Fast and Scalable Method for Resolving Anomalies in Firewall Policies

Hassan Gobjua
Verizon

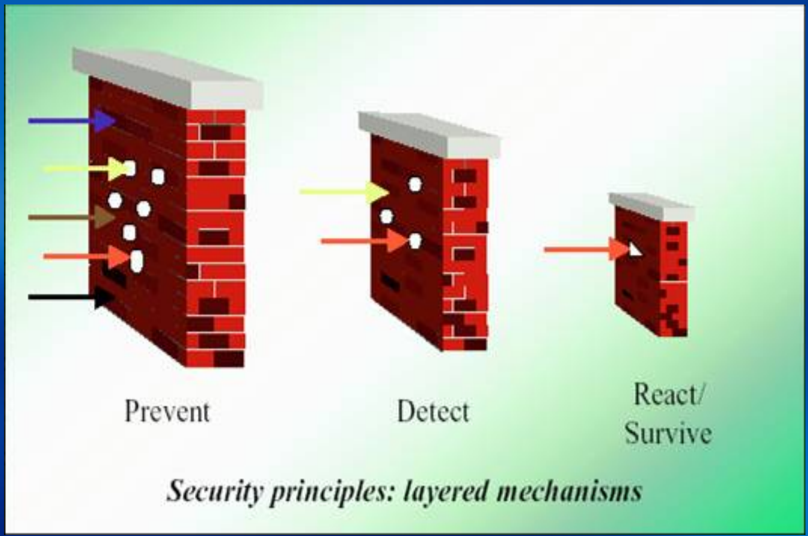
Kamal Ahmat
City University of New York

Introduction

- Firewalls
- Types of Anomalies
- Related Work
- Data Structure and Algorithm
- Experimental Results
- Conclusion

Firewalls

- Firewall
  - System acting as an interface of a network to one or more external networks.
  - Implements the security policy of the network
    - By deciding which packets to let through
      - Based on rules defined by the network administrator.



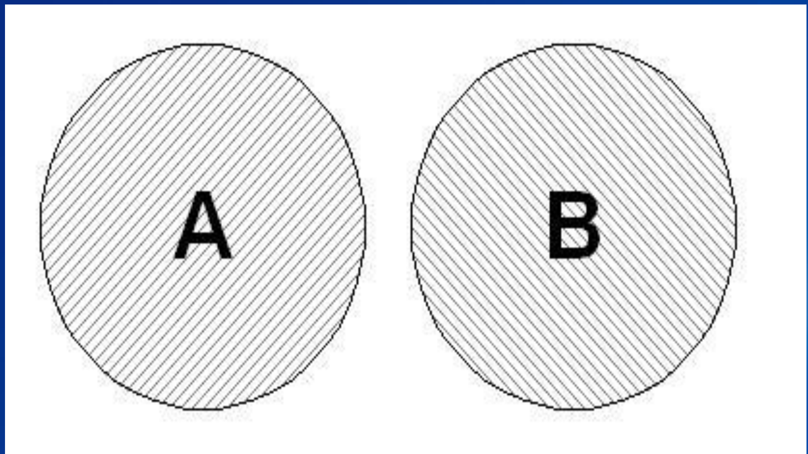
Example

Id	Protocol	Source IP address	Source Port	Destination IP address	Destination Port	Action	Probability
r_1	TCP	71.123.10.*	any	10.0.0.1	21	pemit	0.3
r_2	TCP	*.*.*.*	any	10.0.0.1	21-23	deny	0.25
r_3	TCP	71.*.*.*	any	10.0.0.1	21	pemit	0.15
r_4	TCP	71.123.*.*	any	10.0.0.1	21	deny	0.1
r_5	TCP	*.*.*.*	any	10.0.0.1	23-25	pemit	0.1

Protection Methods

- Firewalls – Firewall policy rules should be designed carefully!
- Challenges
  - Rules are created by multiple people
  - Rules are created over extended period of time
  - Number of rules in a firewall policy can be 5K+!
  - Rules are dynamic!

