Miner Alerts Module to Generate Itemsets Based on FP-Growth Algorithm Improvement

Karim H. Al-Saedi*, Raghda Abd Al-Rab
Department of Computer Science, College of Science, Mustansiriyah University, IRAQ
*Correspondent email: karimnav6@gmail.com

Abstract
Data mining techniques becomes very useful for all areas, Which gives impressive results and accurate. It is can be works with huge data and variance type's data. The intrusion detection system (IDS) has huge numbers of alerts without classify and almost alerts be false positive. In this paper, we proposed a new miner module to generating Itemsets of IDS alerts by using FP-Growth Algorithm Improvement, which it is produce from compact Fp growth algorithm with Apriori algorithm. This new module contains three phases: Compute support, Resort, and Generating K-Itemsets. It is applied on Darpa 1999 datasets to generating Alerts sets based on IDS Snort. The obtain result was very useful because it is make the alerts ready to classify.

Keywords: Apriori Algorithm, Fp-Growth Algorithm, Data Mining, Network Security.

Introduction
Data Mining (DM) is the technique designed to select the significant information through the huge data. It is a technique for the results of a long method of study; it is utilized as synonyms to one another. DM is used to select the datum of any system through analyzing the data facts [1]. DM techniques have ability utilized to build up Intrusion Detection System. There are several important issue that contribute into an Intrusion detection application using DM [2]: deleting normal activity from alert data for focusing real attacks; Identifying false alerts and sensor signatures; Finding abnormal action that detect a real attack; and Identifying long and ongoing patterns. Various Algorithms utilizing in data mining; like Apriori, FP-Growth, Genetic, K-means Algorithms, etc., Will be explained in the following paragraphs in detail Apriori, FP-Growth algorithms which be using in the research.

Apriori Algorithm
Apriori algorithm is a classical algorithm introduced by R. Srikant and R. Agrwal in 1994 [3], for studying association rule mining. Data mining have a wide domain of usages in which Apriori employ a "bottom up" way, for which frequent subsets are extended one item at a time (a step known as candidate generation, and groups of candidates are tested against the information. There is variance algorithms have been suggested to determine frequent pattern in data mining the first proposed algorithm was named Apriori [4] [5]. This algorithm based on a hash tree structure and breadth first search to compute nominee item sets efficiently.
Figure 1 shows Apriori Algorithm Presented in [6] [7].

**Input:**
A transaction database $DB$ and a minimum support threshold  
**Output:**
Table contain $K$-item sets  
**Process:**
1. Scan DB once; find frequent 1-item set.  
2. Generate 2-item set depended on junction 1-item set  
3. If minimum support < threshold then delete item set  
4. stop when cannot generate item sets frequency descending order  
5. resort item set depended high frequent  
6. Scan DB again, construct FP-tree  
7. Traversal item set depended on the FP-Tree starting from the bottom of the header table

![Figure 1: The Apriori Algorithm.](image)

**Fp-Growth Algorithm**
Association rules are very important; as an example of these rules is the FP growth algorithm. This algorithm utilizes a prefix tree impersonation of the specific database (FP-tree). It allows the discovery of the frequent item set without generating the candidate item set. This can be achieved by doing a two-stages [8]:

**Stage 1:** Building the consolidated data structure, named the FP-tree; and  
**Stage 2:** Extracting the repeated item sets straight from the FP-tree Figure 2 shows the FP-growth algorithm growth algorithm. Figure 3 shows the phases of the FP growth algorithm.

**Input:**
A transaction database $DB$ and a minimum support threshold  
**Output:**
FP-tree, the frequent-pattern tree of $DB$  
**Process**:
1. Scan DB once; find frequent one-item set.  
2. Order frequent items in

![Figure 2: The FP-growth Algorithm](image)

**Architecture of the Proposed Mining Module**
In Figure 4, the architecture of the proposed module is illustrated. This module is designed in order to enhance the output of Fp growth and Apriori algorithms by used FAI algorithm.

**Mining Module**
This module uses Fp-growth Apriori Improvement (FAI) algorithm, which is extract from merged FP-growth algorithm with the Apriori algorithm. This algorithm helps enhance execution, reduce the time it takes, reduce the storage space, and get better results. The steps below demonstrate the way these features are utilized in this algorithm:

**Step1:** This step involves selecting the following features alerts (IPs, Ports,
IPd, Portd) with Rule File Comes feature;

**Steps 2:** This step implies generating 1-itemset;

**Step 3:** This step includes descending sort; and

**Step 4:** This step helps generate the K-item set. The FAI Algorithm is illustrated in Figure 5.

**Calculating the Support**

This sub-module helps generate one item set and compute the repeats number of each item. Table 1 shows how calculated support between (IPs and RFC).

