Test Tool for Equivalence of Access Control List

Hirokazu Sayama  
Department of Information and Computer Sciences  
Saitama University  
255 Shimo-ookubo Sakura-ku Saitama Japan  
Email: hsayama@fmx.ics.saitama-u.ac.jp

Noriaki Yoshiura  
Department of Information and Computer Sciences  
Saitama University  
255 Shimo-ookubo Sakura-ku Saitama Japan  
Email: yoshiura@fmx.ics.saitama-u.ac.jp

Abstract—Computer network security is one of the important issues in the Internet age. Network administrators of organizations such as companies or universities filter IP packets at network equipment such as Layer 3 switch or firewall between their organizations and the Internet to keep the security of the computer networks. One of the expressions of the filtering rules of IP packets is access control list. Access control lists are lists of rules, which describe permission or denial of packet transition based on source IP address, destination IP address, port numbers and so on. Access control lists are not always fixed; network administrators change access control lists according to the change of network topology or network security policy. After several changes, access control lists may include redundancies and network administrators have to modify the access control list to remove redundancies. This modification must keep the semantics of access control list. After modification, the network administrators must confirm that the semantics of access control list does not change. One of the methods of equivalence of two access control lists is to send test IP packets to the network equipment that filters IP packets and to check the transitions of the IP packets. This paper proposes the method of generating test packets to confirm the equivalence of two access control lists.

Index Terms—Access Control List; Network Operation; Computer Network Security;

I. INTRODUCTION

Computer network security is one of the important issues in the Internet age. Network administrators of organizations such as companies or universities filter IP packets at network equipment such as Layer 3 switch or firewall between their organizations and the Internet to keep the security of the computer networks. The firewalls are one kind of the equipment that protects computer networks from computer network attacks such as portsweep, brute force attack and so on. The firewalls check whether the IP header of a packet matches predefined rules. The set of predefined rules is called "Access Control List" (ACL).

ACLs are lists of rules, which describe permission or denial of packet transition by source IP address, destination IP address, port numbers and so on. ACLs are not always fixed; network administrators change ACLs according to the changes of network topology or network security policy. These changes of ACLs are done in the network equipment at work. Therefore, the network administrators have to change ACLs without stopping the network equipment and prevent troubles from occurring due to ACL changes. It follows that the network administrators change ACLs as few as possible or only add new rules to current ACLs in order to keep current ACLs unchanged like Figure 1.

After these changes, ACL may include redundancies, which network administrators have to remove from ACLs by modifying ACLs. The modifications of ACLs for removal of redundancies must keep the semantics of ACLs of pre-modification and post-modification; the ACL before modification must be equivalent to that after modification. The equivalence between two ACLs is that for any packet, it can pass one ACL if and only if it can pass the other ACL. After modification, the network administrators must confirm that the semantics of ACL does not change.

There are several researches of ACLs; many researches of them are related with optimization of ACLs[2], [3], [5], which is a NP-complete problem[1]. The optimization of ACLs is different between in the case that ACLs are processed in software and in the case that ACLs are processed in hardware. In software processing of ACLs, the optimal ACL is the ACL in which processing time of all packets are minimum. On the other hand, in hardware processing of ACLs, the optimal ACL is ACL whose size is minimum.
because the hardware processing time of ACL is $O(1)$. Thus, the optimal ACL does not depend on the kinds of packets that pass network equipment.

As compared with optimization of ACL, equivalence check of ACLs have not been researched so much. However, there are several methods of equivalence check of two ACLs. One of them is to use two or over two dimensional space and map the semantics of ACLs into the space[4]. Equivalence check is converted to equivalence check of mapped space. However, the complexity of equivalence check seems NP-hard because equivalence check can be converted to 3-SAT problem and theoretical approaches of equivalence check of ACLs are impractical.

