# WiFi Miner: An Online Apriori-Infrequent Based Wireless Intrusion Detection System

Ahmedur Rahman
School of Computer Science
University of Windsor
Windsor, Ontario N9B 3P4
rahma21@uwindsor.ca

C.I. Ezeife
School of Computer Science
University of Windsor
Windsor, Ontario N9B 3P4
cezeife@uwindsor.ca

A.K. Aggarwal
School of Computer Science
University of Windsor
Windsor, Ontario N9B 3P4
akshaia@uwindsor.ca

## ABSTRACT

Intrusion detection in wireless networks has become a vital part in wireless network security systems with wide spread use of Wireless Local Area Networks (WLAN). Currently, almost all devices are Wi-Fi (Wireless Fidelity) capable and can access WLAN. This paper proposes an Intrusion Detection System, WiFi Miner, which applies an infrequent pattern association rule mining Apriori technique to wireless network packets captured through hardware sensors for purposes of real time detection of intrusive or anomalous packets. Contributions of the proposed system includes effectively adapting an efficient data mining association rule technique to important problem of intrusion detection in a wireless network environment using hardware sensors, providing a solution that eliminates the need for hard-to-obtain training data in this environment, providing increased intrusion detection rate and reduction of false alarms.

The proposed system, WiFi Miner solution approach is to find frequent and infrequent patterns on pre-processed wireless connection records using infrequent pattern finding Apriori algorithm proposed by this paper. The proposed Online Apriori-Infrequent algorithm improves the join and prune step of the traditional Apriori algorithm with a rule that avoids joining itemsets not likely to produce frequent itemsets as their results, there by improving efficiency and run times significantly. An anomaly score is assigned to each packet (record) based on whether the record has more frequent or infrequent patterns. Connection records with positive anomaly scores have more infrequent patterns than frequent patterns and are considered anomalous packets.

## Keywords

Data mining, wireless intrusion, network intrusion detection, hardware sensors, infrequent patterns, no training data

## 1. INTRODUCTION

Security of computer networks has become a very crucial issue. Traditionally, the firewall is considered as the first line of defense, but the unsophisticated firewall policy cannot meet the requirements of some organizations, which need high security. Existing data mining based intrusion detection systems include ADAM [4], MADAMID [9], MINDS [5], DHP [10], LERAD [12], ENTROPY [18], but all these systems are designed for wired network environment. In the last few years, wireless technology has advanced rapidly, providing convenience and flexibility but few studies have been done on intrusion detection of wireless networks. Data mining has been applied successfully to wired network intrusion detection since early 2000. ADAM [4] was one of the early research that featured a system applying data mining techniques to the problem of network intrusion detection, using association rule mining Apriori algorithm [8]. Other systems include MADAMID and MINDS. MADAMID [9] focused on efficiency and automation of the process of network connection feature constructions. One limitation of these systems is that some of them are currently off-line but a more effective intrusion detection system should be real time, to minimize chances of compromising network security. Another limitation in some models is that they compute only frequent patterns in connection records. However, many intrusions like those that embed all activities within a single connection, do not have frequent patterns in connection data. These types of intrusions might go undetected in these models. A limitation of MINDS [5] is that it needs training data to learn the classifier and another limitation of MINDS is that it only analyzes the header parts of data and does not pay attention to payload. As a result, U2R (User To Root) or R2U (Root To User) attacks may go undetected in their system.

Our studies show that current wireless IDSs are still dependent on training data and without prior training these systems cannot detect intrusions in real time and some wireless IDS based on Association rule mining technique [11] detect intrusions only for ad-hoc network and are not applicable for infrastructure based WLAN. This paper proposes a network intrusion detection system (WiFi Miner) for wireless environment, which uses wireless hardware sensors to capture wireless traffic, which a newly proposed real time and online Apriori-Infrequent based data-mining algorithm promptly analyzes to detect new attacks. Wireless Fidelity (WiFi) is used to represent 802.11 wireless networks capable of transmitting data over short distances. Our WiFi Miner's proposed Real-time Online Apriori-Infrequent algorithm is introducing for the first time, the technique for analyzing incoming datasets to find infrequent patterns without any

.

prior training with safe data. The proposed technique can detect new types of wireless attacks efficiently with a reduced time complexity in comparison to traditional Apriori based systems and can flag anomalous connections in real time on the fly.

**Types of Wireless Attacks**
Wireless intrusions belong to four broad categories [17], namely: (1) passive attacks, (2) active attacks, (3) man-in-the-middle attack and (4) jamming attacks. A passive attack (e.g., war driving) occurs when someone listens to (or eavesdrops) on network traffic. Armed with a wireless network adaptor that supports promiscuous mode, the eavesdropper can capture network traffic for analysis using easily available tools, such as Network Monitor in Microsoft products, or (Transmission Control Protocol) TCPdump in Linux-based products, or AirSnort in Windows or Linux. War driving is the act of searching unsecured Wi-Fi networks by a person with a Wi-Fi equipped computer. As long as somebody is sniffing the network packets and trying to discover some useful information from gathered packets (e.g., WEP key used in the network or available open ports), we classify these activities as passive attacks. Once this information is discovered through passive attacks, then hackers can launch some active attacks. Active attacks launched by hackers who access the network to launch these active attacks include unauthorized access, Denial of Service (DoS) and Flooding attacks like (SYNchronized) SYN Flood attacks, and (User Datagram Protocol) UDP Flood attacks. DoS attack attempts to engage a host of computer resources so that these resources are not available to other users. DoS is an attack in which the attacker keeps the resource too busy or too full to handle other legitimate requests, and thus, it denies legitimate users access to a machine [14]. In SYN Flood attack, the attacker sends a lot of TCP packets, where both SYN and (ACKnowledgment) ACK flags in the header are set to 1 using tools like Engage Packet Builder [16]. The attacker's IP address is fake and destination IP address is the server victim's address. Receiving so many packets from attacker prevents victim from accepting new legitimate requests and may crash the victim server. Man-in-the-middle attack entails placing a rogue AP (Access Point) within range of wireless stations. If the attacker knows the SSID in use by the network (which is easily discoverable) and the rogue AP has enough strength, wireless users have no way of knowing that they are connecting to an unauthorized AP. Because of their undetectable nature, the only defense against rogue APs is vigilance through frequent site surveys using tools such as Netstumbler and AiroPeek, and physical security. Jamming is a special kind of DoS attack specific to wireless networks. Jamming occurs when spurious RF (Radio Frequency) frequencies interfere with the operation of the wireless network. Intentional and malicious jamming occurs when an attacker analyzes the spectrum being used by wireless networks and then transmits a powerful signal to interfere with communication on the discovered frequencies. Fortunately, this kind of attack is not very common because of the expense of acquiring hardware capable of launching jamming attacks and it leads to a lot of time and effort being expended merely to disable communications.

