viet Nam.

Summary
The firewall device has a main task that is protecting the internal network against attacks from outside the internal network, and it must itself against attacks aimed directly at himself, one of which is offensive attack DoS against default firewall rule. Several techniques have been proposed to resist this type of attack, the proposed techniques are aimed at how to reject a packet (which will be rejected by default rule) as soon as possible to reduce resource cost and time for the rejecting that packet. The early packet rejected is done by constructing the early packet filter based on the original packet filter or properties of the data flows through the firewall and the packet rejecting is done with this early packet filter. In the early packet rejection, the examination for a coming packet is performed on all the fields in the packet header and the checked time is proportional to the number of checked fields. This paper proposes the using XOR operator to combine two or more fields together and balanced-tree construction for the purpose of reducing average processing time per coming packet in early packet rejected. The effectiveness of the proposed technique is demonstrated by experiment when compared with other techniques.

Key words:
firewall; packet classification; early packet rejection; security policies in firewall.

1. Introduction

A firewall always has a security policy that is set by the system administrator. This policy includes a set of packet filtering rules. Every rule includes the condition values of the fields to check for packet header of every packet passing (source IP address, destination IP address, source port, destination port, protocol...) and an action associated with it. The firewall packet filtering is performed in order from the first rule until finding a match rule. If the packet is not match with any rule in rules set, it will be handled by a default rule that normally are associated with prohibits operation (deny). With treatment as above, if a packet is handled by the default rule, the cost (in terms of system resources and processing time) for this packet would be the largest. Based on this characteristic, an attacker can perform a DoS attack on the firewall device by sending a large number of packets (that will be rejected by the default rule) in a short period of time, then the firewall will use the majority of its resources to handle this packets and it will be paralyzed when the number of packets to be large enough (Figure 1). A solution has been proposed to combat this attacks, it base on ideal: How to reject the packets as quickly as possible in order to avoid overloading the firewall, given the technical implementation of this idea is the technical early packet rejection on firewall.

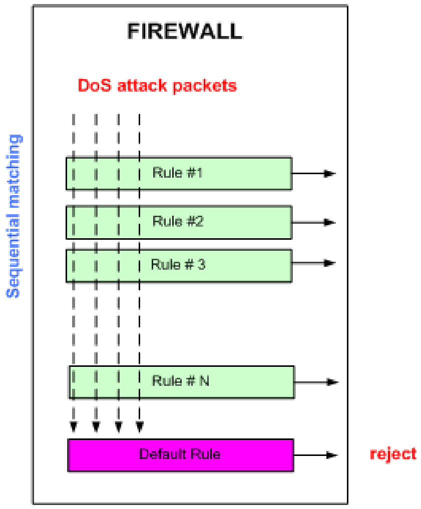


Fig. 1 DoS attacks on default rule of firewall.

The checking for a coming packet in packet classification is done on all fields that are defined by rule set. Packet Pkt matches rule R if and only if all fields in its header matches the conditions specified in the R for those fields. For example, if rule R has conditions for source IP address, destination IP address, source port, destination port, then to determine whether the packet Pkt matches the R we must check in 4 fields respectively. Time to determine whether the packet matches a rule or not will be the total time checking out all the fields. Thus the time to check a coming packet will directly proportional to the number of fields (the dimension) must be checked on this packet. In this paper, we propose a new approach for early packet rejection based on the using XOR operator to combine two or more fields in the packet header to reduce the dimensions of the packet filtering and using balanced tree construction for early packet rejected and so we will achieve effective in early packet rejection

Manuscript received October 5, 2015
Manuscript revised October 20, 2015

2. Some early packet rejection techniques

2.1 Field Value Set Cover (FVSC)

This technique analyzes the set of firewall rules to create a small rejected rule that can reject undesirable packets before forward to the original rules [1]. The basic idea of this technique is that if a packet does not match any of the common values of all rules "accept" then these packets are eligible to immediately reject. This means the rule early packet rejection can be created by combining the common values present in all policy rules. For example, if all rules "accept" uses a destination IP address or a port, all packets that do not contain the same values can be eliminated without further examination. An example of a early rejection rule can be in the form: RR = (DP $\neq$ 80) $\wedge$ (SP $\neq$ 1500) $\wedge$ (DIP $\neq$ 15.16.17.18) $\wedge$ (P $\neq$ UDP).