This paper proposes a method of equivalence check of two ACLs. This method consists of generating test packets from two ACLs, sending the test packets to network equipment under each of two ACLs and checking the correspondence between test packets passing under two ACLs. Figure 2 shows the outline of the proposed method. The easiest method of generating test packets is to generate all packets by combining all IP addresses for source and destination addresses. This method generates $2^{64}$ test packets because of $2^{32}$ IP addresses. However, these test packets are too many to check equivalence of two ACLs. This paper proposes a test packet generating method that can generate test packets that can be used practically for equivalence check of two ACLs. This paper also proves that this test packets are enough to check equivalence of two ACLs; two ACLs are equivalent if and only if the generated test packets can pass through one ACL are the same as those through the other ACL. Some network equipment processes ACL by hardware; ternary content addressable memory (TCAM) is used to process ACLs. Hardware processing of ACL has high performance and many packets can be processed by TCAM in a short duration. Therefore, many test packets can be processed in practical duration and using test packets can check equivalence of two ACLs in practical duration.

This paper is organized as follows; Section 2 describes the method that is proposed in this paper. Section 3 shows the proof sketch of correctness of the method. Section 4 shows the results of experiment. Section 6 discuss the results of the experiment. Section 5 concludes this paper.

II. EQUIVALENCE CHECK OF ACCESS CONTROL LISTS

Figure 2 shows the outline of the proposed method. This paper defines test packets to be pairs of source IP address and destination IP address. This paper also assumes IPv4 network; maximum test packets are $2^{64}$ test packets.

A. ACL of OmniSwitch

This paper uses ACLs of the network equipment in the university that the authors belong to. The backbone network equipment of the university is OmniSwitch of Alucatel Lucent. Table I shows the description of ACLs in OmniSwitch [6], [7].

B. Test Packet Generation

$2^{64}$ test packets are too many to use for equivalence check of ACLs. Thus, this paper proposes the method of generating a smaller number of test packets. This paper uses OmniSwitch ACLs to propose the test generation method, which can be used for another network equipment such as Cisco Catalyst Switch and so on because the difference between ACLs of OmniSwitch and almost all network equipment is only syntax of ACLs. Especially, OmniSwitch ACL is more complicated than Cisco Catalyst Switch and other network equipment. Thus the proposed method can be used for other network equipment by a little modification.

ACLS can be described by source IP address, destination IP address, source port number, destination port number or so on, but this paper deals with source IP address and destination IP address because of the following reasons; the authors have operated network equipment at several universities and organizations, in which almost all parts of ACL are related only with source and destination IP addresses. This is one of the reasons why the paper focuses on source and destination IP addresses. Another reason is that this attempt of generating test packets is the first for the authors, who would like to simplify generating test packets.

This paper focuses on condition description in OmniSwitch ACLs. This description includes source and destination IP addresses for packet filtering. The proposed method of test packet generation uses two addresses in condition description in ACLs.
C. Test packet generation by condition description

The proposed method generates test packets from source IP address and destination IP addresses in condition description. Table II and Table III illustrate how to generate test packets.

<table>
<thead>
<tr>
<th>Source IP Address</th>
<th>Destination IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>133.36.0.0/14</td>
<td>180.20.10.0/24</td>
</tr>
<tr>
<td>133.32.0.0/12</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>133.0.15.0/24</td>
</tr>
</tbody>
</table>

Condition C1 in Table II consists of network group G1 and G2 as source IP address and destination IP address. Network group G1 consists of IPaddressA/mask and IPaddressB/mask and network group G2 consists of IPaddressC/mask. Test packets for condition C1 are packets from IPaddressA/mask to IPaddressC/mask and from IPaddressB/mask to IPaddressC/mask. These test packets are generated by ad hoc method. Similarly the test packets for condition C2 are generated. Table III shows test packets generated by the ad hoc method.

D. Reducing test packets by subnetmask

The test packets that are generated by the previous procedure can be reduced by subnetmask. The packets IPaddressA/mask → IPaddressB/mask represent many test packets whose source IP address belongs to IPaddressA/mask and destination IP address belongs to IPaddressB/mask. However, it is unnecessary to check whether all the test packets can pass under ACL1 and ACL2. One of the test packets is enough to check equivalence between ACL1 and ACL2. This proposition will be proved in the next section in this paper. Figure 3 shows the overview of reducing test packets. This figure considers the following three kinds of test packets.