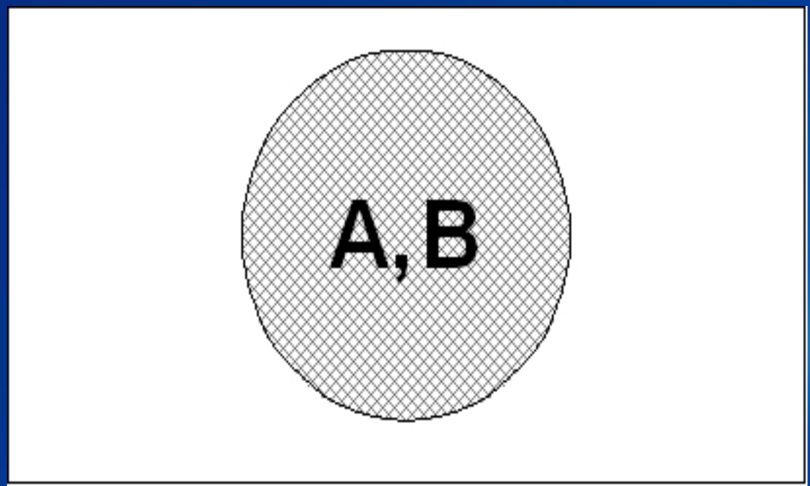
Relationships Between Rules - Disjoint Rules



- Two rules r and s are disjoint if they have at least one criterion for which they have completely disjoint values

- Example:
  - <IN, TCP, 64.233.179.104, 80, ***192.168.20.****, ANY, ACCEPT>
  - <IN, TCP, 64.233.179.104, 80, ***172.16.20.****, ANY, REJECT>

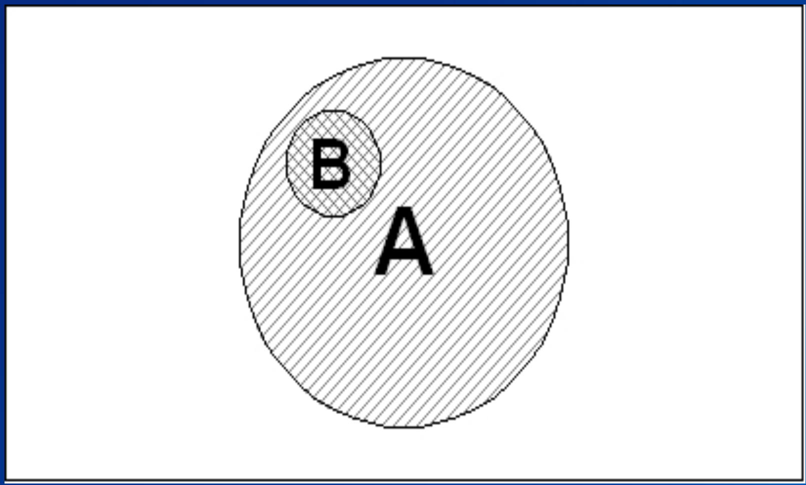
Relationships Between Rules - Exactly Matching



- Two rules r and s are exactly matched if each criterion of the rules match exactly.

- Example:
  - <IN, TCP, 64.233.179.104, 80, 192.168.20.*, ANY, ACCEPT>
  - <IN, TCP, 64.233.179.104, 80, 192.168.20.*, ANY, ACCEPT>

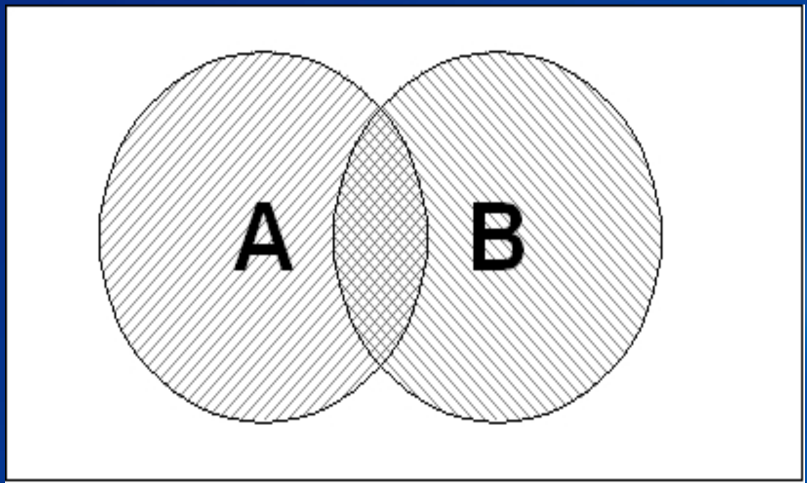
Relationships Between Rules - Inclusively Matching (Shadowing)