## 1.1 Contributions and Outline

This paper proposes a wireless intrusion detection system called WiFi Miner, with the following two objectives:
1. Eliminating the need for hard-to-get training data. This it does with a proposed Online Apriori-Infrequent algorithm, which does not use the confidence value parameter and does not create any rules, but efficiently uses only frequent and non-frequent patterns in a record to compute an anomaly score for the record to determine whether this record is anomalous or not on the fly.
2. Real-Time Detection of Intrusions: This our system does by integrating proprietary hardware sensors, where streams of wireless packets (e.g., Media Access Control or MAC frames ) from Access Points (AP) are promptly captured and processed with the proposed Online Apriori-Infrequent algorithm. Our proposed Real-time Online Apriori-Infrequent algorithm improves the join and prune steps of the traditional Apriori algorithm, detects frequent and infrequent patterns in connection records, assigns anomaly scores to connection records without generating association rules from frequent patterns, and increases the efficiency and run times significantly. The proposed system targets mostly active, passive and main-in-the-middle wireless attacks, which are not easily detected by existing wired attacks.

Section 2 presents related work, Section 3 presents the proposed system: WiFi Miner, Section 4 describes the experimental results, while section 5 concludes the paper.

## 2. RELATED WORK

ADAM [4] is a wired Apriori based network intrusion detection system. First, ADAM collects normal, known frequent datasets through mining as training datasets. Secondly, during detection, it runs an on-line algorithm to find last frequent connections, which it compares with known mined training normal datasets and it discards those recent connections which seem to be normal. With suspicious records, it then uses a classifier, previously trained to classify and label suspicious connections as a known type of attack, unknown type of attack or a false alarm. The central theme of MADAMID [9] approach is to apply data mining programs to the extensively gathered audit data to compute models that accurately capture the actual behavior or patterns of intrusions and normal activities. In MADAMID they have used association and frequent episode rule for sequence analysis. Another research [12] presented an efficient algorithm called LERAD (Learning Rules for Anomaly Detection). Another important research in this field is MINDS [5], which uses a suite of data mining techniques to automatically detect attacks against computer networks and systems. In their research they presented two specific contributions: (1) an unsupervised anomaly detection technique that assigns a score to each network connection that reflects how anomalous the connection is, and (2) an association pattern analysis based module that summarizes those network connections that are ranked highly anomalous by the anomaly detection module. An Online K-means algorithm (KMO) was used in [20], where authors analyzed network traffic data streams collected and recorded from a WLAN system and detected all types of attack behaviors through data mining clustering technique. The log they used is specifically for wireless traffic and they extracted these data from several access points (APs). The main limitation of their approach was that they used training data which is hard to get. Another hybrid anomaly detection approach is proposed in [11], which uses association rule mining technique and cross fea-

**Table 1: Example Database Records**

| TID | Items |
|---|---|
| 1 | A B D |
| 2 | A C E F |
| 3 | B C D F |
| 4 | A B C D |
| 5 | A B C E |

ture mining to build normal behavior profiles of network activities for an individual node.

**Data Mining Association Mining Related Algorithms**
Association rule can be used to find correlation among items in a given transaction. A well-known example is market basket analysis, which analyzes customer buying habits by finding associations between the different items that customers place in their shopping baskets. If most customers who buy milk also buy bread, we can put milk and bread in the same shelf to increase sales and profit. Association rule mining was proposed in [8], where the formal definition of the problem is presented as: Let L = $\{i_1, \ldots, i_n\}$ be a set of literals, called items. Let database, D be a set of transaction records, where each transaction T is a set of items such that $T \subseteq L$. Associated with each transaction is a unique identifier, called its transaction id (TID). We say that a transaction T contains X, a set of some items in L, if $X \subseteq L$. An association rule is an implication of the form $X \rightarrow Y$, where $X \subseteq L$, $Y \subseteq L$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in the transaction set D with confidence c if $c$% of transactions in D that contain X also contain Y. The rule $X \rightarrow Y$ has support s in the transaction set D if $s$% of transactions in D contain $X \cup Y$. An example is shown in Table 1. Here, there are five transactions with TID 1, 2, 3, 4 and 5. Rule $\{A\} \rightarrow \{C\}$ is an association rule because with a given minimum support of 60% or 3 out of 5 transactions, the 2-itemset $AC$ which, this rule is generated from, has a support of 4/5 or 80%. The confidence for this rule is 4/4=100%.

According to [8] and [1], the problem of mining association rules can be decomposed into the following two steps:
1) Discovering the large itemsets or frequent patterns, i.e., the sets of itemsets that have transaction support above a pre-determined minimum supports.
2) Using the large itemsets (frequent patterns) to generate association rules for the database that have confidence above a pre-determined minimum confidence.

