Detecting Policy Anomalies in Firewalls by Relational Algebra and Raining 2D-Box Model

Naveen Mukkapati[†], **Ch.V.Bhargavi**[††]

[†]CSE Dept, NRI Engineering College-Vijayawada, Andrapradesh.
[††] IT Dept, Laqshya Inistitute of Science and Technology-Khammam, Andrapradesh.

ABSTRACT
Firewalls are crucial elements in the computer networks. Due to lack of tools for analyzing firewall policies, most firewalls on the internet have been plagued with policy anomalies. In this paper, we propose a method; which analyzes the firewall by using Relational Algebra and Raining 2D-Box Model. It can find out all the anomalies in the firewall rule-set in the format that is usually used by many firewall products such as Cisco Access Control List, IPTABLES, IPCHAINS and Check Point Firewall-1. While the existing analyzing methods consider the anomalies between any two rules in the firewall rule-set, we consider more than two rules together at the same time to find out the anomaly. Therefore we can find all the hidden anomalies in the firewall rule-set. Results from analyzing can be used with the proposed rules-combination method presented in this paper, to minimize the firewall rule without changing the policy. This method could help administrator to analyze and modify a complex firewall policy.

Keywords
Firewall, policy, relational algebra, correlation anomaly, raining 2D-Box Model

1. INTRODUCTION

A firewall is a system that acts as an interface of a network to one or more external networks and regulates the network traffic passing through it. The firewall decides which packets to allow to go through or to drop based on a set of "rules" defined by the administrator. These rules have to be defined and maintained with utmost care, as any slight mistake in defining the rules may allow unwanted traffic to be able to enter or leave the network, or deny passage to quite legitimate traffic. Unfortunately, the process of manual definition of the rules and trying to detect mistakes in the rule set by inspection is very prone to errors and consumes a lot of time. Thus, researches in the direction of detecting anomalies in firewall rules have gained momentum of recent.

Firewall rules are usually in the form of a criteria and an action to take if any packet matches the criteria, these actions are usually may accept or reject. A packet arriving at a firewall is tested with each rule sequentially. Whenever it matches with the criteria of a rule, the action specified in the rule is executed, and the rest of the rules are skipped. For this reason, firewall rules are order sensitive. When a packet matches with more than one rule, the first such rule is executed. Thus, if the set of packets matched by two rules are not disjoint, they will create anomalies. For instance, the set of packets matching a rule may be a superset of those matched by a subsequent rule. In this case, all the packets that the second rule could have matched will be matched and handled by the first one and the Second rule will never be executed. More complicated anomalies may arise when the sets of packets matched by two rules are overlapped.

Conflicts and incorrect order within firewall rules can make system work improperly. Writing a rule-set usually contain many hidden conflicts. Pasi Eronen [2] proposed an Expert System that is based on constraint logic programming (CLP) for user to write higher-level operations for detecting common configuration mistakes. Scott Hazelhurst [3] using Binary Decision Diagrams (BDDs) to present and analyze rule-set. Using SET theory Ehab Al-Shader et.al [1] presents an anomaly discovery algorithm. They presented a method for finding some error within rule-set that called "anomaly". But their research cannot discover all anomalies when considering more than two rules at the same time.

In this paper, we propose an alternative approach using Relational Algebra (RA) technique and Raining 2DBox Model for finding anomaly within the rule-set. It can ascertain the entire hidden anomaly when considering more than two rules together. This paper is organized as follows. In section 2, we present how to map the firewall rules into Relation [4] using Cartesian product. In section 3, we classify and define firewall policy anomaly. In section 4, we present how to remove anomaly and how to reduce the rule-set's size by combining many rules together. We conclude this paper in section 5.

2. FIREWALL AND RELATIONAL ALGEBRA BACKGROUND

Firewall can allow or deny any packets by considering the specified rule-set. Example of rule-set is shown in Fig. 1. Firewall rule has four fields. Those are source-ip address, destination-ip address, destination port number and action.

Manuscript received May 5, 2013
Manuscript revised May 20, 2013

Action is either accept or deny. Accept indicates that allowing particular packet into the network. deny indicates that disallowing particular packet into the network.
The term" any" in the fig-1 indicates that any port number is the destination port number, any address is the destination address and any address is the source address.

