IRS: An Issue Resolution System for Cyber Attack Classification and Management

Chris B. Simmons, Sajjan Shiva, Vinhthuy Phan, Vivek Shandilya
Department of Computer Science
University of Memphis
Memphis, TN, USA
{cbsmmons, sshiva, vphan, vmshndly}@memphis.edu

Lakisha Simmons
Department of Management of Information Systems
Indiana State University
Terre Haute, IN, USA
lakisha.simmons@indstate.edu

***Abstract*—Cyber-attacks have greatly increased over the years, where the attackers have strategically improved in devising attacks toward a specific target. In order to correctly classify cyber-attacks there is a considerable need to neatly organize a representation scheme that is useful in an application setting. The classification of cyber-attacks within knowledge bodies, such as Computer Emergency Readiness Team (CERT) and Common Vulnerabilities and Exposures (CVE), are very useful for organizations gathering data as information is made available. However, there is substantial information to decipher when locating relevant details that are prevalent in local networks. We propose an issue resolution system (IRS) to detect and extract information from external vulnerability repositories and internal log files to assist with classifying and disseminating defenses. In this work we provide a frequent pattern classification algorithm that performs data mining techniques to classify attack vector information from the national vulnerability database (NVD). The results suggest the IRS presents a viable solution to correctly extract vulnerability information within a local knowledge base.**

Keywords-Security; Security Management, Information Extraction; Algorithm; Taxonomy

I. INTRODUCTION

Cyber-attacks have received increased attention over the last decade, where researchers are investigating the relationships between attacks and the associated defenses. Organizations, particularly small-to-medium sized, lack the capacity to effectively capture cyber-attack related information and disseminate appropriate defenses. These particular organizations rely on a select set of knowledgeable security personnel to resolve cyber-attack related issues. With cyber threats on the rise, it is necessary to correctly identify the suspected threat in a timely manner. Frequent pattern analysis has been used consistently within data mining through the ability to relate patterns.

Frequent patterns are defined as itemsets, subsequences, or subsets that appear in a data set with a certain level of frequency [1]. Sequential pattern mining is the discovery of frequent subsequences [2]. Both frequent and sequential analysis types are beneficial in correlating attack vector information using frequencies and sequences. Correlation is typically used with machine learning approaches or pattern detection algorithms. Lee et al. [3] used timed signatures to tag signatures in discovering database intrusions. We extend this approach to a repository of common vulnerabilities and exposures (CVE), where the vulnerabilities are associated to a specific application setting.

Han et al. [1] highlighted the importance of frequent pattern analysis for data indexing, classification, clustering, etc. Accurate cyber-attack classification is pertinent for suitable damage assessment and recovery. Web logs are used locate potential attack vectors within a particular application. Parsing web log files and database insertions enable an analysis of the current state and transitions of the application. Information stored within the repository can be easily queried to retrieve frequent and/or sequential data items. Although beneficial, using this method can become expensive when querying a large dataset. Pei et al. [2] highlighted the need for extending sequential patterns towards considering time constraints, time windows, and/or a taxonomy. We extend this concept by utilizing a cyber-attack taxonomy, consisting of Attack Vector, Operational Impact, Defense, Informational Impact, and Target (AVOIDIT). AVOIDIT facilitates a classification mechanism that increases the efficiency of correlating attack vector information. From a cursory scan of literature, there is a lack of research focused on correlating external knowledge bodies, such as Computer Emergency Readiness Team (CERT), with internal extracted information, such as web log files, to produce a knowledge base containing a sequential order of attack steps. One of the problems faced by research pertaining to attack classification is how to classify the vast nature of attacks and their potential to polymorph. Understanding cyber-attack defense is chess and not checkers, we can provide a way to capture the wide array of moves an attacker may take through appropriate classification and response.

In this paper, we propose initial work of an Issue Resolution System (IRS) for extracting and dessiminating attack vector information in a local application setting. The IRS uses a classification algorithm which consists of two major methods, a classification method and a decision tree method. The classification algorithm uses AVOIDIT to identify the related attack vector information for classification. Once classified, IRS presents the information via a SilverStripe knowledge base for analysis. We

demonstrate the IRS applicability through mining and extracting 160 CVE descriptions from the National Vulnerability Database.

