# A Framework for the Application of Association Rule Mining in Large Intrusion Detection Infrastructures

James J. Treinen(1) and Ramakrishna Thurimella(2)
1 IBM Global Services, Boulder, CO 80301, USA jamestr@us.ibm.com
2 University of Denver, Denver, CO 80208, USA ramki@cs.du.edu

# Team Members


- Dishank Kalra(145)
- Kanishya Mohan(005)
- Shefali(207)
- Riona Chakrabarti(063)

# INTRODUCTION

**PROBLEM -** Is alarm false or real ?

**SOLUTION -** Make some rules which identify attack patterns

**PROBLEM -** Above solution is error prone because it may be possible that some new attack is not present in rule base.

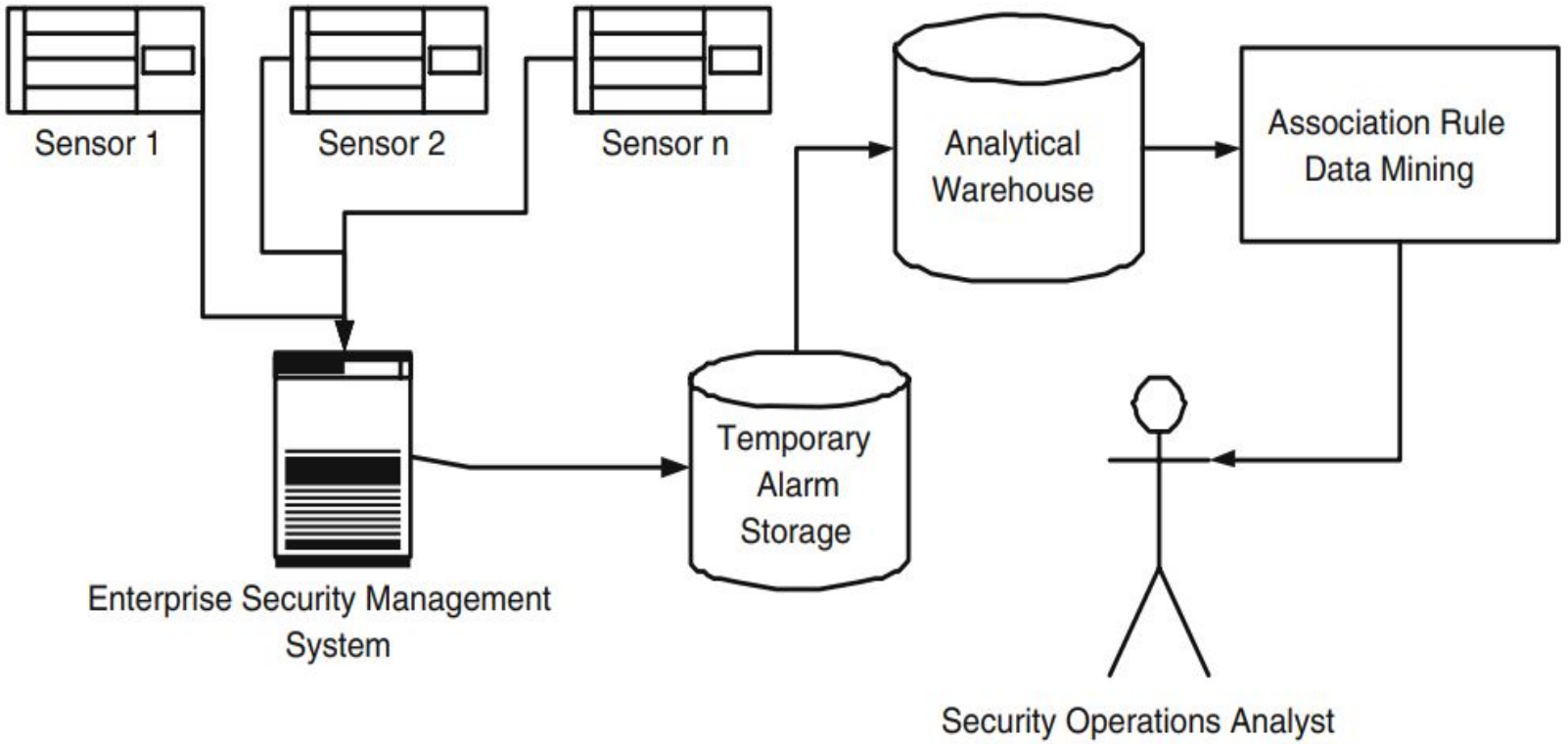
**SOLUTION BY AUTHOR -** Using association rule of mining to reduce time between detection of a new attack and adding definition of that attack in the rule base.