Order	src_ip	dst_ip	dst_port	Action
1	1.0.0.5	2.0.0.1	20-21	accept
2	1.0.0.5	2.0.0.1	21-22	accept
3	1.0.0.5	2.0.0.1	23-25	deny
4	1.0.0.5	2.0.0.1	22-23	accept
5	1.0.0.0/30	2.0.0.1	80	deny
6	1.0.0.1	2.0.0.0/30	80	accept
7	1.0.0.0/30	2.0.0.0/30	80	deny
8	1.0.0.1	2.0.0.1	80-143	accept
9	1.0.0.1, 0.4	2.0.0.1	80	deny
10	1.0.0.0/24	2.0.0.0/24	any	accept
11	1.0.0.0/24	2.0.0.0/24	80	deny
12	any	any	any	deny

Fig-1: Example of firewall rule-set

Relation is a subset of Cartesian product of domain [4]. Relation Algebra is a procedural query language that consists of a set of operation on the Relation(s). Example of operation in Relational Algebra are SELECT, PROJECT, UNION, and DIFFERENCE (See Fig-2 to Fig-9). Example; when src_ip = 3.0.0.0/30, dst_ip = 4.0.0.1, dst_port = 80, and action = deny, it can be mapped to Relation 1 (R1) as shown in Fig. 2, while R2 is the relation that is mapped from rule; src_ip = 3.0.0.0/31, dst_ip = 4.0.0.1, dst_port = 80-81, action = accept.

R1 (Relation 1) is

src_ip	dst_ip	dst_port	action
3.0.0.0	4.0.0.1	80	deny
3.0.0.1	4.0.0.1	80	deny
3.0.0.2	4.0.0.1	80	deny
3.0.0.3	4.0.0.1	80	deny

Fig- 2: Relation R1

R2 (Relation 2) is

src_ip	dst_ip	dst_port	action
3.0.0.0	4.0.0.1	80	accept
3.0.0.0	4.0.0.1	81	accept
3.0.0.1	4.0.0.1	80	accept
3.0.0.1	4.0.0.1	81	accept

Fig -3: Relation R2

PROJECT
R3 =project(src_ip,dst_ip,dst_port) R1

src_ip	dst_ip	dst_port
3.0.0.0	4.0.0.1	80
3.0.0.1	4.0.0.1	80
3.0.0.2	4.0.0.1	80
3.0.0.3	4.0.0.1	80

Fig- 4: Relational Algebra Operation Project on R1

PROJECT
R4 =project(src_ip,dst_ip,dst_port) R2

src_ip	dst_ip	dst_port
3.0.0.0	4.0.0.1	80
3.0.0.0	4.0.0.1	81
3.0.0.1	4.0.0.1	80
3.0.0.1	4.0.0.1	81

Fig-5: Relational Algebra Operation Project on R2

SELECT
R5 = select (dst_port=80) R4

src_ip	dst_ip	dst_port
3.0.0.0	4.0.0.1	80
3.0.0.1	4.0.0.1	80

Fig-6: Relational Algebra Operation Select on R4

UNION
R6 = R3 union R4

src_ip	dst_ip	dst_port
3.0.0.0	4.0.0.1	80
3.0.0.1	4.0.0.1	80
3.0.0.2	4.0.0.1	80
3.0.0.3	4.0.0.1	80
3.0.0.0	4.0.0.1	81
3.0.0.1	4.0.0.1	81

Fig-7: Relational Algebra Operation Union on R3,R4

INTERSECTION
R7 = R3 intersect R4

src_ip	dst_ip	dst_port
3.0.0.0	4.0.0.1	80
3.0.0.1	4.0.0.1	80

Fig- 8: Relational Algebra Operation Intersection on R3,R4

DIFFERENCE
R8 = R3 difference R4

src_ip	dst_ip	dst_port
3.0.0.2	4.0.0.1	80
3.0.0.3	4.0.0.1	80

Fig -9: Relational Algebra Operation Difference on R3,R4

3. IREWALLPOLICY ANOMALY DETECTION

In this section, we define and classify the types of anomaly using Raining 2D-Box Model. *Rule-x* is defined as a rule number *x* from the rule-set table. Therefore, if $x < y$, then *Rule-y* follows *Rule-x*. Rx is a relation that has been mapped

from *Rule-x* using PROJECT operation to exclude the *action* attribute.

