International Journal of Innovative Computing, Information and Control
Volume **8**, Number **4**, April **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 2865–2874

APPLICATION OF TYPE-2 FUZZY LOGIC TO RULE-BASED INTRUSION ALERT CORRELATION DETECTION

Chenn-Jung Huang[1,*], Kai-Wen Hu[1], Heng-Ming Chen[2], Tao-Ku Chang[1]
Yun-Cheng Luo[4] and Yih-Jhe Lien[3]

[1]Department of Computer Science and Information Engineering
[2]Department of Electrical Engineering
[3]Institute of Electronics Engineering
National Dong Hwa University
No. 1, Sec. 2, Da Hsueh Rd., Shoufeng, Hualien 97401, Taiwan
*Corresponding author: cjhuang@mail.ndhu.edu.tw

[4]Department of Computer Science and Information Engineering
National Tsing Hua University
No. 101, Sec. 2, Kuang-Fu Rd., Hsinchu 30013, Taiwan

Received December 2010; revised April 2011

Abstract. *An intrusion detection system (IDS) is a security layer that is used to discover ongoing intrusive attacks and anomalous activities in information systems, which means usually working in a dynamically changing environment. Although increasing attention to IDSs is evident in the literature, network security administrators are still faced with the task of analyzing enormous numbers of alerts produced from different event streams. The intrusion detection model therefore needs to be continuously tuned, in order to reduce correlative alerts and help the administrator to accurately determine critical attacks. In this work, an alert correlation detection module is proposed to analyze the alerts produced by IDSs, providing a more succinct and comprehensive view of intrusions. An automatically-tuned IDS rule-generation module that is based on a type-2 fuzzy logic technique is used to block highly correlative alerts. The experimental results reveal that the proposed model is effective in achieving alert reduction and abstraction.*

Keywords: Intrusion detection system, Intrusion correlation, Alert reduction, IDS rule tuning, Adaptive tuning

1. Introduction.

Recently, ubiquitous infrastructure such as high-speed backbones and local area networks has evolved in modern network topologies, providing end-users with bandwidths that are several multiples larger than those that were available a few years ago. Obviously, the Internet has become a more complex infrastructure which poses difficulties in terms of management for governments, companies, institutions, and millions of everyday users.

The inspection and security monitoring of network infrastructure is for the most part performed using Intrusion Detection Systems (IDSs) [1-3], including network-based IDS (N-IDS) [4,5] and host-based IDS (H-IDS) [6,7]. An IDS is a security layer used to discover ongoing intrusive attacks and anomalous activities in information systems. These systems monitor information about the activities performed in computer systems and networks, looking for symptoms of malicious behavior. Attacks against a system expose themselves in terms of being events distinguished in terms of differing nature and level of granularity.

In recent years, numbers of different intrusion detection systems that perform in particular domains, such as hosts or networks, or other specific environments, have been developed. Even with more IDSs being developed, network security administrators are still confronted with the task of analyzing enormous numbers of alerts resulting from the analysis of different event streams. In addition, IDSs are not perfectly refined, and may produce both false positives and non-relevant positives. Clearly, efficient tools and techniques are essential in order for administrators to aggregate and combine the output alerts of multiple IDSs, to filter out spurious or incorrect alerts, and to provide a succinct, high-level view of the security situation of the protected network. Notably, the correlation issue needs to be addressed, with a view to providing high-level reasoning that goes beyond low-level sensor capabilities.

In this work, we propose adaptive alert correlation detection and rule tuning algorithm for IDSs to reduce the alerts for the same kind of attack and alleviate the loading of network equipment while processing the correlative alerts. The unique feature of our work is that we use a type-2 fuzzy logic-based rule tuning module to improve response time and provide a real-time protection. Traditional IDSs cannot make a real-time response for unknown alerts because the accuracy of the alerts depends on the expert knowledge. Hence, not only may their false negatives increase sometimes, but also the reduction rate has no significant improvement.

The remainder of this paper is organized as follows. Section 2 presents related work on alert correlation. Section 3 presents the overall architecture of our correlation process and our IDS rules tuning module. Section 4 presents the results of applying the rules generated from the IDS tuning module. Finally, Section 5 draws conclusions and outlines future work.