The biggest drawback of the technique is algorithm to find the set early packet rejection rules has complexity be NP-hard. In addition, this technique will be limited in the number of rules generated and it depends on the percentage of packets being rejected early compared to the total number of packet been rejected. The number of early packet rejection rule proportional to the number of rules in the firewall policies and the dispersion of the values in each field of early packet rejection rule. Especially, the using of approximation algorithms to generate the early packet rejection rules, when the firewall rule set changed, the ability to update this change in early packet rejection rule set is not possible.

2.2 Self Adjusting Binary Search on prefix lengths (SA-BSPL)

This technique uses the properties of the Self Adjusting Binary Search (SABS) to optimize the early packet rejection unwanted packets on the firewall. The model is given in [2] consists of a set of self adjusting filters that each filter using the binary search on prefix length [3] base on SABS tree model used to improve search time mean value [4], [5], [7].

The idea of early packet rejection on this technique is: if a packet does not match any prefix length in search of a tree filter, it will be immediately rejected. Conversely, if you find a node containing a list of n F1 rules, the inspection process will be implemented with the next filter, if the filter next stop node set includes m F2 then check F1 and F2 have the same common rule, if there is not then the packet is immediately rejected without further examination. An improvement of this technique is combined with consideration of the properties of the data flows to reduce the average time classification on each coming packet SSF-BSPL[6]. The difference of technique [6] to [2] are: packet filtering process has been carried out on all cases, however filters are arranged in descending order staring from the field with the highest rejection statistics. For example, according to the type of the packet rate due to invalid source IP address, destination IP, source port, destination port, respectively: 30%, 40%, 10%, 20%, the actual order current filtering will be done in this order: destination IP, source IP, destination port, source port.

Drawback of this technique is that, when a large number of sets of rules, the hash table and the number of prefixes in each hash table would be large. Especially, the intermediate prefixes in the hash table will make increase the memory size and search time for each packet arrives.

2.3 Policy Boolean Expression Relaxiation (PBER)

Technical PBER [8] perform packet filtering in two layers: The first layer packet classification performed by a early packet filter module; The second layer packet classification using original filter module. In this technique, if the packet passes through the first layer that is not categorized (allowed or not allowed to go through), the classification will be done in the second layer. Early packet filter module in the first layer is built in the form of logical expressions from the original sets of rules in which each rule corresponds to an expression. In this representation, each bit in the packet header is considered an input binary variable into the Boolean expression and only packets that satisfy this expression are accepted and passed through the system successfully.

Technical PBER using Binary Decision Diagrams (BDD) data structure for storing logic expressions. The checking a packet will have to be done through all the bits of the field in its header. In the nomal firewall, the number of bits in packet header (that be checked) is 104 (32 * 2 bit source and destination IP addresses, 16 * 2 bit source and destination ports, 8 bits for the protocol). To avoid cases of inefficiency of this technique the authors offer solutions: Set the height of the tree when classifying a packet, if the process has reached this threshold, the categorization on the first layer will be stopped; Set the threshold for classification ratio in the first layer, if the proportion of packets are classified at the first layer is less than this threshold, the early filter module is shutdown operations soon.

When the number of rules is large in then logical expression is generated will be very complicated, and the classification of the packet will not be efficient.

3. Proposed work

The early packet rejection techniques were introduced had use of the idea of change in order to optimize the filter or based on characteristics of the data traffic through the firewall to achieve effective in early rejected invalid packets. In such techniques, the early rejection of incoming packets must be based on information from many fields in each packet header. We find out that the number of field checked (dimensional classification) per packet will be proportional to the time classification, so if we reduce the number of checked fields on each packet then we will improve the speed of classification. However, there are some problems need to resolve with reducing the dimension in early packet rejection:

- The cost of packet processing with the reduced number of dimensions must be less than the cost of packet processing in normal case.
- The early packet rejection with the reduced number of dimensions must be accurate.
- Action to reduce the dimension classification on filters must be feasible.

Starting from the idea that we propose a technique that rejects early packet with the following key points:

- Preprocess original filter to create early packet rejected filter by using XOR operator to combine two or more fields together to form a single field (the XOR field).
- XOR is a fast speed operation.
- The early packet rejected based on the XOR field.
- The early packet rejection uses the XOR field with balanced tree structure to achieve higher speeds.

The next section we will describe the details of this proposal.