**Source**:https://sonitrolsecurity.com/support-verified-alarm-laws-to-lower-the-98-national-false-alarm-rate/

# EFFECT OF INACCURATE INTRUSION DETECTION SENSORS

- Reduction in the level of vigilant monitoring by security operations staff.
- Using operations staff to examine all of the alarms produced in a day can make the deployment of a typical IDS system extremely expensive.
- Situation is worsened in large monitoring infrastructures when the number of managed sensors are very high, generating millions of false alerts per day.

# Experimental Environment




Source: Research Paper

# Data Mining Terminology

Association rule mining - It locates non-obvious interrelationships between members of a large data set .

Analysis - Finding associations between the various attack and IP addresses.

These associations constitutes true attacks on the network, and then we capture them as rules in the ESM so that the SOC can easily detect future attack.

$$[x][y] \rightarrow [z]$$

$$Support = 50$$

$$Confidence = 80$$

The Confidence value states that 80 percent of the time when items x and y were found together the item z was also found.

# Data Set Reduction:

## Modeling alarms as directed graphs

Each entry in the data warehouse includes both the source IP address and destination IP address for which the alarm was raised.

A directed graph G = (V,E) with each IP address as a vertex ,and a detected alarm as an edge.

Table 1. Typical Intrusion Detection Alarms

| Network ID | Source IP | Destination IP | Signature |
|---|---|---|---|
| Network A | 10.0.0.1 | 10.0.0.4 | Signature 1 |
| Network A | 10.0.0.2 | 10.0.0.4 | Signature 1 |
| Network A | 10.0.0.3 | 10.0.0.4 | Signature 2 |
| Network A | 10.0.0.5 | 10.0.0.7 | Signature 2 |
| Network A | 10.0.0.6 | 10.0.0.7 | Signature 2 |
| Network A | 10.0.0.7 | 10.0.0.8 | Signature 2 |
| Network A | 10.0.0.9 | 10.0.0.13 | Signature 3 |
| Network A | 10.0.0.10 | 10.0.0.13 | Signature 4 |
| Network A | 10.0.0.11 | 10.0.0.13 | Signature 5 |
| Network A | 10.0.0.12 | 10.0.0.13 | Signature 6 |

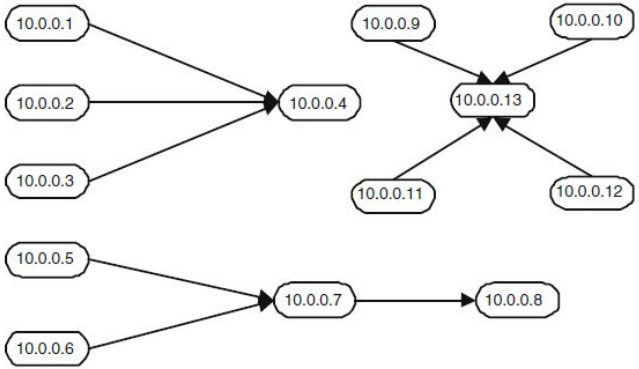



Fig. 2. Intrusion Detection Alarms as a Directed Graph With Three Connected Components

# False Positive alarms:

A **signature** is a set of rules that an IDS and an IPS use to detect typical intrusive activity.

