Intrusion Detection: Eliminating False Alarms

MAFTIA Final Review
Andreas Wespi
Overview

- MAFTIA and Intrusion Detection
- Eliminating False Alarms
- Demonstrator of an Intrusion-Tolerant IDS
MAFTIA and Intrusion Detection
Objectives

- We are interested in finding solutions to the well-known problem of the high rate of false positive and false negative alarms generated by current IDSs.

**Intrusion Detection ⇒ MAFTIA**

- These false alarms can also be due to attacks against the IDSs themselves, therefore the need to design IDSs which are intrusion-tolerant.

**MAFTIA ⇒ Intrusion Detection**
Three main contributions by WP3

- **False negatives**
  - Method to increase detection coverage by effectively combining IDSs based on a taxonomy, a framework and a tool for analyzing the strengths and weaknesses of IDSs (D3, D10)

- **False positives**
  - Method and tool to discard false alarms by mining alarm clusters (D10)

- **Attacks against the IDS itself**
  - Methods and techniques to make an IDS intrusion-tolerant by taking advantage of, among others, techniques developed within other MAFTIA work packages (D10, D13)
Links to other work packages

- Vision: Intrusion detection is a large scale distributed application

- The ultimate goal, a distributed intrusion-tolerant IDS, requires components developed by other MAFTIA work packages, especially WP2 and WP4

- WP3 focuses on the ID issues, a prerequisite to ensure certain assumptions made within other work packages

- WP1 provides the underlying foundations for the taxonomy we have developed to assess the detection capabilities of IDSs
MAFTIA and Intrusion Detection

- System security officer (SSO)
- Service user
- Insecurity signal
- Intruder alert
- Event reports
- Error reports
- Event analysis
- Fault diagnosis (inc. intrusions, attacks and vulnerabilities)
- Fault isolation (inc. intrusions, attacks and vulnerabilities)
- System reconfiguration
- Error processing
- Fault treatment
- Component or (sub-)system
- Service
- Insecurity signal (from possible lower level)
- API
Eliminating False Alarms

Klaus Julisch
Problem statement

- Intrusion-detection systems (IDSs)
  - monitor and analyze network traffic or event logs,
  - trigger alarms when signs of attacks occur, and
  - have an operator evaluate and respond to alarms.

Practical problems of IDSs include:
- 1000s of alarms per day and false alarm rates above 95%;

Corollaries:
- Manually finding the true attacks is expensive & error prone.
- Tools are needed that automate alarm evaluation!
A similar problem statement …

How to find a needle in a haystack?
Our approach

- Idea: Learn from the past to master the future.
- Mine patterns from historical IDS alarms.
- Understand why patterns occurred (i.e. identify alarm root causes).
- Act to reduce the future alarm load (e.g. fix, block, filter, ...).

Note 1: Patterns are assumed to be persistent.

Note 2: Mining is off-line and uses historical alarms!
Comparison to the “classic" alarm handling approach

Alarm correlation tries to group alarms that pertain to the same phenomenon.

- Alarm correlation versus mining of actionable knowledge
- Real-time (& resource constraint) versus off-line.
- Reconstructing attack scenarios versus finding persistent and mostly benign patterns.
- Generic versus custom-made (i.e. site- & IDS specific).
Data mining technique requirements

- To be of value in our framework, a prospective data mining (DM) technique should be:
  - Scalable to up to several millions of alarms.
  - Noise tolerant (alarms can come "out of the blue").
  - Multi attribute type (categorical, numerical, time, string, ...).
  - Easy to use for non-DM experts.
  - Yielding interpretable & relevant patterns.

- Key question: Which DM techniques qualify?

- We investigate the suitability of episode rules and conceptual clustering.
Episode rules

- Episode rules predict the occurrence of certain alarms based on the occurrence of other alarms:

  \[ \langle A_1, \ldots, A_k \rangle \Rightarrow \langle A_{k+1}, \ldots, A_n \rangle \ [\text{supp, conf, window-width}] \]

- Useful episode rules we found:
  - Attack tools, IDS idiosyncrasies, system administration tasks

- Problems encountered:
  - Abundance of episode rules generated.
  - Difficulty to interpret episode rules.
  - Weak predictiveness, alarm reduction of only about 1%.

- Conclusion: Episode rules do not solve our problem.
Conceptual clustering

- **Clustering** organizes a data set into groups of similar objects.
- In **conceptual clustering**, objects are similar iff they possess a "simple" intentional description in some language (e.g. predicate calculus).
- Example cluster:

<table>
<thead>
<tr>
<th>Name</th>
<th>Sex</th>
<th>Age</th>
<th>Job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>male</td>
<td>41</td>
<td>chemist</td>
</tr>
<tr>
<td>Jim</td>
<td>male</td>
<td>57</td>
<td>physicist</td>
</tr>
<tr>
<td>Chuck</td>
<td>male</td>
<td>44</td>
<td>astronomer</td>
</tr>
<tr>
<td>Ted</td>
<td>male</td>
<td>52</td>
<td>chemist</td>
</tr>
</tbody>
</table>

Intentional description:
\{x \mid (x \text{ is male}) \land (x \text{ is over 40}) \land (x \text{ is scientist})\}

Advantages:
*Understandability and support for categorical attributes.*
Attribute-Oriented Induction (AOI)

- Attribute-Oriented Induction (AOI)
  - summarizes relational database tables
  - by repetitively replacing attribute values by more abstract ones,
  - which are taken from user-defined generalization hierarchies.