3.1 Using XOR operator to combine two or more fields together

With the aim of reducing the checked dimension when considering early packet rejected we use XOR operator to combine two or more fields into one field. For simplicity we combine three fields (source IP address, destination IP address and destination port) into a single field as follows: The firewall has a list of rules and each rule includes four parameters: source IP address prefix, destination IP address prefix, Destination port and Action <SRCPRE, DESPRE, DESPORT, ACT> (suppose that the source IP address and the destination IP address is 8 bits in length, destination port is 4 bits in length).
We make the choice to “accept” rules. For example, a firewall has a rule set with “accept” rules as in Table 1.

Table 1: The list of “Accept” rules in firewall

Rule	***Source IP prefix***	***Destination IP prefix***	***Port Destination***	***Action***
R1	101001*	11110*	2	Accept
R2	101100*	001011*	4	Accept
R3	001110*	0010*	11	Accept
R4	111*	00001*	13	Accept
R5	1101*	111000*	8	Accept
R6	10001*	001*	2	Accept
R7	011*	110*	4	Accept
R8	01010*	10101*	6	Accept

Use XOR operator to combine source IP address prefix, destination IP address prefix, Destination port into one prefix called XORFIELD prefix following principles:

The destination port (DESPORT) is converted to binary string.

The number n is defined as follows:

$$n = MIN\ (length(SRCPRE),\ length(DESPRE),\ length(DESPORT)). \qquad (1)$$

The XORFIELD prefix of the R rule is built by taking the first n bits of the IP source prefix, the first n bits of the IP destination prefix and the first n bits of destination port to XOR together:

$$\text{XORFIELD} = PRESRC(n)\ \mathbf{XOR}\ PREDES(n)\ \mathbf{XOR}\ DESPORT(n). \qquad (2)$$

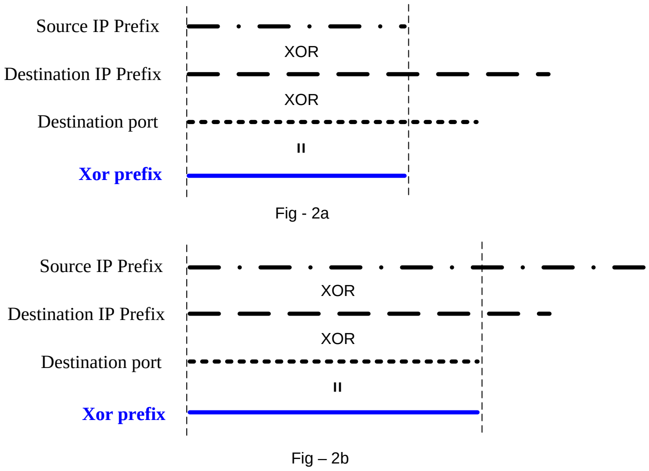


Fig - 2a

Fig – 2b

Fig. 2 Illustrate creating XORFIELD prefix

The XORFIELD prefixes that have built by (2) formula and corresponding to the rules in Table 1 as follows:

Table 2: The list XORFIELD prefixes

Rule	**Source IP prefix**	**Destination IP prefix**	**Port Destination**	**XORFIELD prefix**
R1	101001*	11110*	0010	0111*
R2	101100*	001011*	0100	1101*
R3	001110*	0010*	1011	1010*
R4	111*	00001*	1101	001*
R5	1101*	111000*	1000	1011*

R6	10001*	001*	0010	100*
R7	011*	110*	0100	111*
R8	01010*	10101*	0110	1001*

The packet classification or early packet rejection is done on the XORFIELD field has been built.

3.2 Early packet rejection bases on XORFIELD field

3.2.1 Building early packet rejection filter

The early packet rejection filter on the XORFIELD field is based on the theory of set cover.

Call the Q is the value space of the XORFIELD field, A is the set covering of the XORFIELD field in all rules with accepted action, D set is built according to the formula:

D = Q - A

D will be the set of values of the XORFIELD field which the packet is discarded (rejected).

Assuming the length of the IP address of each packet is n, then Q will be range [0, 2n-1]. So, each prefix XORFIELD in every rule will define a range of values [X1, X2] belongs to [0, 2n-1].

