# A User-centric Intrusion Detection System by Using Ontology Approach

**Shao-Shin Hung[1] and Damon Shing-Min Liu[2]**

Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan 621, Republic of China
{hss[1], damon[2]}@cs.ccu.edu.tw

## Abstract

In the security infrastructure, intrusion detection has become an indispensable defense line in face of increasing vulnerabilities exposed in today's computing systems and Internet. In this paper, our approach uses ontologies as a way of grasping the knowledge of a domain, expressing the intrusion detection system much more in terms of the end users domain, generating the intrusion detection more easily and performing intelligent reasoning. Experimental results show that our anomaly detection techniques are very promising and are successful in automatically detecting intrusions at very low false alarm rate compared with several important traditional classification techniques.

**Keywords**: Ontology, intrusion detection, security

## 1. Introduction

Undercoffer et al. [1,13,14] have defined the ontology for intrusion detection. Our approach can be seen as a more intelligent way of designing an intrusion detection application for two reasons. First, the domain expert can design his intrusion detection application using his own domain expertise (terminology and concepts from his domain) without having to be an intrusion detection specialist. Second, the characteristics of different network components generated by his intrusion detection application takes into account certain properties from his specification.

The rest of this paper is organized as follows. Section 2 discusses the relevant aspects of detecting intrusion. Section 3 overviews our techniques applied on the detecting intrusion and the idea behind our approach. Our results and analysis are presented in Section 4. Experimental Comparisons among related algorithms on KDD 99 are given in Section 5. Conclusions and future works are presented in Section 6.

## 2. Related Works

We divided the related works into four directions. These techniques include anomaly detection, misuse detection, data mining and ontology-based intrusion detection techniques.

- *Anomaly detection*: it tries to determine whether deviation from an established normal behavior profiles can flagged as an intrusion. However, anomaly-based IDS sometimes set false alarms [12]. Besides, defining and maintaining normal profile is a nontrivial and error prone task, leading to sometimes unacceptable levels of false alarms.
- *Misuse detection* [12]: refers to techniques that use *patterns of known intrusions* or *weak spots* of a system to match and identify intrusions. However, newly invented attacks will likely go undetected, leading to unacceptable false negative error rates.
- *Data Mining*: for a network-based attack detection system [10], JAM [11] uses frequent episode mining [10] which is similar to the sequence mining of data items. It generates the normal usage patterns of a specific node in a network. These patterns are used build a base-classifier that determines the abnormality of the network node.
- *Ontology-based intrusion detection techniques*: Denker *et al.* [4,12] and Raskin *et al.* [1] developed a security ontology for DAML+OIL [3] in order to control access and data integrity of Web resources respectively. In applying ontologies to the problem of intrusion detection, the power and utility of the ontology is not realized by the simple representation of the attributes of the attack. Instead, the power and utility of the ontology is realized by the fact that we can express the relationships between collected data and use those relationships to deduce that the particular data represents an attack of a particular type.

## 3. Our Ontology-based intrusion detection model and algorithm

The ontology [23] can be seen as an abstraction of a computer-based lexicon, thesaurus, glossary or some type of structured vocabulary, suitably extended with knowledge about a given domain. The domain ontology can be considered to be a representation of a domain conceptualization describing possible concepts and relationships between these concepts. Our approach is divided into three phases, namely, a specification phase, a mapping phase and a generation phase (see Figure.1).

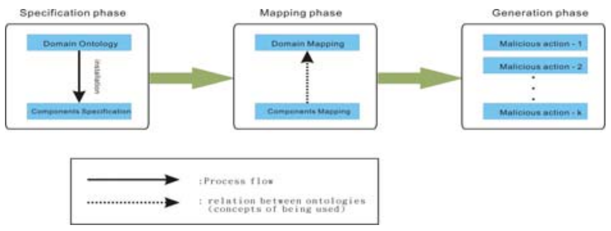


Fig.1: Our ontology-based intrusion detection architecture

## 3.1. Specification phase

The aim of the specification phase is to allow the end-user to define his intrusion detection application at a conceptual level (free from any professional knowledge about intrusion detection or intrusion detection software library) using the domain terminology. Our software uses the DAML+OIL [3] ontology language defined by the DAML ontology [14]. This ontology was already available. It was developed to be the prescriptive reference on the precise SYNtax of the language constructs [14].