Several important association rule mining algorithms including the Apriori [8], [1], [13] and Fp-growth [7], some of which are commonly used in network intrusion detection systems, exist. The basic idea behind the Apriori algorithm [8], [1], is to level-wise, use shorter frequent k-itemsets ($L_k$) to deduce longer frequent (k+1)-itemsets ($L_{k+1}$) starting from candidate 1-itemsets consisting of single items in the set L defined above, until either no more frequent itemsets or candidate itemsets can be found. Thus, the Apriori algorithm finds frequent k-itemsets $L_k$ from the set of frequent (k-1)-itemsets $L_{k-1}$ using the following two main steps involving Joining the $L_k$ with $L_k$ Apriori-gen way to generate candidate k-itemsets $C_k$, and secondly, pruning the $C_k$ of itemsets not meeting the Apriori property or not having all their subsets frequent in previous large itemsets. To obtain the next frequent $L_k$ from candidate $C_k$, the database has to be scanned for support counts of all itemsets in $C_k$. Another Apriori algorithm based algorithm, Signature-Apriori is proposed in [19], which analyzes the previously known signatures to find the signature of related attacks quickly. The only limitation of their system is that it is a misuse detection system and is unable to detect totally new types of attacks.

Since level-wise candidate generation as well as numerous scans of the database had been seen as a limitation of this approach, many optimization techniques of this approach had appeared in the literature and alternative tree-based solution proposal with Frequent pattern tree growth FP-growth [6], [7] had also been used. The FP-growth approach scans the database once to build the frequent header list, then, represents the database transaction records in descending order of support of the $F_1$ list so that these frequent transactions are used to construct the FP-tree. The FP-tree are now mined for frequent patterns recursively through conditional pattern base of the conditional FP-tree and suffix growing of the frequent patterns. Concepts of infrequent pattern computation and use of record anomaly scores computed from both frequent and infrequent patterns can also be applied with the efficient tree-based FP-tree algorithm for association pattern mining in application domains and could be explored in the future.

## 3. THE PROPOSED WIRELESS INTRUSION DETECTION SYSTEM

Section 3.1 presents definitions relevant to the proposed WiFi Miner IDS system, section 3.2 presents the overall WiFi Miner system architecture and algorithm, section 3.3 presents the Apriori-Infrequent algorithm used by the WiFi Miner system, while section 3.4 provides an example application of the Online Apriori-Infrequent algorithm.

### 3.1 Definitions and Properties

The following definitions and properties are used in the discussion of the proposed IDS system.

DEFINITION 3.1. A record has a maximal level of n*: if the record, $R_i$, has its largest frequent itemset being an n-itemset or containing* n *distinct items.* ■

DEFINITION 3.2. A maximal level n record has a set of frequent and infrequent itemsets*: consisting of all its 1-itemsets to n-itemsets that are frequent and infrequent respectively.* ■

DEFINITION 3.3. A Frequent k-itemset*: is a k-itemset which has support greater than or equal to the given minimum support with respect to the entire database stream of records.* ■

DEFINITION 3.4. An Infrequent k-itemset*: is a k-itemset which has support less than the given minimum support with respect to the entire database stream of records and has all its subsets frequent in levels* k-1 *and lower. This type of itemset is also called negative border in some work.* ■

DEFINITION 3.5. A maximal level n Record's Frequent Itemsets, $F_R$*: consists of the set of all its 1-itemsets to n-itemsets, which have supports greater than or equal to the*

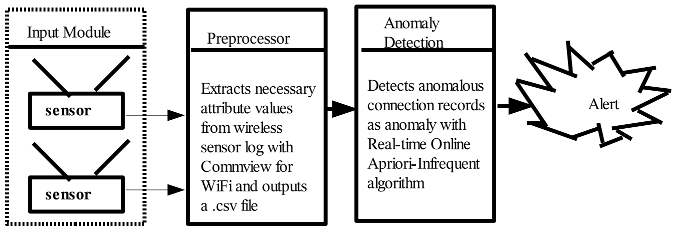
Input Module
sensor
sensor
Preprocessor
Extracts necessary attribute values from wireless sensor log with Commview for WiFi and outputs a .csv file
Anomaly Detection
Detects anomalous connection records as anomaly with Real-time Online Apriori-Infrequent algorithm
Alert

**Figure 1: WiFi Miner Wireless IDS**

*given minimum support with respect to the entire database stream of records.* ■

Definition 3.6. A maximal level n Record's Infrequent Itemsets, $IF_R$*: consists of the set of all its 1-itemsets to n-itemsets, which have supports less than the given minimum support with respect to the entire database stream of records. All subsets of each level* k *Infrequent set are frequent in the levels* k - 1 *and lower.* ■

Definition 3.7. A k-itemset Anomaly Score*: The anomaly score of a level* k *itemset is -k if the itemset is frequent but +k if the itemset is infrequent.* ■

Definition 3.8. A Record's Anomaly Score*: The anomaly score of a maximal level* n *record is the sum of all its levels 1 to* n *frequent and infrequent itemsets' anomaly scores.* ■

Proposition 3.1. A Normal/Anomalous Record Property*: A normal record has more frequent than infrequent itemsets and has a negative total record anomaly score, while an anomalous record has more infrequent than frequent itemsets and has a positive or zero total record anomaly score.* ■