This paper is organized as follows. In section II we highlight the literature involving correlation and frequent pattern algorithms. In section III we propose the issue resolution system's architecture and describe its components in detail. In section IV we provide a preliminary experiment and results of our issue resolution system and section V we conclude our work with insight into future work.

II. Related Work

There are several recent efforts regarding techniques to automate correlating attacks, where defenders can view a collection of data seamlessly. However, this task creates a massive amount of data that defenders are unable to decipher in a reasonable amount of time. There is a wide array of pattern analysis techniques. In this section we provide a review of literature relating the frequent pattern analysis.

Hu and Panda [4] proposed a data mining approach for detection intrusion alerts targeted towards data corruption. Their approach concentrates classification rules to mine the database for dependencies between two or more data items. Hu and Panda [4] used the database logs to deduce data dependencies. Data dependencies that are not compliant are flagged as anomalous. The result performance increased where dependencies were stronger amongst data items.

Han et al. [1] proposed a frequent-pattern tree approach to mining large amounts of frequent patterns in a transactional database. Han et al. provides an extension to the Apriori algorithm through the use of partitions, divide-and-conquer growth patterns. This approach utilizes solutions to smaller tasks. The approach scans the database twice, one to get frequencies and another to construct the frequency tree. Efficiency is achieved using a three techniques, a large database is compressed into smaller data structure, a fp-tree-based mining using pattern-fragment growth to avoid costly generation, and partitions-based divide and conquer method.

Leung, et al. [5] proposed a canonical-order tree algorithm that captures the content of the transaction database and orders tree nodes, called CanTree. This work provided an extension to the FP-tree algorithm for incremental mining. Leung, et al. uses CanTree to efficiently arrange tree nodes according to canonical order, which are unaffected by frequency item changes. This provides easy maintenance when transactions are modified within a database.

Zaki [6] presented a TreeMiner algorithm which discovers all frequent subtrees in a forest. This novel algorithm performed a depth first search for frequent subtrees using a tree representation called scope list. The use of scope list improved the ability for fast support counting of candidate trees. Cheung and Zaiane [7] proposed a FELINE algorithm, which is tree based incremental mining algorithm, containing a CATS tree.

Pei et al. [8] proposed a prefix-projected sequential pattern mining (Prefix-Span) that explores prefix-projection in sequential patterns. Prefix-Span was developed to reduce the time of subsequence generation while mining the complete set of patterns. The goal of PrefixSpan is to examine the prefix subsequences which allow a representation of the postfix subsequences in the database. They presented valuable information needed to successfully gather the prefix attack information from various input sources in order to successfully disseminate the post attack information. This provides insight into the proposed IRS, as attack vector information becomes available it can be used to retrieve potential defenses regarding suspecting attacks.

Ning et al. [9] proposed three utilities to facilitate correlating a large dataset of IDS related alerts. These utilities are adjustable graph reduction, focused analysis, and graph decomposition. This resulted in the correlation of using consequences of earlier events with prerequisites later events. Ning et al. [9] presented an interesting approach to navigate through the enormous amounts of data captured from an IDS. Ning et al. [10] follow-up work is an extension which focused on correlation to construct attack scenarios using hyper-alert type representing prerequisite and consequences of each alert type of an attack.

We propose IRS, which suggests a new correlation algorithm that uses a cyber-attack taxonomy towards a classification of attack vector information. The correlation algorithm uses the discovery of new attack vectors in aspiration of establishing a relationship between attack vectors based on frequency of sequential events. We further propose a tree based algorithm to be used within a knowledge repository to use the classified attack vectors information with assisting defenders to view the complete path of an attack.

III. Issue Resolution System Architecture

The issue resolution system enables an organization to use a formalized apparatus to communicate seamlessly regarding the discovery of attacks and defenses. It supports security awareness within organizations by offering attack identification, attack classification, and assist with attack resolution to the identified attack. IRS provides seamless communication by using the following five components: (i) an Ontology, (ii) a Cyber-attack Taxonomy, (iii) a Classification algorithm, (iv) a Log Parser, (v) and a Knowledge Base. Figure 1 depicts each component in a diagram of the issue resolution system.