1. Sensors scan network packets and use signatures to detect known attacks and respond with predefined actions.
2. The IDS matches a signature with a data flow, following which, actions are taken to protect the system

**Table 2.** Intrusion Detection Alarms for a Multi-Stage Attack

| Network ID | Source IP | Destination IP | Signature |
|---|---|---|---|
| Network A | 10.0.0.5 | 10.0.0.7 | Reconnaissance |
| Network A | 10.0.0.5 | 10.0.0.7 | Exploit 1 |
| Network A | 10.0.0.7 | 10.0.0.8 | Exploit 2 |
| Network A | 10.0.0.6 | 10.0.0.7 | False Alarm |

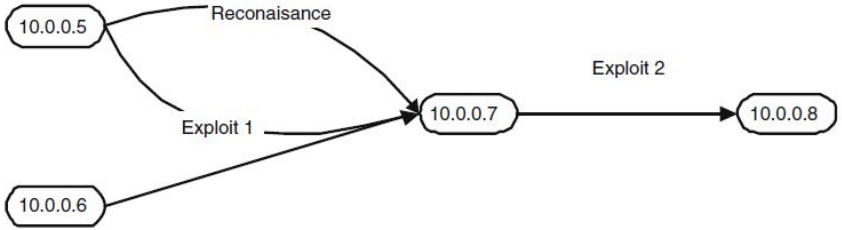
Signature-based intrusion detection generate false positives by misinterpreting normal network activity as malicious. It is beneficial to trim away irrelevant data before starting the mining activities

# Using the Connected-Component Algorithm

The alarm logs are represented as directed graphs and graph algorithms are used to limit the scope of our inquiry.

*Assumption: We have a priori knowledge of a signature for which we wished to discover new rules.*

This reduces the amount of data by **30 percent**.




*Alternative:*
Limiting the alarms to those which were produced by a source IP address that also produced the signature undergoing analysis ✖

This method would imply that we are interested in the detection of single-source attacks for a specific signature and limit any further analysis that we wished to perform on the set of alarms


Their experiments were conducted on the set of alarm logs over a 24-hour period for 135 distinct production networks.

The alarms were loaded into a data warehouse

The authors  repeated the experiments on a daily basis for 30 days.

# Generation of signature specific rules

**The objective:**

The authors' first set of experiments were conducted with the goal of discovering new rules for a signature which was thought to be exhibiting suspicious behavior.

**Algorithm: Find-Signature-Rules(G,s)**

Require: G = (V,E), a directed graph of IDS Alarms, s a subject signature
1: C ←Connected-Components(G)
2: **for** all C' ∈ C do
3: **if** s ∈ C' then
4: copy all alarms in C to T
5: **end if**
6: **end for**
7: R ← Association-Rules(T)
8: Return R

- In this method the success rate is **low.**
- Over the course of our experiments, we were able to successfully generate rules for specific signatures roughly **10%** of the time.
- Approximately half of the experiments uncovered patterns involving signatures other than those which were the original subject of our exploration.
- If the process of filtering through connected components was not satisfied then the.
  - Large number of rules(even upto 8000)
  - time to generate them
  - load that went to the staff
- To solve this we adjust the values of support and confidence for the mining algorithm.
- It was suggested to set the Support value low and Confidence value high.

# Generation of Single Source Rules

- The authors framework generated the greatest number of high Confidence rules when they grouped the transactions in the database on the basis of **source IP address.**
- This information is used to limit the data set before executing the association rules algorithm.
- When performing single-source analysis, they also found that setting the minimum values for the Support and Confidence parameters to 0 was useful.
- They were looking for correlations between signatures which were generated by a single source, it was obvious that no rules would be generated for these IP addresses.

# Web Server Attack

| Network ID | Source IP | Destination IP | Signature |
|---|---|---|---|
| Network A | 24.9.61.170 | 192.168.2.4 | AWStats configdir Command Exec |
| Network A | 24.9.61.170 | 192.168.2.5 | XMLRPC PHP Command Execution |
| Network B | 24.9.61.170 | 192.168.2.16 | AWStats configdir Command Exec |
| Network B | 24.9.61.170 | 192.168.2.17 | XMLRPC PHP Command Execution |
| ... | ... | ... | ... |

| Rule for Multi-Stage Web Server Attack |
|---|
| [AWStats configdir Command Exec]⇒ [XMLRPC PHP Command Execution] |
| Confidence = 100 |
| Support = 3.45 |