Ehab S. Al-Shaer et.al [1] classified anomaly into 4 types. They are Shadowing anomaly, Correlation anomaly, Generalization anomaly and Redundancy anomaly. We also classify anomaly into 4 types but considering more than two

rules at the same time. We also present many definitions and that will be used to discover anomalies. The formal proposed definitions and algorithm can be found in publication [5]. There are 6 theorems that come out as the result of analyzing in these 4 anomalies types. More detail about proved of theorem 1-6 is presented in [6].

3.1 Shadowing Anomaly

Shadowed rule is a rule which will never be executed because all the packets that matched this rule are already matched by the one or more rules that are written above in the rule-set. For example, *Rule-4* is shadowed because R4 – (R3 U R2 U R1) = Φ. No packet can be passed to *Rule-4* (see Fig. 1 or Fig. 3(a)). In Ehab S. Al-Shaer [1] research, shadowed rule

is defined as a rule that is shadowed by only one previous rule which has different action. In this example, it would be *rule-11* which shadowed by *rule-10*. Thus their algorithm cannot classify that *rule-4* is shadowed. But in fact, *rule-4* is shadowed by two rules (*rule-2* and *rule-3*).

We represent anomalies as a two-dimension box that contains relations that are mapped from rules in the order described in the rule-set. A rectangular is used to represent relation of the rule and specified action within each box. If action is not specified in the rectangular, it can be any actions (accept or deny). This model simulates packets that fall from the top to the bottom like raining. For example, when the part of the relation in the box is not wet, means that it is shadowed, as shown in Fig-10 (a). When shadowed rules are discovered, we should remove them to reduce the size of rule-set.

Theorem 1: The firewall policy does not change even if we remove Rule-x, when Rule-x is shadowed.

3.2 Correlation Anomaly

Two rules (with different actions) are correlated if the first rule in order matches some packets that matched the second rule and the second rule matches some packets that match the first rule. This definition is similar to [1]. But in their Definition, two rules (*Rule-x* and *Rule-y*) are correlated if some fields in *Rule-x* are sub-sets or equal to the corresponding field in *Rule-y*, and the rest of the fields in *Rule-x* are super-sets of the corresponding fields in *Rule-y*. By

using their definition, *Rule-5* and *Rule-6* are considered to be correlated.

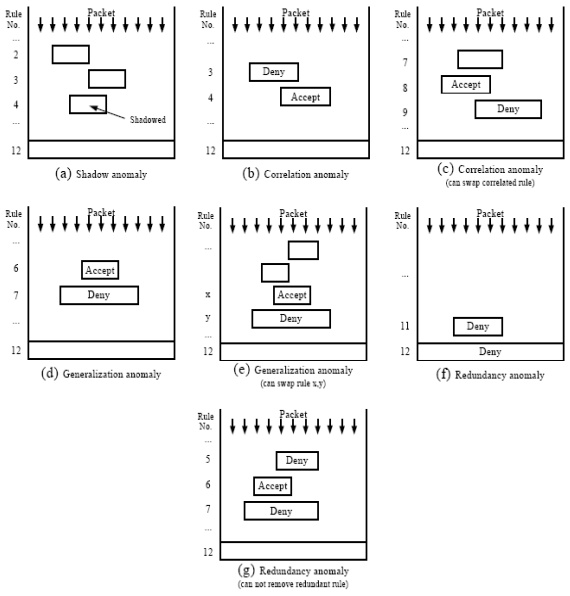

Fig- 10: Raining 2D-Box Model

Another example of correlation anomaly is *Rule-3* and *Rule-4* (see Fig-1and Fig-10(b)). However, research [1] will not be discovered by their algorithm because both *Rule-3* and *Rule-4* have one attribute (*dst_port*) that is partially correlated, i.e this field is not the subset. There are many firewall products available in the market that allows this kind of correlation to be occurred, such as when using port-range attribute (field) in IPTABLES and multi-address in Check Point Firewall-1. Therefore using our definition, we can discover this kind of correlation.

However, even though *Rule-x* and *Rule-y* are correlated, we may swap the order of these two rules without policy-change in some cases, for example, *Rule-8* and *Rule-9*. The reason for this example comes from the fact that correlated part between *Rule-8* and *Rule-9* is shadowed by previous rules (in this case *Rule-7*). It can also be explained using Raining 2D-Box model in Fig-10(c). There are 6 theorems (theorem 1-6) that are proved in [5] and [6]. In this, we proof 7-8.

Theorem 2: The firewall policy does not change even if we swap Rule-x and Rule-y, when Rule-x and Rule-y are consecutively non-correlated.