## 3.2 The WiFi Miner Framework

The proposed WiFi Miner system framework comprises of three main modules. They are: Input Module, Preprocessor Module, and Anomaly Detection Module as shown in Figure 1. The proprietary Network Chemistry wireless hardware sensors [15] first need to be properly installed and configured before they can be used to capture wireless network packets. Installing the sensors entails installing both a sensor server and sensor client software systems and logging on to the sensor client software console system to initialize and configure the sensors. Input Module consisting of properly configured hardware sensors, collects network traffic data from hardware wireless sensors attached to the system, which capture data from airwaves as most of the wireless attacks may occur before data are in wired network and Access Points. The Preprocessor Module converts the raw data to readable format with the help of CommView for WiFi software, which is used to extract sensed data from the hardware sensor's firebird database and saved in a .csv file (csv stands for Comma Separated Values where attributes values are simple text separated by commas). With CommView, necessary features can be extracted for analyses to detect anomalies and extracted records stored as text file are processed directly by our WiFi Miner system. These records may also be logged into database tables for more offline processing and possible tracking of anomalous records. The focus of our approach is online processing, that is independent of training data. After the data are preprocessed, they are sent to the Anomaly Detection Module, which includes the core algorithm (Online Apriori-Infrequent) for finding infrequent patterns or anomalies.

The proposed Online Apriori-Infrequent algorithm contributes by

1. Providing a mechanism for computing the anomaly scores of a record, that is based on the relative sizes and numbers of infrequent and frequent itemsets contained in just this record without the need for hard-to-get training data. This is based on the premise that infrequent itemsets are likely anomalous as is the case with many wireless attacks.

2. Providing a smart-join mechanism that improves the Aprior-gen join step and prune steps when computing candidate itemsets, which speeds up infrequent and frequent pattern generations.

3. Providing a mechanism that eliminates the need to generate association rules from frequent patterns in order to detect anomalies.

Given a record, an anomaly score is computed from all its level *1* to level *n* patterns (both frequent and non-frequent patterns), where *n* is the largest number of items in the maximal frequent pattern as presented in the definitions. To compute the anomaly score of a record, each level *k* frequent pattern in the record is assigned an anomaly score of *-k*, while each level *k* infrequent pattern is assigned an anomaly score of *+k*, and the anomaly score of a record is the sum of the anomaly scores of all its frequent and infrequent patterns. If a record's total anomaly score becomes positive, then, this record has more infrequent than frequent patterns and is considered anomalous. On the other hand, if a record's anomaly score is negative, then, the record has more frequent than non-frequent patterns and is considered normal. If a record has zero anomaly score, it means it has the same number of frequent and infrequent patterns, and for increased security, the proposed system treats such a record as anomalous since it is safer to have a false alarm than harmful undetected intrusion. This anomaly detection module generates anomaly alerts for records with positive anomaly scores. The simple logic behind anomaly score weight assignment to frequent and infrequent itemsets is that the more the number of items in an infrequent itemset, the lower the chances of this itemset being in an arbitrary record. Thus, the presence of an infrequent 3-itemset is more rare than the presence of an infrequent 2-itemset in a record. Therefore, the anomaly weights of infrequent itemsets are proportionately increased with their size levels, while those of frequent itemsets are decreased with increasing number of items in the itemset. For example, while an infrequent 2-itemset like AC would have an anomaly score of +2, a frequent 2-itemset like AF would have anomaly score of -2. However, an infrequent 3-itemset would have an anomaly score of +3, while a frequent 3-itemset would have an anomaly score of -3.

The WiFi Miner algorithm is presented as Algorithm 1. The proposed scheme finds anomaly/infrequent patterns without training classifiers offline with safe data. Instead of finding frequent patterns at first and then comparing these patterns with incoming data to detect the anomalies during third step, our method finds the infrequent data/anomalies during the first step with an online Apriori-Infrequent algorithm, which tries to find both infrequent patterns and frequent patterns, improves candidate set generation scheme in one step by improving the runtime complexity of Joining and Pruning. The rest of the section describes both the Online Apriori-Infrequent algorithm and the Anomaly scoring scheme adopted by the proposed WiFi Miner system.

ALGORITHM 1. *(WiFi Miner: Wireless IDS)*

```
Algorithm WiFi Miner()
Input:   Network connection packets (P), sensors (S),
         access points (AP)
Output:  Anomalous connections (A)
begin
    While (true)
    (1) Capture wireless packets from AP using sensors (S)
    (2) Extract connection packets (P) from sensors S with
    Commview for WiFi software and save as .csv file
    (3) Call Apriori-Infrequent Algorithm with
    "Incoming-connection" .csv file records as input
    and output anomalous records as alerts.
end
```

## 3.3 The Proposed Apriori-Infrequent Algorithm

The goal of the Apriori-Infrequent Algorithm is to generate all frequent patterns as well as all infrequent patterns at every level, and be able to use this knowledge to compute anomaly scores for records. In order to compute frequent and non-frequent itemsets efficiently, the proposed algorithm argues that the Apriori's method for computing candidate (i+1)-itemsets by joining all frequent i-itemsets ($L_i$) with themselves, if their first (i - 1) items are the same and the first itemset comes before the second itemset in the $L_i$ list, can be improved on, with a third condition. The third join condition introduced by the Apriori-Infrequent algorithm states that an itemset in the $L_i$ list will only be used to join other items in the $L_i$ list that meet the first two conditions if this itemset's last item (or ith item) appears in a joinable item list called Z list, consisting of all (i-1)th item of $L_i$. The purpose of the Z list is to prevent ahead of time, the need to join itemsets which produce itemset results that have no chance of being frequent because their subsets are not frequent. Such itemsets in the Apriori algorithm are pruned during this step but we avoid both creating them in the first place, computing their subsets and pruning them. Our algorithm looks for infrequent patterns (which were frequent in the previous level but when they are combined with some other attributes, they become infrequent). These infrequent itemsets are similar to negative borders [13], but is computed in a more efficient fashion in our online Apriori algorithm. This concept of fast detection of infrequent pattern is useful for intrusion detection domain because suppose for example, in connection record, Flag ACK (ACKnowledgment) is frequent but when ACK is combined with Flag SYN (SYNchronized), it may be an attack. The formal Apriori-Infrequent algorithm is given as Algorithm 3 and the Smart-Join technique it uses is also given as Algorithm 2.