- The first stage of the attack appeared in the alarm logs as multiple instances of the signature, [AWStats configdir Command Exec], which fired as the attacker attempted to execute an unauthorized command using the configdir variable of the awstats.pl CGI script.
- In the second phase of the attack the signature, [XML RPC PHP command Execution], was triggered as attempts were made to exploit an XMLRPC vulnerability via SQL injection

# Rule for reconnaissance activity

- A pattern of two TCP based reconnaissance signatures followed by a LANMan share enumeration was found in the alarm logs for the customer under study.

| Rule for Reconnaissance Activity |
| --- |
| [TCP Port Scan][TCP Probe HTTP ]⇒ [LANMan share enum] |
| Confidence = 66.66 |
| Support = 1.7 |

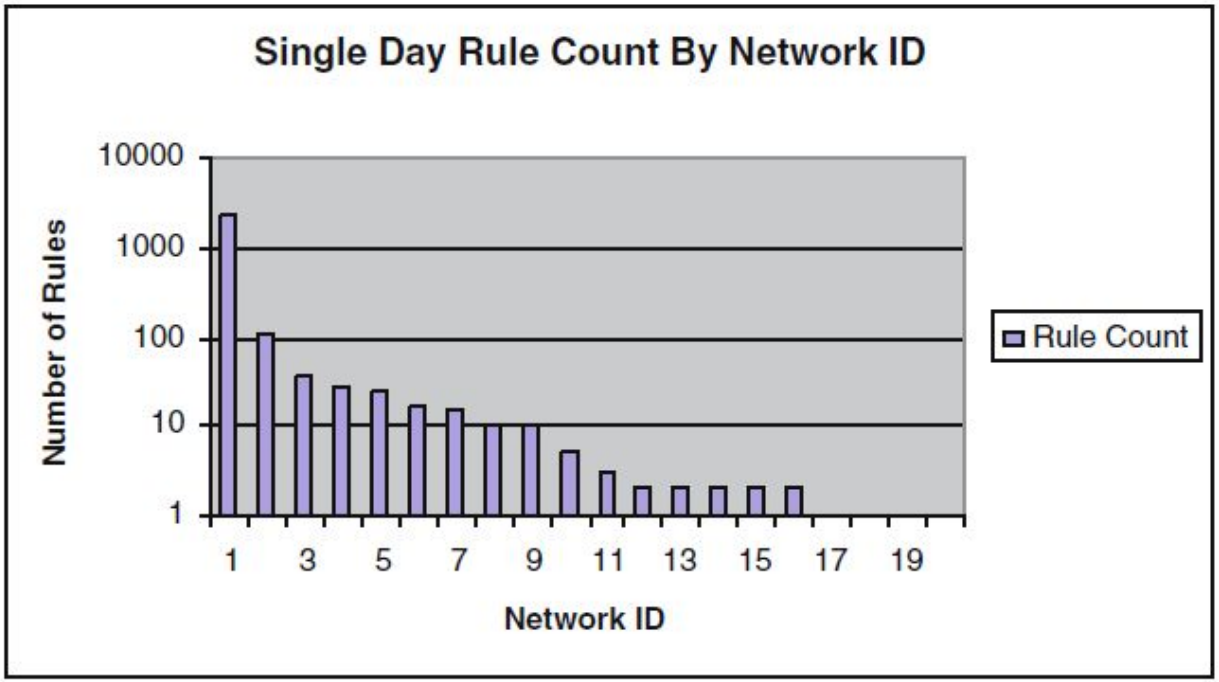
# Worm Related Rules

| Rule for Black/Nyxem Worm |
| --- |
| [NETBIOS SMB–DS IPC unicode share access][ICMP L3retriever Ping]⇒ [NETBIOS SMB–DS Session Setup And request unicode username overflow attempt] Confidence = 100 Support = 41 |