For example, with n = 8, then Q = [0, 255] and the range of values of XORFIELD prefix in Table 2 as follows:

Table 3: The range of values of XORFIELD prefix

Rule	XORFIELD prefix	Range of value
R1	0111*	112-127
R2	1101*	208-223
R3	1010*	160-175
R4	001*	32-63
R5	1011*	176-191
R6	100*	128-159
R7	111*	224-255
R8	1001*	144-159

According defined all elements of D shall be long to Q and will not be belong to A. D should be able to build a simple way by removing from Q all about the value determined by the prefix XORFIELD of any rule (with Accept action) in rules set, the algorithm is constructed as follows:

ALGORITHM 1: Building D

```
void Build_D()
{
    D= Q;
     for(i=1;i≤[Number of Accept rule];i++)
    {
      Ai =[range of XORFIELD prefix];
      D = D – Ai;// Remove Ai from D.
    }
}
```

Applying the algorithm 1, we find a D set associated rules set in Table 1 as follows:

Table 4: The building D set

Step	Removed range	D set
0		[0,255]
1	[112,127]	[0,111], [128,255]
2	[208, 223]	[0, 111], [128, 207], [224, 255]
3	[160, 175]	[0, 111], [128, 159], [176, 207], [224, 255]
4	[32, 63]	[0, 31], [64,111], [128, 159], [176, 207], [224, 255]
5	[176, 191]	[0, 31], [64,111], [128, 159], [192, 207], [224, 255]
6	[128, 159]	[0, 31], [64,111], [192, 207], [224, 255]
7	[224, 255]	[0, 31], [64,111], [192, 207]
8	[144, 159]	[0, 31], [64,111], [192, 207]

The finally, D set is [0, 31], [64,111], [192, 207].

3.2.2 Early packet rejection

The early packet rejection will be done by examining the packet arrived on set D (that is built in algorithm 1). If a packet arrives, with the XORFIELD (generated by XORing the source IP address, destination IP address and destination port) belongs to D, then surely it will be rejected by the firewall. The creating XORFIELD of a coming packet is demonstrated in Fig.3

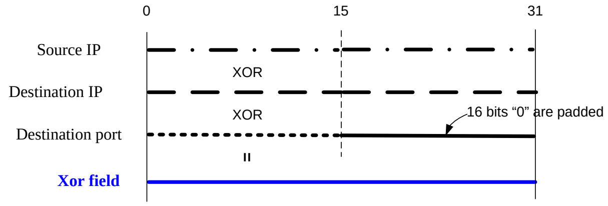


Fig. 3 Creating XOR field of a coming packet.

ALGORITHM 2: Early Packet Rejection in D set

```
void EarlyPacketReject(packet pkt)
{
  XORFIELD= [pkt.ipsource] XOR
                 [pkt.ipdestination] XOR
                 [pkt.portdestination]
  if XORFIELD∈ D then
    REJECT
  else
    forward pkt to original Firewall;
    //continued check in original filter of
     firewall
}
```

3.2.3 Accuracy in Early packet rejection of proposed technical

To confirm the accuracy of the early packet rejection of the proposed technique, we prove the following two theorems:
THEOREM 1 If a packet Pkt with the XORFIELD (created by XORing three fields the source IP address, IP destination address and the port destination) does not belong to the range value of the XORFIELD field in the R rule then at least Pkt.IPSource not match source IP prefix of R or Pkt.IPDestination not match destination IP prefix of R or Pkt.PortDestination not match destination port of R.

Proof:
Suppose R rule has prefix XORFIELD prefix is PrefixXOR which length is n bits.
The range of values determined by PrefixXOR is $[V \times 2^{32-n}, V \times 2^{32-n} + 2^{32-n} - 1]$, where V is the value of of PrefixXOR in base-10 system.

Packet Pkt has *Pkt.XORFIELD = [Pkt.ipsource]* XOR *[Pkt.ipdestination]*XOR *[Pkt.portdestination]* *(**ValueXORFIELD*** = Pkt.XORFIELD-value).

Because ***ValueXORFIELD*** does not belong to range value of ***PrefixXOR***, the occurrence of one of the two following cases:

(a) ***ValueXORFIELD*** $< V \times 2^{32-n}$ or
(b) ***ValueXORFIELD*** $> V \times 2^{32-n} + 2^{32-n} - 1$