Theorem 3: The firewall policy does not change even if we swap Rule-x and Rule-y, where x < y, and Rule-x is consecutively non-correlated downward to Rule-y, and Rule-y is consecutively non-correlated upward to Rule-x.

Theorem 4: The firewall policy does not change even if we swap Rule-x and Rule-y, where Rule-x and Rule-y are correlated and Rule-xy is shadowed, and Rule-x is consecutively non-correlated downward to Rule-(y-1), and Rule-y is consecutively non-correlated upward to Rule-

(x+1). Note: Rule-xy is rules that unmapped from Rx ∩ Ry

3.3 Generalization Anomaly

A rule is said to be a generalization of the previous rule if it matches all the packets that matched previous rule, when actions are different. For example, *Rule-7* is a generalization of *Rule-6*, see Fig-10(d). In general, two rules that are generalized cannot be swapped. But in some cases, such as shown in Fig-10(e), when *Rule-y* is generalized to *Rule-x*, but it can be swapped because *Rule-x* is shadowed by previous rules.

3.4 Redundancy Anomaly

In general, when two rules are considered to be redundant, we should be able to delete Redundant-Rule. For example, as shown in Fig-10(f), we can remove *rule-11* without any changes in firewall policy. However, we cannot remove *Rule-5*, although it is redundant to *Rule-7* (see Fig-1 and Fig-10(g)). If we remove *Rule-5*, Firewall will accept packet came from 1.0.0.1 and destination to 2.0.0.1 at port 80 (the intersection part of R5 and R6).
By considering more than two rules at the same time, we can discover anomaly that cannot be explained by using SET approach. For example, Firewall Policy Advisor in [1] will recommend user to remove Rule-5 because it is redundant by Rule-7, which in fact it may not be removed as explained.

Theorem 5: The firewall rule does not change even if we remove Rule-x from the Rule List when Rule-x is consecutively redundant by Rule-y.

Theorem 6: The firewall rule does not change even if we remove Rule-x from the Rule List when Rule-x is redundant by Rule-y, and Rule-x is consecutively non-correlated downward to Rule-(y-1).

4. ANOMALY REMOVING AND RULES COMBINATION

As explained in section 3, we should remove rules that are shadowed, and rules that are redundant (with some exceptions).Also we should alert administrator when generalization anomaly or correlation anomaly are discovered.
Removing anomalies (shadowing and redundancy), and "rules combination" method (will be explained below) can shorten the size of rule-set and make firewall policy easier to understand. Reordering the rules in the rule-set may also help administrator understand the rule-set easier. It also can increase the performance of the firewall because the rules that are matched by many packets are on the top in the rule set. We can combine many rules together by using the UNION operation to the Relations. We present theorem 7 to describe the rules combination.

Theorem 7: Rule-x and Rule-y can be combined to Rule-z, where Rz = Rx U Ry, and actions are same, and y=x+1, without changing the policy.

For proving, we will guide reader to understand something about Relation (which mapped from Rule) in 2D-Box Model.

If p is the set of packets which falling to Rule-1, we found that $p \in R1$ (See 2D-Box-Model)

If p is the set of packets which falling to Rule-2, we found that $p \in R2 - R1$

If p is the set of packets which falling to Rule-3, we found that $p \in R3 - R2 - R1$

If p is the set of packets which falling to Rule-i, we found that $p \in Ri - Ri - 1 - Ri - 2 - \ldots - R1$

By using the SET Theory in Mathematics, set of packets which falling to Rule-i is $p \in Ri - (Ri - 1 \cup Ri - 2 \cup \ldots \cup R1)$

Proof of Theorem 7:

Although we swap Rule-x with Rule-y or remove each rule, packets which falling to any rules above Rule-x and Rule-y
will not changes. Thus we consider 2 cases according.
(1) Packets which falling to Rule-x or Rule-y
(2) Packets which falling to rules below Rule-y

Define $A = Rx - 1 \cup Rx - 2 \cup \ldots \cup R1$
$Rz = Rx \cup Ry$
Consider (1)
Packets which falling to Rule-x are in set p, *whenever* $p \in Rx - A$.
Because of y = x+1, thus packet which falling to Rule-y are in $p \in Ry - Rx - A$
Thus packets that fall to Rule-x or Rule-y are

$$p \in (Rx - A) \cup (Ry - Rx - A)$$
$$p \in (Rx \cup (Ry - Rx)) - A$$
$$p \in (Rx \cup Ry) - A$$
$$p \in Rz - A$$

From equation, we found that p equal to packets which falling to Rule-z.