ALGORITHM 2. *(Apriori-SmartJoin:Computing Candidate $C_k$ from $L_{k-1}$)*

```
Algorithm Apriori-SmartJoin()
Input:   A list of large (k-1)-itemsets: $L_{k-1}$,
Output:  A list of candidate k-itemsets: $C_k$,
Other variables: Z-list for smart join
begin
    $C_k = \emptyset$
    Z = the set of all (k-2)th item in $L_{k-1}$.
    For each pair of itemsets M and P $\in L_{k-1}$ do
        begin
        M joins with P to get itemset $M \cup P$
        if the following conditions are satisfied.
        (a) itemset M comes before itemset P in $L_{k-1}$
        (b) the first k-2 items in M and P (excluding just
        the last item) are the same.
        (c) the last item (or (k-1)th item) of each itemset
        in $L_{k-1}$ is joinable only if this item is in the Z list
        If M and P are joinable then
        $C_k = C_k \cup M \cup P$
        end
end
```

ALGORITHM 3. *(Apriori-Infrequent:Computing Infrequent Patterns)*

```
Algorithm Apriori-Infrequent()
Input:   A list of candidate itemsets: $C_1$,
         minimum support count $\lambda$
Output:  A list of frequent itemsets: L,
         Anomaly score for each record.
Other Variables: A list of Infrequent itemsets: S,
begin
    k = 1
    1.Compute frequent $L_k$ and infrequent $S_k$ with
    minimum support $\lambda$ from $C_k$.
    2. While ($L_k \neq \emptyset$) do
    begin
    2.1. k = k+1
    2.2. Compute the next candidate set $C_k$ from $L_{k-1}$
         as $L_{k-1}$ Apriori-smart join $L_{k-1}$.
    2.3. For each itemset in $C_k$ do
         2.3.1. Calculate all possible subsets and prune if
         not previously large.
    2.4.If $C_k = \emptyset$ then break and go to step 3
    2.5.Compute frequent $L_k$ and infrequent $S_k$ with
         minimum support $\lambda$ from $C_k$.
    2.6.Update Anomaly Score for Connection Record by calling
    Anomaly Score function with $L_k$ and infrequent $S_k$
    end
    3.Compute all Frequent patterns as $L = L_1 \cup \ldots L_k$
    end
```

**Anomaly Score Calculation**

The proposed WiFi Miner system is able to calculate or give each connection packet an anomaly score on the fly. This is an important step as it eliminates the need to generate association rules from frequent patterns as done by many existing approaches in order to identify intrusions. The simple anomaly score rule assigns a positive anomaly score of *+n* to every n-itemset infrequent pattern in a record that is equal to the number of items in the infrequent pattern but assigns a negative anomaly score of *-n* to a frequent pattern with $n$ items. This rule is based on the premise that certain anomalies are infrequent events that embed themselves in frequent or normal packets. The anomaly score of each database transaction is computed in parallel with support counting of each level candidate set of the Apriori-Infrequent algorithm and this utilizes the records while they are still in memory without incurring additional I/O costs. Thus, the total anomaly score of a record is computed as the sum of all the anomaly scores of this record's itemset level *1* to level

**Table 2: Database Records Anomaly Scores**

| TID | Items | Anomaly Score | | |
|---|---|---|---|---|
| | | Pass 1 | Pass 2 | Final Score |
| 1 | A B D | -3+0 = -3 | -4+2 = -2 | -5 |
| 2 | A C E F | -2+2 = 0 | -2+10 = 8 | +8 |
| 3 | B C D F | -3+1 = -2 | -4+8 = 4 | +2 |
| 4 | A B C D | -4+0 = -4 | -8+4 = -4 | -8 |
| 5 | A B C E | -3+1 = -2 | -6+6 = 0 | -2 |

$n$ frequent and infrequent patterns, where $n$ is the last non-empty level of frequent patterns for the record. A record is declared anomalous if its total anomaly score is zero or positive but normal if its total anomaly score is negative.

## 3.4 An Application of the Apriori-Infrequent and Anomaly Score

Assume that wireless network connection records were captured and preprocessed to produce a database transaction table similar to columns one and two of Table 2, with candidate 1-items as {A, B, C, D, E, F}. In pre-processed wireless packets or records, the attributes depicted as A to F above would represent connection features like: connection date and time, source and Destination MAC address, packet size in bytes, access point MAC address(BSSID), Frame Type/Subtype, transmission rate, Client/AP sequence number, signal power, access point name, source type (station or access point), channel, etc.

**Example 1:** Using the WiFi Miner Apriori-Infrequent and Anomaly score counting technique, identify the anomaly or alert records from Table 2 (first two columns) if the minimum support threshold is 60% or 3 out of 5 transactions.