We have
$V_1 \times 2^{32-n} + 2^{32-n} - 1 \geq$ ValueXORFIELD $\geq$ V1$\times 2^{32-n}$ + V2$\times 2^{32-n-1}$ (3)
where V1 is the value in the base-10 system of first n-bit of ValueXORFIELD, V2 is value in the base-10 system of ValueXORFIELD's bits (that starts (n + 1)th bit to end).
Case (a):
We have ValueXORFIELD$< V \times 2^{32-n}$ combine with (3) so:
V1 $\times 2^{32-n}$ + V2 $\times 2^{32-n-1}$ $\leq$ValueXORFIELD$< V \times 2^{32-n}$
↔ V1$\times 2^{32-n}$ + V2$\times 2^{32-n-1} < V \times 2^{32-n}$
↔ (V–V1)2^{32-n}> V2$\times 2^{32-n-1}$ ↔ V–V1 >$\frac{V_2}{2}$
Since V2 ≥ 0 so V–V1 > 0 or other word V not equal V1, thereby deduce the first n bits of ValueXORFIELD not equal the first n bits of PrefixXOR and results in at least one case: the first nbits of the source IP address of the Pkt not match the first nbits of R.IPSourcePrefix, the first n bits of the destination IP address of the Pkt not match the first n bits of R.IPDestinationPrefix or the first n bits of the destination port of the Pkt not match the first n bits of R.PortDestination.
Case (b): We have ValueXORFIELD> $V \times 2^{32-n} + 2^{32-n} - 1$ combine with (3) so:
↔ $V_1 \times 2^{32-n} + 2^{32-n} - 1 > V \times 2^{32-n} + 2^{32-n} - 1$
↔ (V1–V)$2^{32-n} > 0$↔ V1– V > 0 ↔ V1> V
We have V not equal V1, so the result is similar case (a).
Results of two cases (a) and (b) indicate that: At least Pkt.IPSource not match source IP prefix of R rule or Pkt.IPDestination not match destination IP prefix of R rule.

THEOREM 2 When a Pkt packet with the XORFIELD field belongs to the D, then it will not satisfy any ACCEPT-rule yet.

Proof:
We suppose Pkt packet with the XORFIELD field belonging to the D which correspond ith ACCEPT-rule, then we have D will have to contain at least one element in the interval defined by the prefix of XORFIELD ith rule, this is conflict with the building set D above. Therefore, if you have the XORFIELD PKT belongs to the D then it will not satisfy any ACCEPT-rule.

3.3 Using the balanced tree structure with XOR field in early packet rejection.

Binary search tree (BST) with great advantage as easily perform insertion and deletion, very convenient for handling collective dynamic elements. However, BST only effective if data is inserted into the tree be the random key values. If data is sorted before inserting into the tree then the BST will not be effective, in this case will BST becomes a linked list. To overcome this problem, the AVL tree[10] and a red-black tree [10] are proposed.
AVL and Red-black trees have inserting, deleting operators and rotation operators to rebalance tree.
In our proposed technique, we use balaced tree to: Build and store D set; The early packet rejection is done in that balance tree.

3.3.1 Build balanced tree

The building balanced tree has input parameter be list of "ACCEPT" rules and output parameter be balance tree that stores D set. A node of tree has key be a range that belongs to D set.
ALGORITHM 3: Build Balanced-Tree

Step 1: Create root node of balanced tree with key value [0, MAX], (If the length of the source IP address, IP destination is 32 bits, then MAX= 2^{32} ...).

Step 2: Building XORFIELD prefix of corresponding rule and converted into the segment [a, b].

Step 3: Insert the segment [a, b] into the tree.

Check segment [a, b] in the tree, assuming current node N_i with key value [x, y], consider the following cases:

Table 5: The rule for inserting segment[a, b] into the balanced tree.

Case	Action
(a<x) and (b>y)	Delete Ni of the balanced tree. Insert two node [a,x-1], [y+1,b] into tree.
(a=x) and (b=y)	Delete Ni of the balanced tree.
b < x	Insert node [a,b] into left child of Ni.
a > y	Insert node [a,b] into right child of Ni.
a < x < b <y	Update key value of Ni to [b+1, y] Insert new node with key value [a,x-1] into left child of Ni.
x < a < y < b	Update key value of Ni to [x, a-1] Insert new node with key value [y+1,b] into right child of Ni.

Rebalance the tree.

Step 4: Go to step 3 until all “ACCEPT” rules inserted into the tree.

3.3.2 Early packet rejected with balanced tree

The early packet rejection is done on balanced tree, with a coming packet:

- The first, we calculate XOR field (source IP address XOR destination IP address XOR destination port).
- The seconds, search XOR field on the balanced tree, if the packet to have the XOR belongs to a range of key value of a node on balanced tree, then it will be rejected soon, otherwise the packet will be further checked on the original rules set of firewall.