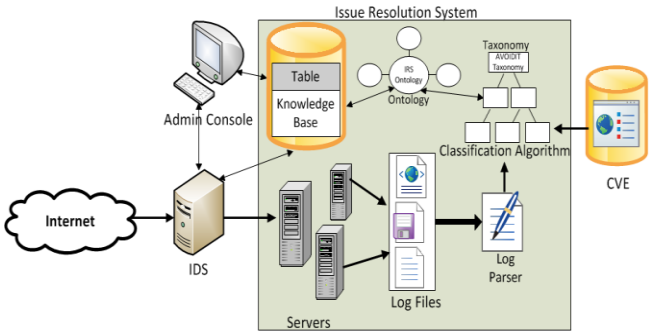

Figure 1. Issue Resolution System Architecture

The ontology is representative of a global communication scheme which consists of the cyber-attack taxonomy. The ontology uses input from the knowledge base and the classification algorithm to assist with facilitating identification, response, and resolution.

The taxonomy is based on a cyber-attack taxonomy called AVOIDIT [11]. Based on the parameters provided as input from external repositories and internal log files, the taxonomy uses the classification algorithm to identify the characteristics of an attack.

The classification algorithm uses frequent and sequential pattern analysis to classify attack vector information. This information is then stored in the repository to identify relevant attacks and potential defenses.

The Log Parsers are scripts written as sensors to identify potential intrusions. These scripts are written in Microsoft Log Parser and provide the ability to parse various log files for suspicious activity. In this paper we limit the discussion to web server log files.

Once an attack is identified, the knowledge base acts as the repository and searches for additional attack vector information. The knowledge base stores information related to the sequence of an attack and provides potential defenses to mitigate and/or remediate the damage to a system.

A. *IRS Ontology*

An ontology is an explicit specification of concepts related to a specific domain and the relationships amongst those concepts to create an organized knowledge base. Ontologies are a common way to organize knowledge and involves the description of objects and relationships [14]. Cyber-attack management is critical in the application of the IRS. The ontology processing will capture attack details from the knowledge base to begin attack analysis.

Figure 2 highlights the IRS ontology to support the communication flow within an organization upon attack discovery. The objective of the security ontology is to provide knowledge representation of the most relevant security concepts within an organization. The **ovals** refer to major concepts that are needed to successfully communicate an incident within the IRS. The **boxes** refer to the terminal entities that provide specifics of the superclass concepts. The **arrows** refer to the relationships between concepts relevant to incident management.

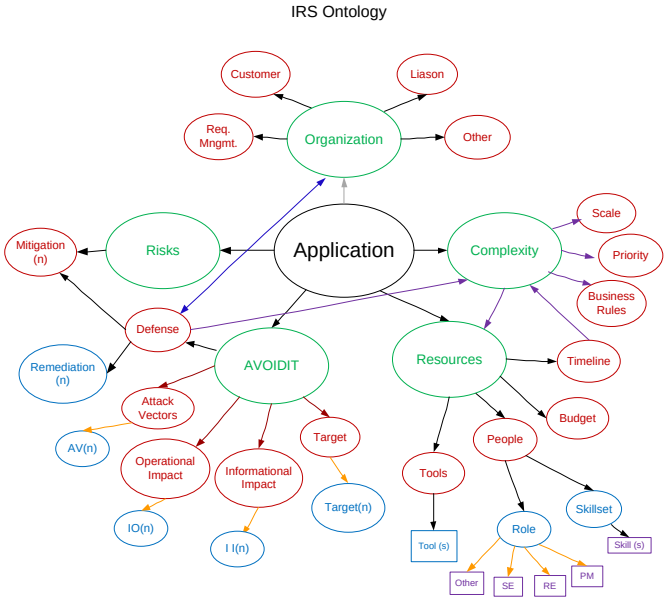

Figure 2. IRS Ontology

B. *Cyber Attack Taxonomy*

This section describes the cyber-attack taxonomy used in an application setting to classify attacks. Simmons, et al. [11] provided the AVOIDIT cyber-attack taxonomy to support comprehending each attack classification and how a variety of attacks are represented in each category. Using AVOIDIT we classify attacks via attack vectors where the repository disseminates the potential attack for an appropriate defense selection. Due to space constraints, Figure 3 highlights a representative subset of AVOIDIT, where each classification is defined in the following sections. Our approach follows this pattern:

- **Classification by Attack Vector**
  When an attack takes place, there is a possibility that it uses several vectors as a path to a complete cyber-attack. An attack vector is defined as a path by which an attacker can gain access to a host.