Because of Rule-x and Rule-y are same action, thus after combine rules packets which ever match Rule-x or Rule-y will
be change to match Rule-z and decide with old Action.
Consider (2)
Define *S* is Sample Space of all packet.
Before we combine Rule-x with Rule-y, packets which falling pass Rule-y to another rules below (packets which not match Rule-y and rules above) are in set *p*

$$p \in S - Ry - Rx - A$$
$$p \in S - (Rx \cup Ry) - A$$
$$p \in S - Rz - A$$

From equation, *p* is set of packets are can fall pass Rule-z to another rules below. (After combine rules) From (1) and (2) we can combine rules without policy change.
For example, in Fig-11, by using theorem 7 above, rule can be combined one by one without changing the policy. The combined rule is the union of R1, R2, R3, and R4. In fact, it is the rule summarization of Rule-1 to Rule-4. Fig-12 shows the rule combination using Raining 2D-Box model.

Before

Order	src_ip	dst_ip	dst_port	Action
1	5.0.0.0/26	6.0.0.0/24	80	accept
2	5.0.0.64/26	6.0.0.0/24	80	accept
3	5.0.0.128/26	6.0.0.0/24	80	accept
4	5.0.0.192/26	6.0.0.0/24	80	accept

After

Order	src_ip	dst_ip	dst_port	Action
1	5.0.0.0/24	6.0.0.0/24	80	accept

Fig-11: Rules Combination (1)

Although the Theorem 7 can works only on consecutive rules, it is used as the basis of other combinations. The rules may not be consecutively ordered in the rule-set. So we need to re-order them, but we also need to ensure that it will not change the firewall policy using proposed algorithm in [5]. Fig-13(a) shows an example of the rule-set. After we move *rule-3* to the top and swapping *rule-5* with *rule-6*, firewall policy does not change, as shown in Fig-13(b). Then *rules 2-5* are then combined using Theorem 7, as shown in Fig -13(c).

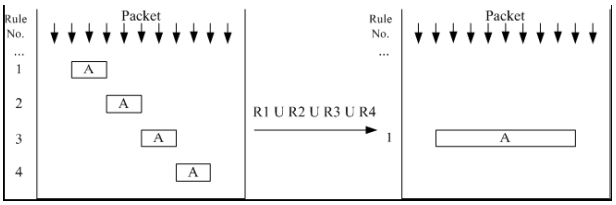

Fig-12: Rules Combination (2)

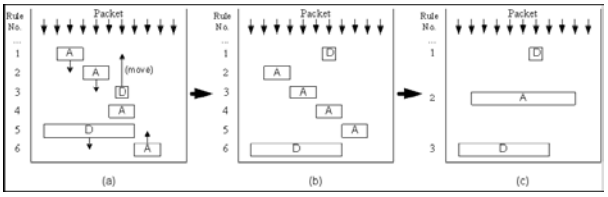

Fig-13: Rules Combination (3)

"Consecutively redundant rule" [5] is a rule that is redundant to the next rule in the rule-set. There is a theorem and definition in [5], showing that policy will not be changed if we remove consecutively redundant rule. Reverse to this theorem, we create theorem 8 below to show that inserting consecutively redundant rule does not change the policy as well. This theorem can be used to combine the rules. For example in Fig-14(a), a new rule (rule-6 in Fig-14(b)) can be inserted, without changing the policy. Then we can use the same techniques to move rules up or down, as shown in Fig-14(c). The result after we combined rules 1-4 using theorem 7 is shown in Fig-14(d).