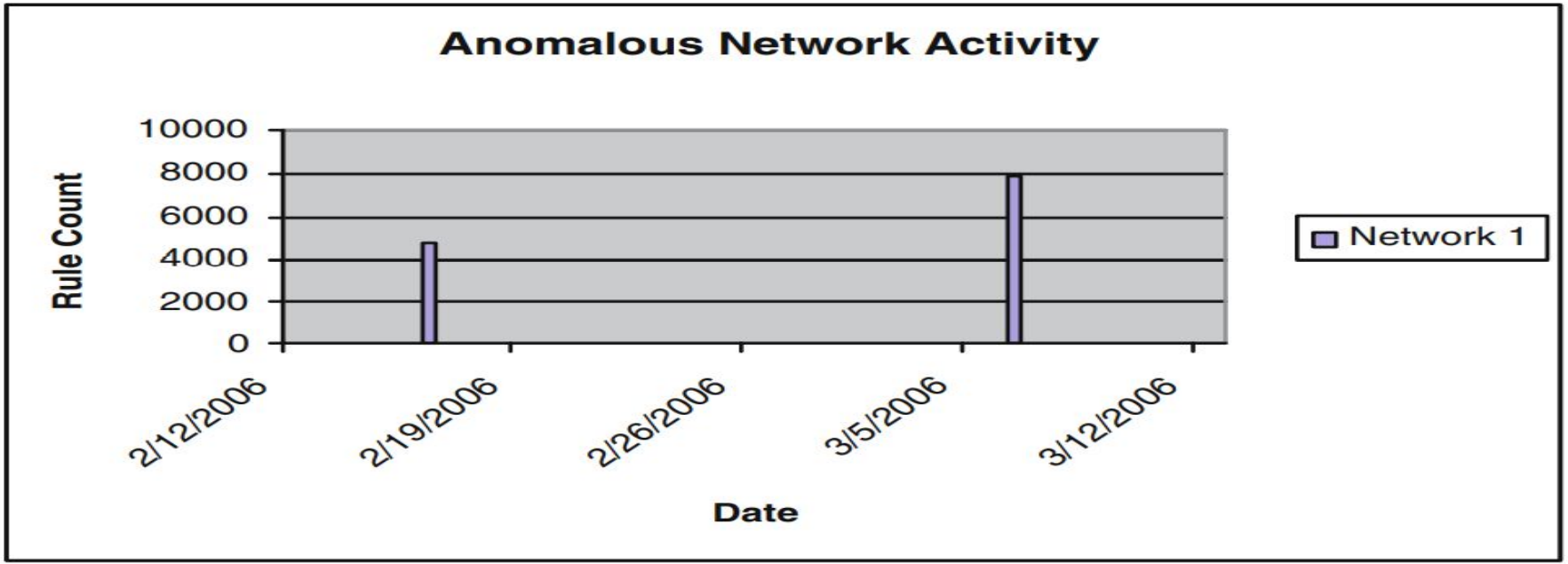
| Rule for SQL Slammer Worm |
| --- |
| [MS-SQL version overflow attempt]⇒ [MS-SQL Worm Propagation attempt] Confidence = 100 Support = 35 |

- Black Worm  is an Internet worm discovered on January 20, 2006 that infects several versions of Microsoft Windows
- SQL Slammer is a computer worm that causes a denial of service on some Internet hosts and generates a damaging level of network traffic with very high speed.

# Anomalous Network Activity as Shown by a Count of Rules Produced Per Network for a Selected Day




This graph shows a typical count of rules generated per monitored network in a 24 hour period on a logarithmic scale




**Anomaly:** There was sudden spike in alarms on 2 days in a month.

**Reason:** System misconfigured IP route between a web application server and client server. Every time that a user attempted to authenticate to the application, The intrusion detection sensors interpreted this as a spoofed source IP address, which resulted in a flood of the corresponding alarms to the security SOC.

# CONCLUSION

| WITHOUT OUR FRAMEWORK | WITH OUR FRAMEWORK |
|---|---|
| **Not reliable** because new attacks may not be present in rule base. | **Reliable** because new attacks are continuously added in rule base. |
| **Vulnerable** to new attacks because it's time consuming process to add new rule. | **More Secured** because time gap between appearance of new attack and addition in rule base is low. |
| **Increased** human error | **Reduction** in human error |
| **Increased** labour cost | **Reduction** of labour cost |

# Thank You