This definition includes vulnerabilities, as it may require several vulnerabilities to launch a successful attack.

- **Classification by Operational Impact**
  Classification by Operational Impact involves the ability for an attack to culminate and provide high level information known by security experts, as well those less familiar with cyber-attacks. We provide a mutually exclusive list of operational impacts that can be categorized and concisely presented to the public.

- **Classification by Defense**
  We extend previous attack taxonomy research to include a defense classification. We provide the possibility of using both mitigation and remediation when classifying attack defenses, as an attack could be first mitigated before a remediation can occur.

- **Classification by Informational Impact**
  An attack on a targeted system has potential to impact sensitive information in various ways. A committed resource must be able defend information warfare strategies in an effort to protect themselves against theft, disruption, distortion, denial of service, or destruction of sensitive information assets [15].

- **Classification by Target**
  Attacks target a variety of hosts, leaving the defender unknowingly susceptible to the next attack. This section is used to classify targets an attack uses to perform unauthorized privileges.

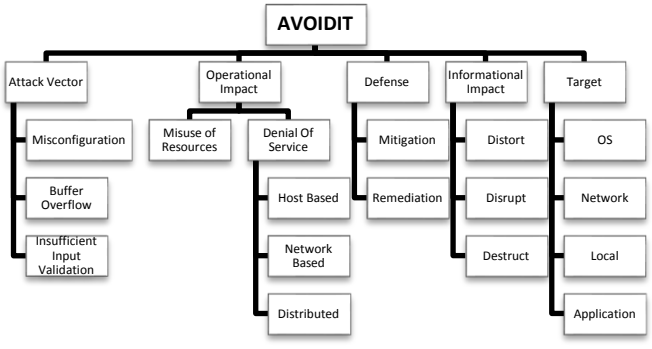

Figure 3. AVOIDIT: A Cyber Attack Taxonomy

C. Classification Algorithm

The classification algorithm is the functioning mechanism behind the issue resolution system, which takes the keywords as input, matches them with those that are accounted for in the AVOIDIT taxonomy to traverse the tree to reach a leaf node. This provides the ability of identifying the attack described by the keywords. The algorithm is implemented in a program written in PHP. The MySQL database is used for the representing and analysis of attack vectors.

Each set of input strings the application retrieves the associated attack vectors using the classification algorithm (Algorithm 1) and AVOIDIT to classify input description against keywords within the repository. If the input description has at least one match, then we select all the classifications of attack vectors associated to the input description. Using each classification correlated with the input description, the frequency is retrieved. The classification algorithm retrieves the attack vector that contains the highest probability of the attack vector being associated to the description received from the input. Once the attack vectors are returned, the attack vector is passed to the tree algorithm (Algorithm 2) to retrieve the associated attack tree. This provides discovery of the initial and later parts of the identified part of the attack to complete the attack. The algorithm is an intuitive depth search algorithm searching all attack trees to find a match. This information can be used by the defender to assist with devising defense strategies. The application continuously receives input description from the vulnerability repository to update information related to attack vectors.