## 3.2. Mapping phase

The mapping phase defines the intrusion detection by mapping the concepts defined in the Domain Ontology onto concepts for intrusion detection building components. First, *Meta Mapping* describes how the DAML+OIL concepts are mapped onto the intrusion detection concepts. In this way, we are able to map every concepts descried using the DAML SYNtax onto concepts described by the Intrusion Detection Ontology. Second, *Domain mapping* describes the domain concepts should be represented by Intrusion Detection object types. Therefore, it is an *instantiation* of the *Meta Mapping*. It provides a kind of default mapping for the instances created for a domain concept. Finally, it is *Component mapping*. In the Domain Mapping, overwritten in the default mapping specified for a concept is allowed in Component Mapping.

## 3.3. Generation phase

We have prototyped the logic portion of our system using *DAMLJessKB* [2] reasoning system. *DAMLJessKB* is employed to reason over instances of our IDS that are considered to be suspicious. These suspicious instances are constrained according to our ontology and asserted into the knowledge base. By means of *DAMLJessKB*, we can parse the DAML+OIL statements representing the ontology, converting them into *N-Triples*, and assert them into a knowledge base as rules.

## 4. Our results and analysis

Our experimental dataset was the KDD Cup 1999 Data [6], which contained a wide variety of intrusions simulated in a military network environment. The dataset is about 4 gigabytes of compressed *tcpdump* data of 7 weeks of network traffic. The simulated attacks fell in one of the following four categories: (1)*DOS* (Denial of Service). For example, a SYN flood, smurf, teardrop, ping–of–death, etc.; (2) *R2L* — unauthorized access from a remote machine. For intsance, password guessing; (3) *U2R* — unauthorized access to local superuser privileges by a local unprivileged user, for example, various of buffer overflow attacks; (4) *Probing* — surveillance and other probing for vulnerabilities. For example, port scanning, ping-sweep, etc. In summary, there were a total of 24 attack types and 41 features.

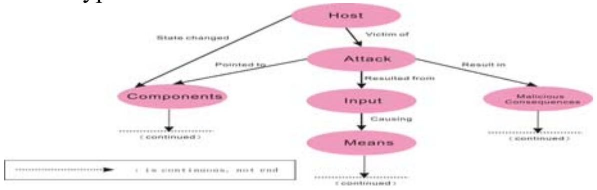


Fig. 2: Our intrusion detection ontology. (*continued*)

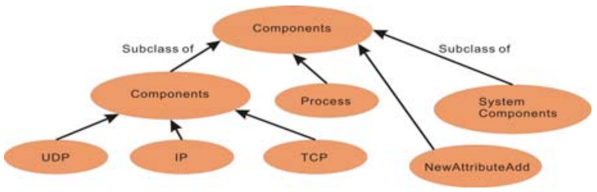


Fig. 3: Our intrusion detection ontology. (*continued*)

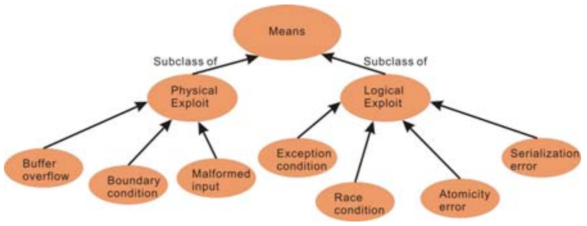


Fig. 4: Our intrusion detection ontology. (*continued*)

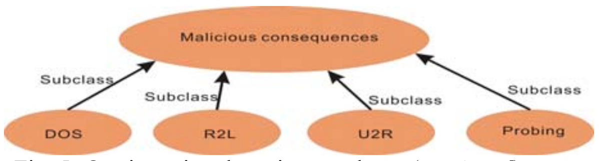


Fig. 5**:** Our intrusion detection ontology. (*continued*)

## 4.2. Our intrusion detection ontology system

We have constructed our ontology in accordance with the results of a detailed analysis of the *CERT/CC Advisories* [7]. Since our dataset is different that of Undercoffer *et al.* [13]. Therefore, our ontology is modified in accordance with our dataset. Our ontology in a high level view is presented in Figure 2~5. The attributes of each class and subclass are not completely depicted because it would make the illustration unwieldy. In all figures, ellipses are used to denote a class, which may have several properties. When two vertices (classes) are connected by a directed edge, the edge represents a property whose domain is denoted by the start of the edge, and whose range is denoted by the end of the edge. An undirected edge between two vertices (classes) indicates that one class is an instance of another class.