2. Related Work.

In recent years, researchers and vendors have proposed alert correlation mechanisms to reduce the number of alerts that need to be manually analyzed, and the relevance and abstraction level of the IDS resulting reports have been correspondingly increased. Even though the goals of alert correlation seem to be well-defined, the correlation approaches proposed in the literature have focused on a variety of purposes in terms of the correlation process, making it difficult to compare the results of each solution [8,9]. A hierarchical distributed alert correlation model was proposed to reduce network transfer loading and the computation of alert correlation [10]. However, there are some limitations of the correlation approaches applied in the proposed model. For example, the time setting of the aggregation algorithm and the number of correlating nodes are uncertain, and the repository of causal correlation needs to be updated. Although Yi et al. [11] presented a deductive approach, using an alert correlation algorithm to perform deductive analysis on the original alerts generated by the IDS in a backward fashion, the validity of the proposed algorithm is not verified.

In spite of the fact that numerous correlation approaches have been proposed, there is still no consensus on *what* the ideal correlation process should be, or *how* it should be implemented and evaluated. For example, existing correlation approaches operate based on only partial aspects of the correlation process, such as the fusion of alerts that are generated by different IDSs in response to a single attack. In this work, we follow the comprehensive approach to intrusion detection proposed by Valeur et al. [12]. Comprehensive correlation approaches have been designed to be as complete as possible, and to include all the aspects of correlation discussed in previous research efforts. We first build a multi-stage framework that implements the correlation process, and then analyze a number of data sets to produce the specific parameters of each individual alert for the IDS tuning module, allowing for the lifetime of temporary rules to be intelligently determined.

3. Alert Correlation Detection and the Rule Tuning Module.

In order to generate temporary blocking rules for ignoring continually correlative alerts from highly related

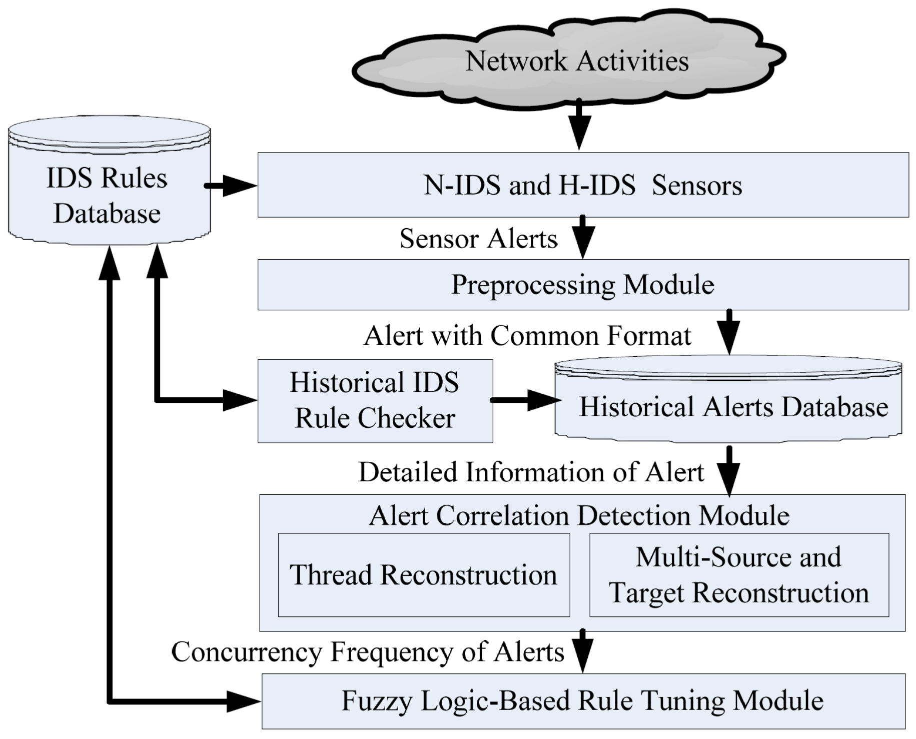
Network Activities
IDS Rules Database
N-IDS and H-IDS Sensors
Sensor Alerts
Preprocessing Module
Alert with Common Format
Historical IDS Rule Checker
Historical Alerts Database
Detailed Information of Alert
Alert Correlation Detection Module
Thread Reconstruction
Multi-Source and Target Reconstruction
Concurrency Frequency of Alerts
Fuzzy Logic-Based Rule Tuning Module