Theorem 8: The firewall policy does not change even if we insert consecutively redundant rule.
Proof of Theorem 8:
Before insert consecutive rule, the (old) rule order is x.
After we insert consecutive rule above line number x, order of the old rule is y; y = x+1 (see fig-14(a) and 14(b))

Define $A = Rx - 1 \cup Rx - 2 \cup ... \cup R1$

Because of Rule-x is Consecutive Redundant of Rule-y, thus

$$Rx \subset Ry$$
$$y = x + 1$$

Before we insert Consecutive Redundant rule, all packets which falling to Rule-y are in set *p*

$$p \in Ry - A \quad \text{.....(a)}$$

After we insert Consecutive Redundant rule, all packets which falling to Rule-y are in set *p*

$$p \in Ry - Rx - A$$
$$p \in Ry - (Ry \cap Rx) - A$$
$$p \in (Ry - A) - ((Ry \cap Rx) - A) \quad \text{.....(b)}$$

From equation (a) and (b), Packets that falling to Rule-y has loss as $p \in (Ry \cap Rx) - A$
Before we insert Consecutive redundant rule (Rule-x), all packets which falling to Rule-x are in set *p*

$$p \in \varphi \quad \text{.....(c)}$$

(*p* is blank set because no have Rule-x in first time)
After we insert Consecutive redundant rule (Rule-x), all packets which falling to Rule-x are in set *p*

$$p \in Rx - A$$

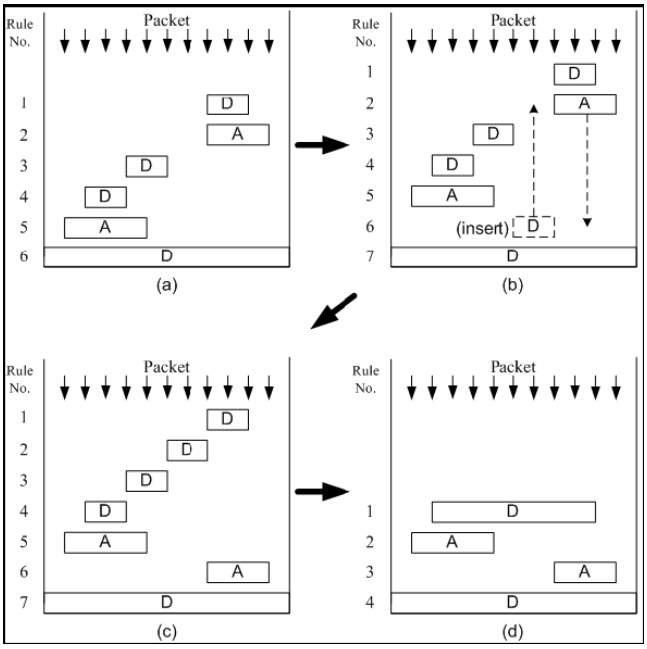

Fig-14: Rules Combination (4)

Because of $Rx \subset Ry$, Thus $p \in (Ry \cap Rx) - A$ ……..(d)
From equation (c) and (d) we find that packets which falling to Rule-x is increase as $p \in (Ry \cap Rx) - A$
After insert; $p \in (Ry \cap Rx) - A$ that ever fall to Rule-y will fall to Rule-x.
Rule-x and Rule-y are same action, thus not have any change of policy.

5. CONCLUSION

In this paper, we proposed a technique that can discover anomalies that are occurred in the firewall rule-set. It can analyze firewall rule-set using Relational Algebra technique and Raining 2DBox Model. We also present theorems (and proof of theorem) to remove and combine rules to minimize the size of firewall rules without changing the policy. Many related works are either complex, or cannot be used to find out the anomalies presented in this paper. This technique can help administrator to analyze firewall rule-set on many commercial and open-source firewall products such as Checkpoint Firewall-1, Cisco Access Control List, IPCHAINS and IPTABLES.

REFERENCES

[1] Ehab Al-Shaer and Hazem Hamed. "Firewall Policy Advisor for anomaly Detection and Rule Editing". IEEE/IFIP Integrated Management IM'2003, March 2003.
[2] P. Eronen and J. Zitting. "An Expert System for Analyzing Firewall Rules". Proceedings of 6th Nordic Workshop on Secure IT-Systems (NordSec 2001), November 2001.
[3] S. Hazelhusrt. "Algorithms for Analyzing Firewall and Router Access Lists". Technical Report TR-WitsCS-1999 Department of Computer Science, University of the Witwatersrand, July1999.
[4] Abraham Silberschatz. Henry F. Korth, Sudharsan S. "Database System Concepts, 3rd Edition". Tata McGraw-Hill, 1997.
[5] Chotipat Pornavalai and Thawatchai Chomsiri. "Firewall Policy Analyzing by Relational Algebra". The 2004 International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2004), JULY 2004.
[6] Chotipat Pornavalai and Thawatchai Chomsiri. "Firewall Policy Analyzing by Relational Algebra". Draft Technical Report, Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Thailand, January 2004.