Table 1. Notations used in AVOIDIT correlation algorithm

Notation	Definition
n	Extracted string from a log
cav_i	Classifications of previous attack vectors
T	Attack tree
v	Set of Vertices
x	Parent node

Algorithm 1. Classification Algorithm (A data retrieval algorithm, which is retrieved directly from repository)

Input: a set of strings N.
Output: A set of attack vectors cav names associated to set N and $freq$

The *AVOIDIT Classification* Algorithm

1. **for** each $n \in N$
2. **if** n_i is at least one match
3. select all classifications cav including frequency
4. **else if**
5. **for** each variation of n_i //uses php pecl function
6. select all classifications cav including frequency
7. **end for**
8. **else**
9. set $n_i \rightarrow$ unclassified
10. **end if**
11. **end for**
12. **retrun classifications** cav and $freq$

The CVE descriptions or application log files are parsed for keywords, which are described in Algorithm 1. Algorithm 1 accepts an input file containing a set of strings. The algorithm parses the data and uses the keyword matches within the repository to classify new attack vectors and their associated target. The algorithm retrieves the associated classifications for each keyword recognized within the repository.

The premise behind the attack tree algorithm is to provide a database scan of the classified attack vectors to retrieve the associated trees. If no tree exists, the attack vector becomes the root level for future related attack tree retrieval. If a tree exists for the classified attack vector, the child nodes are compared with other existing nodes. If a tree exists where two or more attack vectors contain the same parent the attack trees are merged into one tree. This assists an organization with capturing knowledge to itemized phases of a complete attack.

Algorithm 2. Attack Tree Algorithm (A depth first search to retrieve attack tree)

Input: a set of classified attack vectors *CAV*.
Output: A set of attack trees T_{av} that simulates the complete path of an attack.

The *AVOIDIT Tree* Algorithm

for each $cav_i \in CAV$
  find corresponding attack Trees T_n
  if cav_i does not have a corresponding T_n
    $cav_i \rightarrow T_{av}$
  else
    Given a set of tree $T_{1\ to\ n}$
    for each T_n
      for all $v \in T_n$
        visited $(v) = false$
      for all $v \in T_n$
        if not visited (v): $explore\ (T_n)$
      construct a tree T_{av} s.t. all *v* are covered for cav_i
      set parent in T_{av} of $cav_i \rightarrow x_i$
    end for
  end if
end for
for all $x_{1\ to\ i}$ are equal
  union $T_{av}\ (x_i, x_{i+1})$
return a set of attack trees T_{av}.

D. Log Parser

Log data is capable of recording important events, which should be analyzed on an ongoing basis, consistent with the monitoring of other key centralized security controls [12]. MS Log Parser [13] tool was developed to parse recorded events that have occurred in a system and/or application. It contains a core SQL engine facilitating the use of data repository for further analysis.

IRS uses MS Log Parser to turn large amounts unstructured text into a form (in a MySQL database) that can be manipulated to understand patterns, relationships, and meanings by using sensors. This enables IRS to retrieve various log data via a local network to correlate with pre-existing attack vectors. LogParser Query 1 highlights example queries used to retrieve pertinent events within a Windows registry or from an IIS web server log file.

LogParser Query 1. Windows Registry Events and Web Log File Events

Input: Windows registry location.
Output: A set of registry keys that have been modified within the past 24hrs.

Example *Log Parser* Query

```
logparser SELECT path, valuename from hklm\software where lastwritetime >= sub(system_timestamp(), timestamp('0000-01-02', 'yyyy-MM-dd')
```

Input: IIS Web log file.
Output: A set of status code changes for a selected files.

Example *Log Parser* Query

```
logparser SELECT DISTINCT date, time, c-ip, cs-uri-stem, sc-status from %web.log% WHERE c-ip IN (SELECT DISTINCT c-ip FROM %web.log% WHERE sc-status = 304) AND (sc-status=200 OR sc-status=304) ORDER BY date, c-ip
```

E. Knowledge Base

The KB is a component within the IRS that facilitates the storage of various attack related information. Once the IRS identifies an attack vector, this information is forwarded to the KB which then extracts further information related to the attack vector to ensure appropriate classification and retrieve potential solutions.

The attack related information consists of external vulnerability descriptions and internal log data. We utilize information from various sources depicting the complete path to an attack. These various sources highlight information which can be used to correlate disparate and unstructured data within the KB.

We envision the KB to work in either an offline or online mode. The KB can operate in online mode by as described above. The KB operates in offline mode by updating itself semi-autonomously by using its access to online vulnerability databases and security expert intervention.