The range of values on the D set is completely separate from each other, so if two value ranges [a1, b1] and [a2, b2] belong to D then we always have a1> b2 or a2> b1. Early packet rejection algorithm on balanced tree as follows:

ALGORITHM 4: RejectPacketOnBalanced-Tree()

```
Packet pkt;
Balanced-tree btree;
Begin
 node := btree;
 XOR-F:=[pkt.SourceIP]
          XOR [pkt.desIP]
          XOR [pkt.desPort<<16];
   While node<>NULL do
   Begin
     If (XOR-F belongs to node.key)
     then reject Pkt
     Else
     if (XOR-F <node.key.min)
       node = node.left
     Else node = node.right
   end
end
```

In above procedure we use 16-bit left shift with packet destination port that is the padding 16 bits 0 out the destination port.

3.3.3 Conditions for the proposed technique is effective

Call T1 is the average time to process a packet with the proposed technique, T2 is the average time to process a packet in the original firewall, M is total of number packets that pass the firewall, P is the percentage of the packet is discarded by early packet rejection module.

Time to process M packet with original filter is T2M.

Time to process M packet with proposed technique is T1M + T2(1-P)M.

In order to work effectively with early packet rejection, we should need:

$$T_1M + T_2(1-P)M< T_2M \leftrightarrow T_1M < T_2MP\leftrightarrow P >\frac{T_1}{T_2} \quad (1)$$

According to (1) the effectiveness of early packet rejection with proposed technique will depend on the ratio of packets being rejected early. P is only meaningful reality when T1<T2. T1 will be proportional to the number of values in the set D. In the case of sets of rules that D set was build with a large number of elements, the proposed technique may not be effective.

In fact determining T1 and T2 is very difficult so we can test to choose a threshold value Pmin. When P <Pmin, the proposed technique will not be effective and then the early rejection module will soon be deactivated and the packet classification will be made by the original classification module of the firewall.

4. Installation testing and evaluation

To verify the accuracy and effectiveness of using XOR operation to combine two or more fields into a single field with balanced tree in the early packet rejection and packet classification in firewall we have installed and tested the proposed technique and the results has compared with techniques have been proposed previously.

The tested program has written in C language. We've run the test program on the PC with Intel Dual Core 2.8Ghz CPU, 2Gb RAM and installed on the Ubuntu 12.04 operating system.

To ensure close to real applications, programs using artificial data generated by the ClassBench tool that was created by David E. Taylor, Jonathan S. Turner of Applications Research Laboratory, Faculty of Computer Science, Washington University, Saint Louis[13]. The data sets include sets of rules and sets of parameters in packet on the input data are real data sets obtained from Internet service providers. This is the public research community to evaluate the use of algorithms and packet classification device.

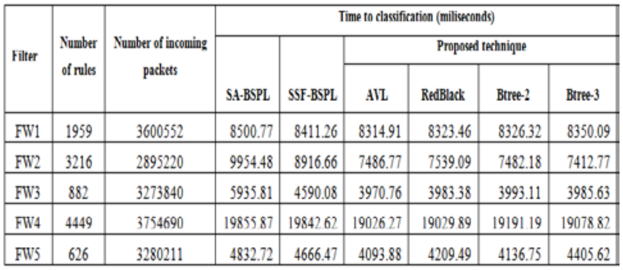
4.1 Compare with other early packet rejection techniques

We have installed the proposed technique on AVL-tree, RedBlack-tree, B-tree (with degree 2 and 3) structures. Testing's results are compared in terms of time to packet filtering with SA-BSPL and SSF-BSPL techniques (that are 2 techniques was identified as better than FVSC and BPER techniques).

4.1.1 Experiment with different filters

We have used 5 filters in database of Classbench tool: FW1, FW2, FW3, FW4 and FW5. Each filter was created together with a packet data file using for packet classification.