- Two rules r is a subset, or inclusively matched of another rule *s* if there exists at least one criterion for which *r*'s value is a subset of *s*'s value and for the rest of the attributes *r*'s value is equal to *s*'s values.

- Example:
  - <IN, TCP, 64.233.179.104, ***80***, ***192.168.20.3***, ANY, ACCEPT>
  - <IN, TCP, 64.233.179.104, ***ANY***, ***192.168.20.****, ANY, ACCEPT>

Relationships Between Rules - Correlated



- Two rules *r* and *s* are correlated if *r* and *s* are not disjoint, but neither is the subset of the other.

- Example:
  - <IN, TCP, 64.233.179.104, ***ANY***, ***192.168.20.3***, ANY, ACCEPT>
  - <IN, TCP, 64.233.179.104, ***80***, ***192.168.20.****, ANY, REJECT>

Existing Work

- E. W. Fulp – O(n^3) algorithm to order rules in a given policy; it doesn't discover correlated ones.
- E. Al-Saher *et al*. – Method for selecting rules based on their probability.
- A. Liu – Method to discover and remove redundant rules (Exact matching).

Our Approach

- We aim at removing few troublesome rules from given policy to resolve anomalies.
- Design a data structure to represent dependencies among rules.
- Remove troublesome rules.
- Return a subset of consistent rules and correlated rules (for editing).

Our Approach

- Design a data structure to represent dependencies among rules.
- Graph D is directed, and U is undirected.
  - Each node in U represents a rule
  - Two nodes are connected in U if there is *shadowing* or *correlation* relationship between these two rules.
- Graph D describes dependency among rules.