133.36.0.0/14 → 180.20.10.0/24
133.32.0.0/12 → 0.0.0.0/0
0.0.0.0/0 → 133.0.15.0/24

Fig. 3. Reducing test packets

E. Implementation

This paper implements two methods of generating test packets from two ACLs. One is an implementation of the method in Subsection II-C; it generates test packets from ACL1 and ACL2 by combining all pairs of source and destination addresses. It is called Procedure A. The other is an implementation of the method in Subsection II-D; it generates test packets from ACL1 and ACL2 by picking one test packets for each area in two dimensional space of source and destination IP addresses. It is called Procedure B. This paper implements Procedure A and B by Perl.

F. Correctness of test packets

The following theorem shows correctness of the proposed method. The proof of the theorem is omitted because of space.

Theorem 1. The following two propositions are equivalent.

- Two ACLs are equivalent.
- Test packets that Procedure B generates from two ACLs behave equivalently under two ACLs.
III. Experiment

This section shows the experiment of the proposed method; we use Procedure A and B to generate test packets from two ACLs. Table IV shows two ACLs that are used in this experiment. Table VI shows the number of test packets that are generated by Procedure A and B.

<table>
<thead>
<tr>
<th>TABLE IV</th>
<th>TWO ACLS USED IN THE EXPERIMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACL1</td>
</tr>
<tr>
<td>Maximum number of policy rules</td>
<td>103</td>
</tr>
<tr>
<td>Maximum number of policy conditions</td>
<td>103</td>
</tr>
<tr>
<td>Maximum number of policy actions</td>
<td>2</td>
</tr>
<tr>
<td>Maximum number of policy service</td>
<td>13</td>
</tr>
<tr>
<td>Maximum number of policy groups (network, MAC, service, port)</td>
<td>181</td>
</tr>
</tbody>
</table>

Table V shows the environment of the experiment. This paper can expect the time of equivalence check by test packets in Table VI. This paper assume that network equipment has 1 Gbps interfaces and wire speed ability of processing IP packets. Such network equipment has 1488100 pps (packet per second). Table VII shows expected time to check behavior of test packets. This result shows that test packets that are generated by Procedure B can be checked in practical duration.

<table>
<thead>
<tr>
<th>TABLE V</th>
<th>EXPERIMENT ENVIRONMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>Windows Vista 32bit</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel(R)Core2 Duo 2.40GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>2.00GB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE VI</th>
<th>THE NUMBER OF TEST PACKETS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Procedure</td>
<td>the number of test packets</td>
</tr>
<tr>
<td>All test packets (I)</td>
<td>$2^{64}$</td>
</tr>
<tr>
<td>Procedure A (II)</td>
<td>$27399981123850$ ($=2^{45}$)</td>
</tr>
<tr>
<td>Procedure B (III)</td>
<td>$160446$ ($=2^{17}$)</td>
</tr>
</tbody>
</table>

This paper only proposes the method of generating test packets and shows that test packets can be checked in practical duration. However, to use the proposed method requires solutions of several problems; one of the problems is how to check transitions of test packets. Network equipment has high ability of processing IP packets, but PCs do not have the same ability. PCs that are used in the proposed method would require more time to check transitions of test packets than the network equipment. One of the solutions is creating PCs that can check transitions of test packets in practical duration. If firewalls, which handle ACLs, are used to check transitions of test packets, this check requires to stop the firewalls or the firewalls are required to process practical packets and test packets at the same time. Since the test packets that are generated in the experiment are not many, they can be checked in the running network equipment without making influence to practical packet processing.

Another problem of using the proposed method is hardware that sends and receives test packets. Using the network equipment to check transitions of test packets requires hardware of sending and receiving test packets. If the network equipment has many network interface ports, the hardware has to have many network interface ports. The hardware also has to have ability of sending and receiving packets without dropping packets that should be received.

V. Conclusion

This paper proposed the method of checking equivalence of two ACLs. This method generates test packets from two ACLs. Transitions of test packets under two ACLs decides the equivalence of two ACLs. Test packets are generate in practical duration. The experiment of this paper showed that test packets can be checked in practical duration by the network equipment.

There are several future works; one of the important works is to create hardware or software to check behavior of test packets.

REFERENCES