## 5. Experimental comparisons among related algorithms on KDD 99

To identify the performance differences on our ontology-based system and other related algorithms, one measure is the *cost per example* that requires two quantities to be defined: *cost matrix* [6] and *confusion matrix* [6]. A *cost matrix* (*C*) is defined by associating classes as labels for the rows and columns of a square matrix: in the current context for the KDD dataset, there are five classes {DoS, R2L, U2R, Probing, Normal}, and therefore the matrix has dimensions of 5x5. An entry at row *i* and column *j*, *C(i,j)*, represents the non-negative cost of misclassifying a pattern belonging to class *i* into class *j*. A *confusion matrix* (*CM*) is similarly defined in that row and column labels are class names: a 5x5 matrix for the KDD dataset. An entry at row *i* and column *j*, *CM(i,j)*, represents the number of misclassified patterns, which originally belong to class *i* yet mistakenly identified as a member of class *j*.

Given the cost matrix as predefined in [6] and the confusion matrix obtained subsequent to an empirical testing process, *cost per example* (*CPE*) was calculated using the following formula.

$$CPE = \frac{1}{N}\sum_{i=1}^{5}\sum_{j=1}^{5} C(i,j) * CM(i,j)$$

where *CM* corresponds to confusion matrix, *C* corresponds to the cost matrix, and *N* represents the number of patterns tested. A lower value for the *cost per example* indicates a better classifier model. In this paper, we compare the performance of classifiers for a given attack category implemented through the *probability of detection* along with the false alarm rate, which are widely accepted as standard measures [13,1,2].

## 5.1. Classification accuracy among the related algorithms

First, three distinct algorithms were tested on the KDD dataset. These algorithms were *K-means* [5], *nearest cluster* algorithm [5] and *C4.5 decision tree* [5]. In addition, *accuracy* reflects the *overall correctness* of the classifier and we will compare these algorithms on *accuracy* based on *CPE* measure. The followings are their CPE measures. *K*-means clustering algorithm achieved the lowest cost per example (= 0.2498) had 8 clusters in each class. The *nearest cluster* algorithm's cost per example for the KDD test data was equal to 0.2603. The cost per example achieved for the best decision tree model was equal to 0.2443. Finally, the cost per example achieved for our ontology-based model was equal to 0.2378.

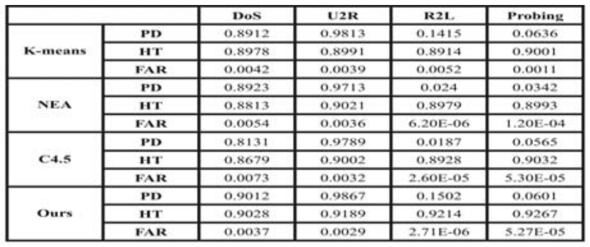
| Standard Metrics | | Predicted Connection Label | |
|---|---|---|---|
| | | Normal | Intrusions (Attacks) |
| Actual Connection Label | Normal | True Negative (TN) | False Alarm Rate (FAR) |
| | Intrusions (Attacks) | False Negative (FN) | Correctly Detected Intrusions -- Hit Rate (HT) |

Table 1. Standard metrics for evaluations of intrusions (attacks)

## 5.2. Performance Comparisons among the related algorithms

In this subsection, we will explain our measure and terminology. A *false positive* (*false alarm*) occurs when an event is predicted as intrusive but it is in fact normal. Table 1 lists the four possibilities, where a specific prediction rule is invoked. We also define the signal threshold for events. If the target value is greater than the given signal threshold, the data record is signaled as intrusive, considered as normal otherwise. The *hit rate* (HT) is the ration of the number of hits to the total number of the truly intrusive data records. The *false alarm rate* (FAR) is the ration of false alarms to the total number of the truly normal data records. For a given algorithm, its *probability of detection* (PD) [15], *hit rate* (HT) and *false alarm rate* (FAR) performance on a specific attack category was recorded. Experimental results are presented in Table 2. All above parameters are indicated for each algorithm and each attack category.

Table 2 shows that for a given attack category, certain algorithms demonstrate superior detection performance compared to others. In case of DoS category, our algorithm detected more than 90% of attack records and others did not. It means that ontology-based model may be a good candidate for intrusion detection system. In other categories, the ontology-based model always detected more than 91.8% of attack records.



| | | DoS | U2R | R2L | Probing |
|---|---|---|---|---|---|
| K-means | PD | 0.8912 | 0.9813 | 0.1415 | 0.0636 |
| | HT | 0.8978 | 0.8991 | 0.8914 | 0.9001 |
| | FAR | 0.0042 | 0.0039 | 0.0052 | 0.0011 |
| NEA | PD | 0.8923 | 0.9713 | 0.024 | 0.0342 |
| | HT | 0.8813 | 0.9021 | 0.8979 | 0.8993 |
| | FAR | 0.0054 | 0.0036 | 6.20E-06 | 1.20E-04 |
| C4.5 | PD | 0.8131 | 0.9789 | 0.0187 | 0.0565 |
| | HT | 0.8679 | 0.9002 | 0.8928 | 0.9032 |
| | FAR | 0.0073 | 0.0032 | 2.60E-05 | 5.30E-05 |
| Ours | PD | 0.9012 | 0.9867 | 0.1502 | 0.0601 |
| | HT | 0.9028 | 0.9189 | 0.9214 | 0.9267 |
| | FAR | 0.0037 | 0.0029 | 2.71E-06 | 5.27E-05 |

Table 2. PD, HT and FAR for various algorithms.