**Algorithm:** FAI algorithm

**Process:**

1. Scan DB once; find frequent one-item set.

2. Order frequent items in frequency descending order resort item set depended high frequent

3. Generate 2-item set depended on junction 1-item set

4. If minimum support < threshold then delete item set

5. stop when cannot generate item sets

Figure 5: shows the FAI Algorithm.

**Resort**

This sub-module helps rearrange the support descending from high to low value. Table 2 shows the explained process.

### Table 1: Calculating the Support.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Items</th>
<th>TID</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.207.160.115</td>
<td>SHELLCODEx86incccxNOOP, WEBCLIENTPCREcharacterclassdoublefreeerrorflowAttempt</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>10.207.161.23</td>
<td>ICMPDestinationUnreachablePortUnreachable, ICMPPING,NETBIOSDCERPCNCACN-IP TCPwinregOpenKeyoverflowattempt, SHELLCODEx86incccxNOOP</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>10.207.160.247</td>
<td>ICMPPING,NETBIOSDCERPCNCACN-IP TCPwinregOpenKeyoverflowattempt, SHELLCODEx86incccxNOOP</td>
<td>3</td>
</tr>
</tbody>
</table>

### Table 2: Resort.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Items</th>
<th>TID</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.207.161.23</td>
<td>ICMPDestinationUnreachablePortUnreachable, ICMPPING,NETBIOSDCERPCNCACN-IP TCPwinregOpenKeyoverflowattempt, SHELLCODEx86incccxNOOP</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>10.207.160.247</td>
<td>ICMPPING,NETBIOSDCERPCNCACN-IP TCPwinregOpenKeyoverflowattempt, SHELLCODEx86incccxNOOP</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>10.207.160.115</td>
<td>SHELLCODEx86incccxNOOP, WEBCLIENTPCREcharacterclassdoublefreeerrorflowAttempt</td>
<td>2</td>
</tr>
</tbody>
</table>

**Generating K-Item Set**

This sub-module generates the K-item set whereas the 1-itemset is used to generate 2-itemset. Similarly, 2-itemsets are used to generate 3-itemsets, and so on. Such a process goes on until there is no item set left to be generated. Later on, if the support of the resulting group is greater than or equal to the minsup, the support will be used frequently. Otherwise; is the support will be rejected due to the infrequency [6]? Table (3) shows the way K-item is generated by assuming that minsup is (>=1).

**The Evaluation of the Mining Module**

In this module, two algorithms were merged, so as to get an improved algorithm that helps
generate K-items sets in a different and fast ways. When using the Apriori algorithm, the generating process of K-items will be slower whenever the data is larger. On the other hand, with FP growth algorithm, the whole process is faster as the data is larger; however, no K-items will be generated. These two algorithms helped creating a new one that is characterized by new features. Table 4 compares the Apriori and Fp growth algorithms with FAI algorithm:

Table 3: Generating K-Item Set.

<table>
<thead>
<tr>
<th>Items</th>
<th>TID</th>
<th>Sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.207.161.23, 10.207.160.247</td>
<td>ICMPPING, NETBIOSEXDCERPCNCACN-IP</td>
<td>3</td>
</tr>
<tr>
<td>10.207.161.23, 10.207.160.115</td>
<td>TCPwinregOpenKeyoverflowattempt, SHELLCODEx86inccexNOOP</td>
<td>1</td>
</tr>
<tr>
<td>10.207.160.247, 10.207.160.115</td>
<td>SHELLCODEx86inccexNOOP</td>
<td>1</td>
</tr>
<tr>
<td>10.207.161.23, 10.207.160.247, 10.207.160.115</td>
<td>SHELLCODEx86inccexNOOP</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4: compare the Apriori and Fp growth algorithms with FAI algorithm.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Apriori Algorithm</th>
<th>Fp growth Algorithm</th>
<th>FAI Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hash tree</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>flexibility</td>
<td>slow with large data and fast with few data</td>
<td>Fast with large data and slow with few data</td>
<td>fast with large and few data</td>
</tr>
<tr>
<td>Generate K item</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sort</td>
<td>high order</td>
<td>low order</td>
<td>low order</td>
</tr>
<tr>
<td>No. of features</td>
<td>Single feature</td>
<td>Single feature</td>
<td>Multi features</td>
</tr>
</tbody>
</table>

Conclusions

In the mining module, the K-item set was generated using the proposed algorithm (FAI). Such an algorithm helped gain good results, which have not been arrived at by some of the previous work. We working now on the new module to classify alerts based on the output of this module.

References

[5] Han, Jiawei, Jian Pei, and Micheline Kamber, Data mining: concepts and techniques, Elsevier, 2011.