Table 6: Results with the different filters



Filter	Number of rules	Number of incoming packets	Time to classification (miliseconds)					
					Proposed technique			
			SA-BSPL	SSF-BSPL	AVL	RedBlack	Btree-2	Btree-3
FW1	1959	3600552	8500.77	8411.26	8314.91	8323.46	8326.32	8350.09
FW2	3216	2895220	9954.48	8916.66	7486.77	7539.09	7482.18	7412.77
FW3	882	3273840	5935.81	4590.08	3970.76	3983.38	3993.11	3985.63
FW4	4449	3754690	19855.87	19842.62	19026.27	19029.89	19191.19	19078.82
FW5	626	3280211	4832.72	4666.47	4093.88	4209.49	4136.75	4405.62

Test results are shown in the following Table 2. It has shown that: Our proposed technique has different efficiency for different filters, but this technique is more effective than SA-BSPL and SSF-BSPL techniques.

4.1.2 Experiment with different number of coming packet

Results of packet classification with proposed technique in comparison with SA-BSPL and SSF-BSPL techniques shown in the Fig. 4 (the experimental results of a fixed number rules in 626 and change number of coming packets):

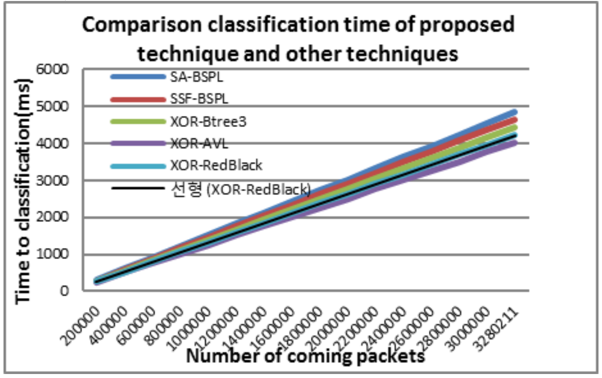


Figure 4 – Comparison classification time of technique proposed and other techniques.

In the testing, the time to classification with our proposed technique includes time to early rejecting and time to processing packet in original filter (if packet not be early rejected).

The results in Fig 4 showed that our proposed technique more effective than SA-BSPL and SSF-BSPL techniques and

4.2 The experiment extended

In this section we expand the experiment to test the effectiveness of using XOR operator to reduce the dimension of the classification of packets. We tested in two cases:

- Using XOR field in packet classification with sequential search algorithm.
- Using the XOR field in technique SA-BSPL

4.2.1 Using XOR field in packet classification with sequential search algorithm

For simplicity, we assume that the packet classification bases on three-dimensional be source IP address, destination IP address and destination port. With a packet coming and R rule, we consider two cases:

Case 1: Packet classification is based on the common

- Step 1: Checkout the source IP, if source IP match with source IP prefix of R then forward to step 2;else skip checking with R.
- Step 2: Compare destination IP with destination IP prefix of R. If destination IP matches destination IP prefix then forward to step 3; else skip checking with R.
- Step 3: Compare destination port with destination port prefix of R.

Case 2: Packet classification using the XORFIELD field

- Step 1: Check out the XORFIELD, if XORFIELD match with XORFIELD prefix of R then forward to step 2;else skip checking with R.
- Step 2: Check out field that have the largest prefix length: if match then forward to step 3;else skip checking with R
- Step 3: Check outfield that have the second largest prefix length.

Theoretically, the two cases are the same number of steps, however, case-2 will have to add the cost of preprocessing rules and the process of implementing the XORing (IP source XOR IP destination XOR destination port) for each packet arrives. However, in practice the efficiency of Case-2 is determined by the probability to perform the steps 2 and 3 will be lower than Case 1.The results made with the different data sets as follows:

Table 7: Results with the different data sets

Filter	Numb er of rules	Number of incoming packets	Time to classification (miliseconds)	
			Case 1	Case 2
FW1	1959	3600552	39509.20	37260.47
FW2	3216	2895220	47325.12	46758.04
FW3	882	3273840	17748.26	17561.33
FW4	4449	3754690	78578.11	78018.50
FW5	626	3280211	13611.84	13333.15

4.2.2 Using the XOR field in technique SA-BSPL

In SA-BSPL technical, prefixes (source IP and destination IP, destination port) are stored in arrays of hash tables, the *ith* hash table will store the prefixes that have length is *i*. The classification of coming packets will be performed on each field and the final result is made by finding common elements of sub-results.

The using XOR field in technique SA-BSPL was adjusted as follows:

- The prefixes are four fields: the XOR prefix, the source IP prefix, the destination IP prefix and destination port in 4 arrays hash tables.

- The stored prefixes follow the principle: The R rule is stored field prefixes: XORFIELD prefix, field prefix that have the largest length and field prefix that have the second largest length.