IV. Experiment and Results

This work emphasizes an issue resolution system appropriate for auditing external repositories and internal web log files to correlate attack vector information. In this paper focus was placed on the classification algorithm for searching and classifying attack vectors information within a repository. Experiments show the classification algorithm is practical within the issue resolution system to proceed with further development.

A. *Methodology*

The use of the common vulnerabilities and exposures (CVE) database was used to classify pre-existing attack vectors. In conducting this preliminary experiment the CVEs from the National Vulnerability Database (NVD) were used for classification. Considering the massive number of CVEs, focus was placed on a real world scenario, where an organization uses Joomla! as a web based content management system.

The methodology for our experiment involved a training set of 60 positive CVEs associated with Joomla! for learning. Table 2 depicts a CVE description specific to Joomla!, which highlights the vulnerability information. The algorithm was trained using a standard unigram bag of words approach. In each CVE description, experiential knowledge was used to classify the concepts of interests from the text relative to the attack vector, operational impact, defense, informational impact, and target.

Table 2. CVE Description for Joomla!

CVE -2011-4808	
Summary	SQL injection vulnerability in the HM Community (com_hmcommunity) component before 1.01 for Joomla! allows remote attackers to execute arbitrary SQL commands via the id parameters in a fnd_home action to index.php.
Published	12/14/2011
CVSS Severity	7.5 (High)

The second step involved a test set of 100 unique CVEs associated with Joomla! and 50 random CVEs (for noise) associated with various open and closed source software. The goal is to ensure Joomla! related CVEs are correctly classified and distinguished from irrelevant CVEs. The irrelevant CVEs were discarded by the IRS. Performing this step simulates data being pushed to the user via the IRS providing needs specific attack vector information. Preliminary evaluation on this minimal dataset highlighted the algorithms ability to correctly classify application pertinent incidents specific to an organization.

B. *Results*

In this section we provide preliminary results of the IRS prototype giving insight to the potential success of our concept. Figure 4 highlights the Silverstripe knowledge base used for our experiment.

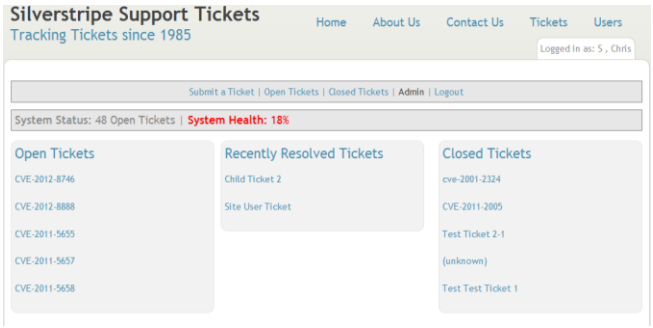

Figure 4. Silverstripe knowledge base interface

Precision was used to measure the accuracy in which the classification algorithm was able to correctly classify attack vector information. The use of precision was measured by dividing the total number of correctly classified items by the total number of extracted items, as provided in Equation 1. Recall was also used to measure the percentage of available correct information extracted, which highlights the algorithms ability to extract relevant information. Recall is the number of correctly classified by the total possible correct classifications, as provided in Equation 2. Table 3 displays the actual precision and recall computed for both the training set and test set using the classification algorithm.

$$Precision = \frac{Total\ Correctly\ Classified}{Total\ Extracted} \quad (1)$$

$$Recall = \frac{Total\ Correctly\ Classified}{Total\ Possible\ Correct\ Classifications} \quad (2)$$

Table 3. Algorithm Precision and Recall Computation

	CVEs Correctly Classified	Precision	Recall
Training Set	47/60	78.3%	78.3%
Test Set	92/100	92%	92%

Based on the results we can derive the following facts. The percentage of correctly classified CVEs are relatively high. This may be due to the use of only Joomla! as the test application for this experiment. It further seems a lot of CVEs mainly deal with input validation vulnerabilities, which allowed our classification algorithm to remain somewhat consistent.

Although, the classification algorithm is preliminary, it presents an approach of consideration using machine learning and good information extraction to increase efficiency. The time required for processing attack vector information was reasonably small satisfying any stringent timing requirement.