**Solution 1:** Applying Algorithm 3, $C_1$ = {A:4, B:4, C:4, D:3, E:2, F:2}, and $L_1$ = {A, B, C, D} with anomaly score each of -1 and $S_1$ = {E, F} with anomaly score each of +1. The anomaly scores of the transactions in the database table are computed at this level as: TID 1, ABD has an anomaly score of -1(A) -1(B) -1(D) = -3. TID 2, ACEF has an anomaly score of -1(A) -1(C) +1(E) +1(F) = 0. The anomaly scores of transactions 3, 4 and 5 are respectively: -2, -4, and -2. Next, we compute $C_2$ as $L_1$ Apriori-gen join $L_1$ since the Z list at this level is still empty set. Thus, $C_2$ = {AB:3, AC:3, AD:2, BC:3, BD:3, CD:2}. $L_2$ is computed as {AB, AC, BC, BD} with anomaly score of -2 each, while $S_2$ is computed as {AD,CD} with anomaly score of +2 each. The anomaly scores of the database transactions are updated as: Tid 1 (ABD) = -3(score from previous step) - 2(AB) +2(AD) -2(BD) = -5. Tid 2 (ACEF) = 0(score from previous step) - 2(AC) +2(AE) +2(AF) +2(CE) +2(CF) +2(EF) = +8. The rest of the anomaly scores are updated as shown in column 3 of Table 2. During iteration 3, to create $C_3$ list, the Z list is first created from $L_2$ as item (2 -1) or the first item in each $L_2$ itemset. Thus, Z = {A, B}. To join an $L_2$ itemset, if the last element of the itemset is not in the Z list, then, we should not perform the join. This means that we first reduce our $L_2$={AB, AC, AD, BC, BD, CD} to {AB} since AC, AD, BC, BD and CD do not have their last elements in the Z list. Thus, our $C_3$ = {AB} Apriori-gen join {AB} = $\emptyset$. Since $C_3 = \emptyset$ as well as $L_3 = \emptyset$, the algorithm ends without computing the anomaly score for this iteration. All records with negative anomaly scores are normal while those with positive or zero anomaly scores are alerts. The final anomaly scores of the example connection records are as given in column 3 of Table 2.

**Table 3: Attack Signatures Used**

| Attack Name | Attack Signature Used |
|---|---|
| SYN Flood | flag = S, dest-host = victim (same), dest-service = vulnerable port (same) |
| UDP Flood | dst-host = victim (same), dst-service = vulnerable port/random port |
| Port Scanning | (flag = S, src-host = attacking machine, dst-service = vulnerable port) (flag = R, dest-host = attacking machine, src-service = dest-vulnerable port) |

## 4. EXPERIMENTS AND PERFORMANCE ANALYSIS

To test the proposed system prototype, we installed Network Chemistry sensors in one PC from where we scanned all APs in ranges and selected the AP for our wireless network and started capturing packets from our Access Point. We created a wireless network with two other PCs where one was the victim and another one was the attacker PC. Within 5 minutes time window we have captured around 19,500 wireless packets, which were generated as a result of some innocent activities. Then we gathered around 500 anomalous packets which contained different kinds of crafted attacks like passive attacks (packets used for WEP cracking and packets used in Port scanning attack), active attacks (packets used for SYN Flood/ UDP Flood Attack), Man-In-the-Middle attack (packets used for establishing a rogue AP). Then, we launched these attacks from the attacker PC to the victim PC. Description of how we gathered these anomalous packets is provided next. Attack signatures used for these attacks are summarized in Table 3.

To crack a WEP (Wired Equivalent Privacy) key, at first, we spoofed a client's MAC address, which was known from previously generated innocent packets. Using this spoofed MAC address, we generated fake ARP packets and sent these packets to the AP (Access Point) using Aireplay [3]. In response to these fake ARP packets, the AP sent back reply packets, which were captured and used by Aircrack [2] to decrypt the WEP key. Our sensors captured all these fake ARP packets and these packets were considered and gathered as anomalous packets. Attack packets for SYN Flood attack, UDP Flood attack and Port Scanning attack can be created with tools like Engage Packet Builder [16]. To gather attack packets for Man-In-the-Middle type of attack, we set up a rogue AP with the same SSID (Service Set Identifier) as the legitimate one in a place nearer than the legitimate AP. Then, using the spoofed client's MAC address we sent de-authentication packets using Aireplay. As a result, the targeted client is disconnected from the legitimate AP and is connected to the rogue AP because of the stronger signal. These de-authentication packets were captured and gathered as anomalous packets. Then, we tested these combined dataset with our system WiFi Miner to verify the detection rate and total runtime. We also tested this input dataset with traditional Apriori based systems like ADAM and Snort Wireless to get a comparative performance analysis of our WiFi Miner system with existing mostly wired IDSs.

At first, we have compared the runtime of our algorithm: Real-time Online Apriori-Infrequent Algorithm with traditional Apriori algorithm concept used in ADAM and noticed an around 35% increase in execution time efficiency in our algorithm as shown in Figure 2. This is because we are not generating association rules with confidence value and also we have improved the join and prune sections of the algorithm with our Smart-Join approach. It should be stated here that from analysis and experiments, the proposed Online Apriori with smart join produces complete and correct frequent and infrequent patterns as the regular Apriori algorithm given the same datasets.

After we collected these 500 anomalous packets, then, we tested the combined (anomalous + innocent) dataset with our system WiFi Miner to verify the detection rate and total runtime. We also tested this input dataset with traditional Apriori based system like ADAM and Snort Wireless to get a comparative view of our system with existing ones. At first, we have compared the runtime of our algorithm: Real-time Online Apriori-Infrequent Algorithm with traditional Apriori algorithm concept used in ADAM and noticed an around 35% increase in execution time efficiency in our algorithm as shown in Figure 2. This is because our WiFi Miner system is not generating association rules with confidence value and also has improved the join and prune sections of the Apriori technique with a more efficient Smart-Join approach while still keeping the algorithm simple. We used around 19,500 innocent wireless packets along with 500 anomalous attack packets and tested them in WiFi Miner, Snort Wireless and Apriori based system in ADAM to see how many anomalous packets get detected in all three systems. We also calculated the false alarms produced by each system. In our testing model we had 500 anomalous packets. So, after each system flags connection packets as anomalous, we verify if that packet belongs to the class of our 500 anomalous packets. If the packet is not an anomalous packet, then it is counted as a false alarm. The total attack detection rate and false alarm comparative analysis of all systems is given in Table 4 and Figure 3 while a more detailed analysis of their detection of specific classes of attacks is provided in Table 5 and Figure 4. It can be seen from the tables that the proposed WiFi Miner system consistently detects more attacks than both Snort Wireless and ADAM in all categories of attacks. The proposed WiFi Miner system also records the lowest amount of false alarms.