Total number prefix to the storage in this case are equal to the total amount of prefixes must be stored in normal case of technical SA-BSPL; the accuracy of the classification of packets is proven in Section 3.2.3.

The experimental results when we fixed number of rules in 626 and change packets coming:

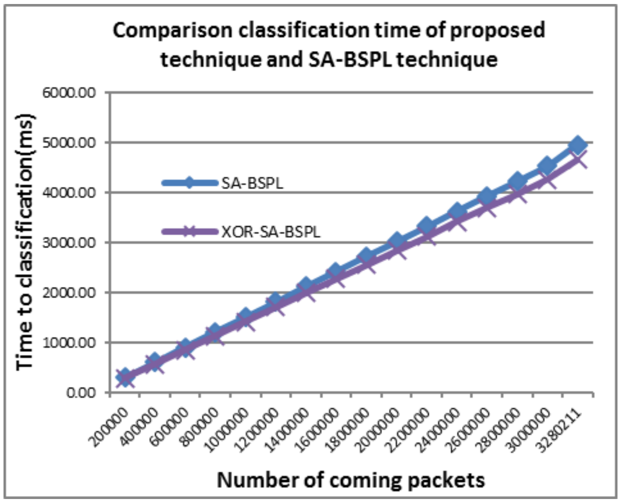


Figure 5 – Result of comparison of the time when using the XOR field for classification in SA-BSPL technique.

The results in Fig 5 showed that the using XOR field more effective than normal cases in SA-BSPL technical.

5. Conclusion

This paper presents the significance of the problem rejected early packet in firewall and summary techniques early packet rejections have been studied. Base on analyzing the advantages and disadvantages of the techniques, we propose solution to combine two or more fields together into a field by using XOR operation to achieve effective time during packet filtering on the firewall. We have shown in detail how to use the XOR field in the early packet rejection and packet classification and demonstrate the accuracy and efficiency of this solution. Test results show that the proposed solution really effective and has highly capable of deployment in the practical.

References

[1] H. Hamed, A. El-Atawy, E. Al-Shaer. “Adaptive StatisticalOptimization Techniques for Firewall Packet Filtering”. In Proceeding of IEEE INFOCOM, pp. 1-12, 2006.

[2] N. Neji, A. Bouhououla. “Dynamic Scheme for Packet Classification Using Splay trees”. Information Assurance and Security, pp. 1-9, 2009.

[3] M. Waldvogel, G. Varghese, J. Turner, B. Plattner. “Scalable High Speed IP Routing Lookups”. In Proceedings of the ACMSIGCOMM (SIGCOMM ’97), pp. 25-36, 1997.

[4] T. Srinivasan, M. Nivedita, V. Mahadevan. “Efficient Packet Classification Using Splay Tree Models”. IJCSNS International Journal of Computer Science and Network Security, 6(5), pp. 28-35, 2006

[5] D. Sleator, R. Tarjan. “Self Adjusting Binary Search Trees”. Journal of the ACM, 32(3), pp. 652-686, 1985.

[6] Zouheir Trabelsi, Safaa Zeidan: Multilevel early packet filtering technique based on traffic statistics and splay trees for firewall performance improvement. ICC 2012: 1074-1078.

[7] T. Srinivasan, M. Nivedita, V. Mahadevan. “Efficient Packet Classification Using Splay Tree Models”. IJCSNS International Journal of Computer Science and Network Security, 6(5), 2006, pp. 28-35.

[8] E. Al-Shear, A. El-Atawy, T. Tran. “Adaptive Early Packet filtering for Defending firewalls against DoS Attack”. In Proceeding of IEEE INFOCOM, pp. 1-9, 2009.

[9] G.M. Adelson-Velsky và E.M. Landis. “An algorithm for the organization of information”, 1962.

[10] Nguyễn Mạnh Hùng. Cấu trúc dữ liệu nâng cao, NXB Quân đội Nhân dân, 2012.

[11] Leo J. Guibas and Robert Sedgewick. A dichromatic Framework for Balanced Trees, in Proc. 19th IEEE Symp. Foundations ò Computer Science, 1978.

[12] R. Bayer. Symmetric binary B-Tree: Data Structures and maintenance algorithms, Acta Information, Volume 1, 1972.

[13] http://www.arl.wustl.edu/classbench.

[14] R. Bayer. Symmetric binary B-Tree: Data Structures and maintenance algorithms, Acta Information, Volume 1, 1972.