<table>
<thead>
<tr>
<th>Original table</th>
<th>After generalization step</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Src-IP</strong></td>
<td><strong>Dest-IP</strong></td>
</tr>
<tr>
<td>ip1</td>
<td>ip3</td>
</tr>
<tr>
<td>ip1</td>
<td>ip4</td>
</tr>
<tr>
<td>ip2</td>
<td>ip4</td>
</tr>
</tbody>
</table>

Any-IP

My-IPs

WWW

DNS

ip1

ip2

ip3

ip4
Attribute-Oriented Induction (AOI)

- Merge identical tuples and update their counts:

  **After generalization step**

<table>
<thead>
<tr>
<th>Src-IP</th>
<th>Dest-IP</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1</td>
<td>DNS</td>
<td>1</td>
</tr>
<tr>
<td>ip1</td>
<td>DNS</td>
<td>1</td>
</tr>
<tr>
<td>ip2</td>
<td>DNS</td>
<td>1</td>
</tr>
</tbody>
</table>

**After merging**

<table>
<thead>
<tr>
<th>Src-IP</th>
<th>Dest-IP</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1</td>
<td>DNS</td>
<td>2</td>
</tr>
<tr>
<td>ip2</td>
<td>DNS</td>
<td>1</td>
</tr>
</tbody>
</table>

- AOI terminates when each attribute assumes at most $d_i$ distinct values.
  - The $d_i$, $i=1, ..., n$, are user-provided.
  - Example: Be $d_{Src-IP} = d_{Dest-IP} = 1$ then terminate after generalizing Src-IP. Final result: (WWW, DNS, 3)
Algorithm

**Input**: Alarm table \( T \), Hierarchies \( H[i] \);

**Output**: Alarm clusters represented by generalized alarms;

1. **for all** alarms \( a \) in \( T \) do \( a.C := 1 \);  // Init counts
2. **while** table \( T \) is not abstract enough do {
3.  Select an alarm attribute \( A[i] \);
4.  **for all** alarms \( a \) in \( T \) do  // Generalize \( A[i] \)
6.  **while** identical alarms \( a, a' \) exist do  // Merge
7.    Set \( a.C := a.C + a'.C \) and delete \( a' \) from \( T \);
8.  }


Applying AOI to IDS alarms

- Problem: IDS alarms are skewed and noisy, which leads to over-generalization!

**Typical IDS alarm log**

<table>
<thead>
<tr>
<th>Src-IP</th>
<th>Dest-IP</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1</td>
<td>ip4</td>
<td>1000</td>
</tr>
<tr>
<td>ip1</td>
<td>ipA1</td>
<td>1</td>
</tr>
<tr>
<td>ip1</td>
<td>ipB1</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>ip1</td>
<td>ipZ1</td>
<td>1</td>
</tr>
</tbody>
</table>

...main signal

- 26 noise alarms

Dest-IP is generalized twice if $d_{Dest-IP} < 27$:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1</td>
<td>My-IPs</td>
<td>1000</td>
</tr>
<tr>
<td>ip1</td>
<td>External-IPs</td>
<td>26</td>
</tr>
</tbody>
</table>

Over-generalized!
Modifications to “classic” AOI

- **Modification 1:**
  - Abandon the \( d_i \) thresholds for the number of different attribute values.
  - Introduce \( \text{min}\_\text{size} \in \mathbb{N} \), and generalize until a cluster \( C \) has a count bigger than \( \text{min}\_\text{size} \) (i.e. \( C.\text{count} > \text{min}\_\text{size} \)).

- **Modification 2:**
  - Remove clusters of a size larger than \( \text{min}\_\text{size} \) and
  - reset the remaining alarms (i.e. undo all generalization steps).

- **Modification 3:**
  - Use heuristics to select a suitable attribute for generalization.
Sample result of modified AOI

Cluster of size 50573 alarms:

<table>
<thead>
<tr>
<th>Alarm Name</th>
<th>IIS View ASP Source Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcIP/SrcPort</td>
<td>External-IPs / Non-Privileged</td>
</tr>
<tr>
<td>DstIP/DstPort</td>
<td>10.8.17.* / 80</td>
</tr>
<tr>
<td>Context</td>
<td>Get /cgi/s?action=...www%2Euva%2E...</td>
</tr>
<tr>
<td>Time Structure</td>
<td>weekdays</td>
</tr>
</tbody>
</table>

Concepts like "External-IPs" come from the hierarchies!
Experimental evaluation

- All experiments use real-world data.

- Experimental setup:
  - Choose an IDS X and month m.
  - Cluster the alarms that IDS X triggered in month m.
  - Interpret the clusters obtained.
  - Manually derive filtering rules for the alarms in the clusters.
  - Evaluate the filtering rules on the alarms of IDS X in month m+1.
  - Measure the alarm load reduction.

- Two sets of experiments:
  - Fixed month, varying IDS.
  - Fixed IDS, varying month.
Alarm load reduction per IDS in December 2001
Alarm load reduction over the year 2001 for IDS 8

Temporary networking problem!
Summary

- Problem: IDSs tend to generate a flood of mostly false alarms.
- Suggested solution: Mine, understand, act.
  - Episode rules: Expensive to use, minor alarm reduction.
  - Conceptual clustering: Well-suited in our framework.

- More abstractly, we have shown that:
  - IDS alarms have a pronounced and persistent structure.
  - Data mining can be used to reveal this structure.
  - Knowledge of this structure is actionable, i.e., can be used to handle future alarms more efficiently.
Dissemination

- Technology deployed by IBM's Managed Security Services organization
- In plan for next release of IBM Tivoli Risk Manager