Currently, the proposed WiFi Miner system has no mechanism for detecting Jamming Wireless attacks. Also, if the minimum support is set too low, there may be large number of frequent itemsets and fewer infrequent itemsets. As a result, attacks may go undetected. Experiments show that for this wireless intrusion detection domain, a good choice of minimum support is 60% or more. Future work should explore improving efficiency of the system, handling more types of attacks and further reduction of false alarms.

## 5. CONCLUSIONS AND FUTURE WORK

This paper proposes a wireless intrusion detection system: WiFi Miner, which uses Apriori-Infrequent based algorithm

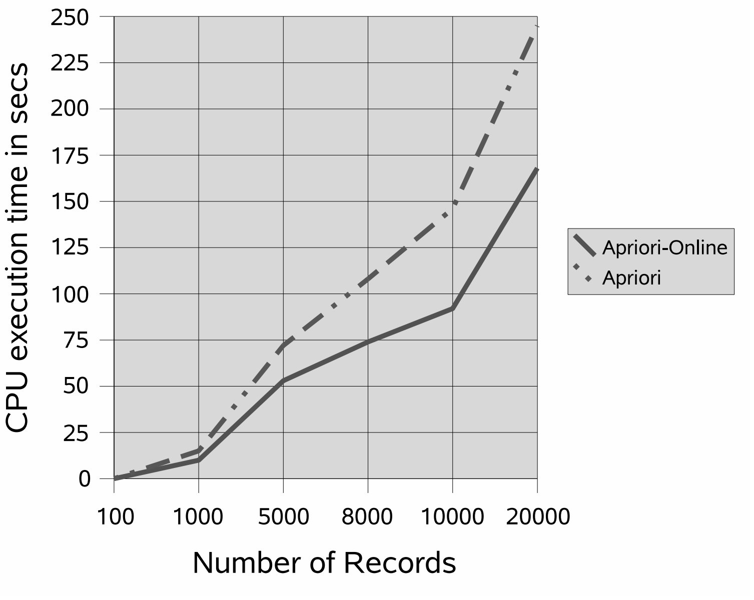

**Figure 2: Comparison of Apriori-Infrequent with Apriori Algorithm**

**Table 4: Attacks Detected and False Alarm Comparison**

| Detected | (Out of 500 attacks) | | |
|---|---|---|---|
| | WiFi Miner | Snort Wireless | ADAM |
| Attacks Detected | 433 | 335 | 377 |
| False Alarm | 180 | 292 | 248 |

**Table 5: Specific Attacks Detected Comparisons**

| Detected | 3 Algorithms | | |
|---|---|---|---|
| | WiFi Miner | Snort Wireless | ADAM |
| Passive Attacks (200 attacks) | 179 (89.5%) | 138 (69%) | 161 (80.5%) |
| Active Attacks (200 attacks) | 171 (85.5%) | 145 (72.5%) | 151 (75.5%) |
| Man-In-Middle (100 attacks) | 83 (83%) | 52 (52%) | 65 (65%) |