## 6. Conclusions and future work

In this paper, a new approach for designing intrusion detection applications has been described. This approach allows user to model an intrusion detection application at the conceptual level and in terms of concepts from the application domain. This is realized by using a Domain Ontology, which captures the domain knowledge. This approach has the following advantages. First, it is better fit requirements of the end-user and customers; Second, the development process can be shortened; Third, fast prototyping is possible; Fourth, better communication between the intrusion detection specialists; and finally, domain knowledge can be exploited. In addition, the use of ontologies allows us to incorporate some intelligence and therefore, intelligent reasoning about the intrusion detection itself becomes possible.

A simulation study was performed to assess the performance of related algorithms on the KDD 1999 Cup intrusion detection dataset. Simulation results demonstrated that for four given attack categories the ontology-based mode performed better. Furthermore, reduction in cost per example was also achieved using the ontology-based model.

More complicated experiments are on designing, such as intruder's communication in encrypted commands and new intrusion evolved from old one. These researches are some part of our future work. Also, we expect more conspicuous improvements shown by our prototype.

## References

[1] V. Raskin, C.F. Hempelmann, K.E. Triezenberg, Nirenburg, "Ontology in Information Security: A Useful Theoretical Foundation and Methodological Tool," *Proceedings of the 2001 Workshop on New Security Paradigms (NSPW-2001)*, pp. 53-59, 2001.

[2] DAMLJessKB Available at: http://edge.cs.drexel.edu/assemblies/software/damljesskb/, October, 2002.

[3] DAML+OIL. Available at: http://www.daml.org/2000/12/daml+oil.daml

[4] G. Denker, L. Kagal, T. Finin, M. Paolucci, K. Sycara, "Security for DAML Web Services: Annotation and Matchmaking," *The Semantic Web (ISWC 2003)*, LNCS 2870, Springer, pp. 335-350, 2003.

[5] R.O. Duda, P.E. Hart, *Pattern Classification and Scene Analysis*, New York, Wiley, 1973.

[6] C. Elkan, "Results of the KDD'99 Classifier Learning," *SIGKDD Explorations, ACM SIGMOD*, Vol. 1, pp. 63-64, 2000.

[7] A. Joshi, J. Undercoffer, *Data Mining, Semantics and Intrusion Detection: What to Dig for and Where to Find it*, MIT Press, 2003.

[8] W. Krueger, J. Nilsson, T. Oates, T. Finin, "Automatically Generated DAML Markup for Semi-structured Documents," *National Science Foundation Workshop on Next Generation Data Mining, Baltimore*, November 2002.

[9] Y.B. Reddyl, R. Guha, "Intrusion Detection using Data Mining Techniques," *Artificial Intelligence and Applications (AIA-2004)*, pp. 232-241, 2004.

[10] S. Stolfo, A.L. Prodromidis, S. Tselepis, W. Lee, D.W. Fan, P.K. Chan, "JAM: Java Agents for Meta-Learning over Distributed Databases," *Proceeding of KDD-97*, pp. 74-81,1997.

[12] N. Tuck, T. Sherwood, B. Calder, G. Varghese, "Deterministic Memory-Efficient String matching algorithms for Intrusion Detection," *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, pp. 2628-2639, 2004.

[13] J. Undercoffer, J., Pinkston, A. Joshi, T. Finin, "Target-Centric Ontology for Intrusion Detection," *IJCAI Workshop on Ontologies and Distributed Systems (IJCAI'03)*, August, 2003.

[14] O. Lassila, R. Swick, *Resource description framework: model and SYNtax specification*, W3C Recommendation, World Wide Web Consortium, Cambridge, 1999.

[15] A. Iranli, H. Fatemi, M. Pedram, "Lifetime-aware Intrusion Detection Under Safeguarding Constraints," *Fourth International Symposium on Information Processing in Sensor Networks, 2005 ( IPSN 2005)*, pp. 189-194, 2005.