V. CONCLUSION

It is imperative the attack vector information is correctly classified and disseminated throughout an organization. In this paper we presented a new approach to attack vulnerability classification that locates keyword

matching from the NVD repository containing CVEs. Such keyword matching is correlated to the local network for issue resolution of the current system.

Our preliminary experiment indicated a promising solution so as to move forward with more detailed research. We assume the collected keywords were pertinent to the type of attack vector information within the repository and can be used generally. We believe providing a mechanism to distribute public information and correlate within a local network will assist organizations with identifying the underling details of an attack. Although our approach is preliminary, we believe the approach provides a direction for information extraction and machine learning in security management.

Future work will investigate an elaborate machine learning technique through the use of natural language processing that will allow the classification algorithm to learn various descriptions within the CVE for use within a local setting. We will use attack test beds such as metasploit [17] to explore the ability of the classification algorithm to identify the correct attack.

We further investigate a technique intended to use the AVOIDIT algorithms in a game inspired defense architecture with aim to extend the functionality of previously proposed game models addressing a broader array of cyber and data engineering problems [16]. Using the AVOIDIT algorithms we intend to build a Game Theoretic Defense System, which will investigate the applicability of AVOIDIT in determining the action space of the attacker and defender.

REFERENCES

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

[2] Pei, J., Han, J., Asi, B.M., Pino, H. PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth, in: Proc. The Seventeenth International Conference on Data Engineering, April 2001, pp. 215–224.

[3] Lee, V. C.S., Stankovic, J. A., Son, S. H. Intrusion Detection in Real-time Database Systems Via Time Signatures. In Proceedings of the Sixth IEEE Real Time Technology and Applications Symposium, 2000.

[4] Hu, Y., & Panda, B. A Data Mining Approach for Database Intrusion Detection. Proceedings of the 19th ACM Symposium on Applied Computing, Nicosia, Cyprus, 711-716, 2004.

[5] Leung, C.K., Khan, Q.I., Li, Z., Hoque, T. CanTree: a canonical-order tree for incremental frequent-pattern mining, Knowledge and Information Systems 11(3) (2007) 287–311.

[6] ZAKI, M. J. Efficiently mining frequent trees in a forest. Inf. Syst. 17, 8, 1021 – 1035, 2005.

[7] Cheung, W. and Zaiane, O.R. "Incremental mining of frequent patterns without candidate generation or support constraint," In Proc. IDEAS 3003, pp. 111–116.

[8] Pei, J., Han, J., Asi, B.M., Pino, H. PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth, in: Proc. The Seventeenth International Conference on Data Engineering, April 2001, pp. 215–224.

[9] Ning, P., Cui, Y., and Reeves, D. S. Analyzing intensive intrusion alerts via correlation. In Proc. of the 5th Int'l Symposium on Recent Advances in Intrusion Detection, October 2002.

[10] Ning, P., Cui, Y., Reeves, D.: Constructing attack scenarios trough correlation of intrusion alerts. In: CCS '02: Proc. 9th ACM Conference on Computer and Communication Security, ACM Press (2002) 245-254.

[11] Simmons, C., Shiva , S., Dasgupta , D., and Wu, Q., "AVOIDIT: A cyber attack taxonomy,", Technical Report: CS-09-003, University of Memphis, August 2009.

[12] Scarfone, K., Grance, T., Masone, K. "Computer Security Incident Handling Guide," NIST SP 800-61, *2008*.

[13] Giuseppini G, Burnett M, Faircloth J, Kleiman D. Microsoft Log Parser toolkit. Syngress; 2005.

[14] Tran, Quynh-Nhu Numi, and Graham Low. "MOBMAS: A methodology for ontology-based multi-agent systems development." *Information & Software Technology* 50, no. 7/8 (June 2008): 697-722.

[15] Cronin, B. and Crawford, H., "Information warfare: Its Application in military and civilian contexts", Information Society, volume 15, pp. 257-263, 1999.

[16] Shiva, S., Dasgupta, D., Wu, Q. "Game Theoretic Approaches to Protect Cyberspace," Office of Naval Research, Grant Number N00014-09-1-0752, 2009.

[17] http://www.metasploit.com/, retrieved on March 3, 2012.