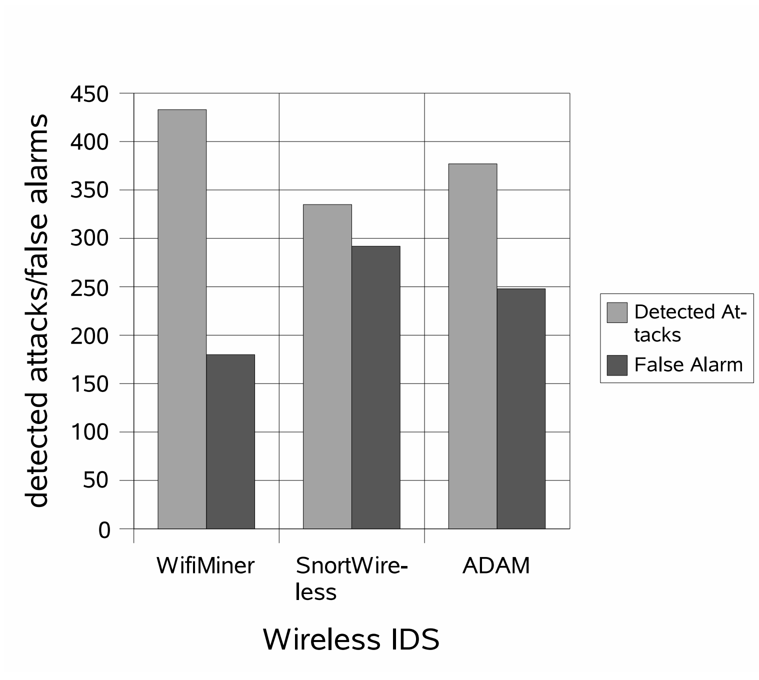

**Figure 3: Comparisons of WiFiMiner with SnortWireless and ADAM**

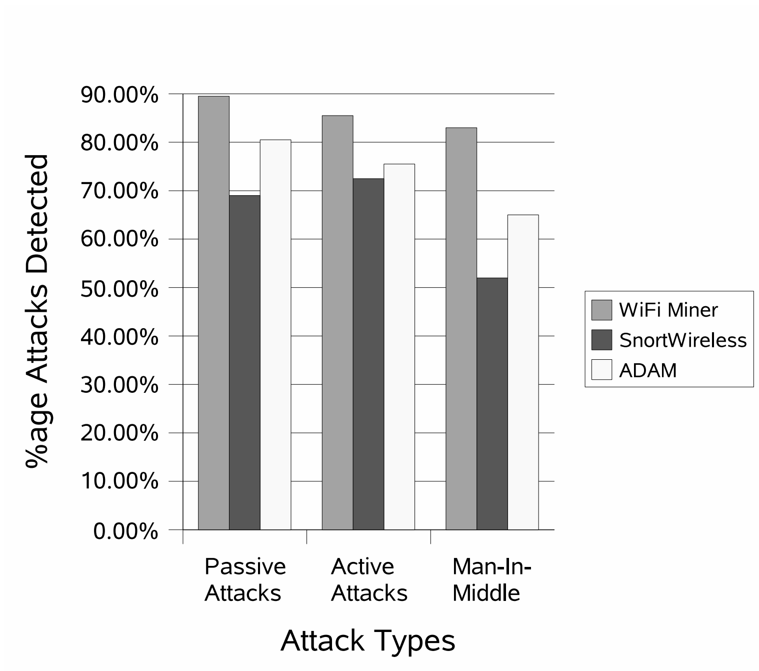

**Figure 4: Specific Attacks Comparisons of WiFiMiner with SnortWireless and ADAM**

to detect infrequent patterns, then our algorithm designed for Anomaly Score Calculation, assigns a score to each wireless packet. Positive or zero anomaly score in a specific connection record means that more infrequent/anomalous patterns are found in that record than frequent patterns while a negative anomaly score indicates a normal packet. We have also used proprietary Network Chemistry hardware sensors to capture real-time traffic in order to improve intrusion response time. Our system is different from existing wireless intrusion systems, since it eliminates the need for hard-to-get training data and detects intrusions in real time. Also, like other existing wireless intrusion systems, it captures the packets from airwaves while wired IDSs use net-flow data from routers. Thus, the major contribution of our system is that it can detect anomalous packets in real time without any training phase. We have tested our system with crafted intrusions and compared it with other two systems and found our system to be more efficient. Another major contribution is that we have introduced Smart-Join, which is an improved version of Join and Pruning steps in original Apriori algorithm.

In the future, we plan to enhance our system to work with many access points, currently it is capable of handling wireless connection records from one access point although our sensors are capable of finding all APs in their ranges. We are also working towards making our system generalized so that it can be used for both wired and wireless intrusion detection. Other future work include applying this online intrusion detection system approach to other domains like environment pollution monitoring systems where excessive levels of pollution can quickly raise alerts as anomalies from sensor captured data.

## 6. ACKNOWLEDGMENTS

This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an operating grant (OGP-0194134) and a University of Windsor grant.

## 7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on very Large Databases Santiago, Chile*, pages 487–499, 1994.

[2] Aircrack. Airdump web page. http://airdump.net/papers/packet-injection-windows, 2007.

[3] Aireplay. Airdump web page. http://airdump.net/papers/packet-injection-windows, 2007.

[4] D. Barbara, J. Couto, S. Jadodia, and N. Wu. Adam: A testbed for exploring the use of data mining in intrusion detection. *ACM SIGMOD RECORD: Special Selection on Data Mining for Intrusion Detection and Threat Analysis*, 30(4), 2001.

[5] L. Ertoz, E. Eilertson, A. Lazarevic, P. Tan, J. Srivastava, V. Kumar, and P. Dokas. *The MINDS - Minnesota Intrusion Detection System in Next Generation Data Mining*, chapter 3. MINDs, 2004.

[6] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, New York, 2000.

[7] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *International Journal of Data Mining and Knowledge Discovery*, 8(1):53–87, Jan 2004.

[8] T. Imielinski, A. Swami, and R. Agarwal. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD conference on management of data*, pages 207 – 216. ACM, 1993.

[9] W. Lee and S. J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM Transaction on Information and System Security*, 3(4):227–261, Nov. 2000.

[10] Q.-H. Li, J.-J. Xiong, and H.-B. Yang. An efficient algorithm for frequent pattern in intrusion detection. In *Proceedings of the International Conference on Machine learning and cybernatics*, pages 138–142, Nov. 2003.

[11] Y. Liu, Y. Li, H. Man, and W. Jiang. A hybrid data mining anomaly detection technique in ad hoc networks. *International Journal of Wireless and Mobile Computing*, 2(1):37–46, 2007.

[12] V. Mahoney and P. K. Chan. Learning rules for anomaly detection of hostile network traffic. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM)*, pages 601 – 604. IEEE, 2003.

[13] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *International Journal of Data Mining and Knowledge Discovery*, 1(3):241–258, Jan 2004.

[14] V. Marinova-Boncheva. Applying a data mining method for intrusion detection. In *ACM International Conference Proceeding Series*. ACM, June 2007.

[15] NetworkChemistry. Network chemistry wireless security business. http://www.networkchemistry.com, 2007.

[16] E. Security. Engage security web page. http://www.engagesecurity.com, 2007.

[17] R. J. Shimonski. Wireless attacks primer. A whitepaper published on windowssecurity.com section: Articles: Wireless security, July 2004.

[18] K. Yoshida. Entropy based intrusion detection. In *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, PACRIM*, pages 28–30. IEEE, August 2003.

[19] H. Zhengbing, L. Zhitang, and W. Junqi. A novel intrusion detection system (nids) based on signature search of data mining. In *1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop*, January 2008.

[20] S. Zhong, T. Khoshgoftaar, and S. Nath. A clustering approach to wireless network intrusion detection. In *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 190–196. IEEE, 2005.