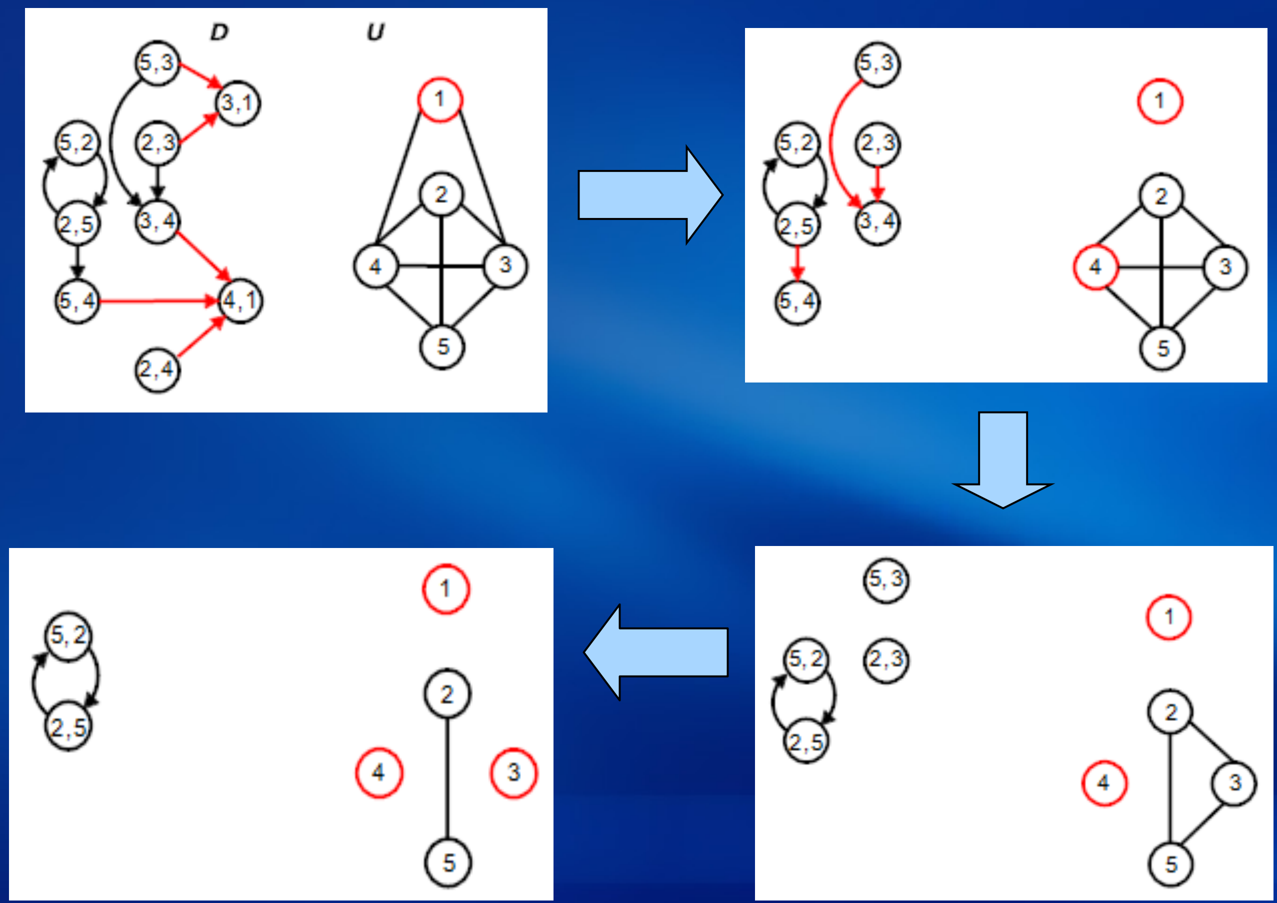
Our Approach

- Select a rule that doesn't depend on any other rule (terminal node) from D.
- Remove corresponding links from U and links/nodes from D.
- If graph U is disconnected and new component formed, continue, else there is correlation
- If there is correlation, choose the rule with highest probability.

Example

Id	Protocol	Source IP address	Source Port	Destination IP address	Destination Port	Action	Probability
r_1	TCP	71.123.10.*	any	10.0.0.1	21	permit	0.3
r_2	TCP	*.*.*.*	any	10.0.0.1	21-23	deny	0.25
r_3	TCP	71.*.*.*	any	10.0.0.1	21	permit	0.15
r_4	TCP	71.123.*.*	any	10.0.0.1	21	deny	0.1
r_5	TCP	*.*.*.*	any	10.0.0.1	23-25	permit	0.1

Example – Our Approach



Complexity

- O(n^2) to construct graphs *D* and *U*
- O(2log *n*) to discover dependencies
- Algorithm complexity O(*n*^2 log *n*)

Experimental Results

- Two sets of test experiments executed:
  - Real-life tests: five policies of size 107, 361, 647, 881, and 1385 over a month period on Verizon firewall using the original (non-improved) approach.
  - Tests done over the same period using improved approach.
- Five test sets have been executed on synthetic policies of sizes 10K – 30K.

Experimental Results – Real-Life Policies

Test	No. of Rules	Avg. Base Comp.	Avg. Correlation	Avg. Dependency	Imp. Ratio
1	107	43.1	2.7	8.5	63.3%
2	361	87.2	1.4	2.2	47.2%
3	647	381.1	3.1	7.9	62.7%
4	881	341.6	3.3	6.4	71.2%
5	1385	715.3	3.8	6.7	74.8%

Experimental Results – Synthetic Policies

Test	No. of Rules	Avg. Base Comp.	Avg. Correlation	Avg. Dependency	Imp. Ratio
1	10K	4224	121.3	13.5	68.6 %
2	12.5K	5584	389.6	11.6	40.5 %
3	15K	8054	274.7	12.0	76.4 %
4	25.5K	14263	649.2	15.2	79.3 %
5	30K	17714	712.4	20.7	87.6 %

Current & Future Work

- Find exact minimum number of rules to eliminate all anomalies from policy.
- Modify algorithm to handle dynamic-policies.
- Improve the algorithm performance.

Thank You All!

Questions?