FIGURE 1. The architecture of the alert correlation detection and rule tuning model for IDS

attacks within a specific time interval, we propose the method of adapting parameters, which are obtained from the correlation module and the lifetimes of historical rules in the IDS tuning module, to intelligently generate temporary blocking rules, which can be further applied in the IDS rule database for N-IDS and H-IDS. Referring to the comprehensive approach to intrusion detection proposed by Valeur et al. [12], we propose a multi-stage approach, which includes five main components, namely: N-IDS/H-IDS sensors, the preprocessing module, the alert correlation detection module, the historical IDS rule checker, and the rule tuning module. The detailed architecture of our intelligent tuning system is shown in Figure 1. In the initial phase, the N-IDS and H-IDS sensors collect each alert from both hosts and networks, and then send these collected alerts into the preprocessing module to undergo normalization into a standard format that is realized by the following related components, such that all alert attributes can be translated into meaningful values. Notably, a historical IDS rule checker is used in this model to automatically examine all the attached temporary rules, comparing them against the central historical alert database. The aim of the rule checker is to remove any out-of-date rules recorded in the database. A rule saved in the database is maintained if some correlative attack alert appears during this last examination. Otherwise, the rule is removed before the start of the next round of the examination. The next two components are the thread reconstruction component and the multi-source and target reconstruction component. These are both found in the alert correlation detection module, which deals with different types of correlation alerts and sends the alert occurrence frequency and the percentage of the specific attack alert as the parameters to the next component. The final module is the fuzzy logic-based rule tuning module which receives the parameters from the alert correlation detection module and determines the new lifetime for each temporary rule to be recorded in the historical database. The advantages of using type-2 fuzzy logic here are its successful application in many fields, and the high-speed and low cost of the fuzzy chips that are available nowadays. A detailed description of our proposed modules and their components is addressed in the following subsections.

3.1. **Preprocessing module.** The correlation process may receive alerts from different IDSs, and these alerts can be encoded in different formats. Firstly, the preprocessing procedure translates all attributes of each sensor alert into a common format [13] and supplies, as accurately as possible, missing alert attributes such as source IP, target IP start time, end time, and service port, for later use by correlation components. Then, the alerts with completely detailed information will be stored in the central historical alert database for further reference, in the alert correlation module, for example.

When the number of alerts surpasses the threshold, the alert reduction module is launched to reduce the number of correlative alerts and verify the attack behavior signaled by alerts. Notably, the initial threshold is defined with an initial value when the intelligent IDS tuning system launches for the first time. After the system finishes a whole round of examination, the threshold is dynamically adjusted by the occurrence frequency of each attack alert. The occurrence frequency of each attack alert is calculated so that the system can refer to the first quartile and third quartile of these frequencies to determine the new threshold that will be used for next round of examination. The computation of the new threshold is expressed by,

$$\begin{cases} N_{Threshold} = V \times (1+P), & P \geq \dfrac{|t_{start} - t_{end}|}{V} \\ N_{Threshold} = V \times (1-P), & P < \dfrac{|t_{start} - t_{end}|}{V} \end{cases}, \tag{1}$$

where V represents the threshold value used in the current round of examination, t_{start} is the time that the alert first appears, and t_{end} is the time that the specific attack alert appears at the end of the current round of examination. P is computed by,

$$P = \left| \frac{1}{Q_3} - \frac{1}{Q_1} \right|. \tag{2}$$

Here Q_1 and Q_3 represent the first quartile and the third quartile of occurrence frequency, respectively.

3.2. **Alert correlation detection module.** The alert correlation module attempts to determine the correlativity of alerts produced from N-IDS and H-IDS. This module includes subcomponents such as the thread reconstruction module and the multi-source and target reconstruction module, to deal with different types of correlation alerts in each individual module, and sends the alert occurrence frequency as the parameter for the IDS tuning module.

3.2.1. *Thread reconstruction.* The aim of thread reconstruction is to correlate alerts that are caused by a single attacker who tries different intrusion activities against a certain program or runs the same intrusion activity multiple times to try out correct values for certain parameters such as the offsets and memory addresses for a buffer overflow. An attack thread comprises a series of alerts that refer to attacks launched by one attacker against a single target. Attack threads are formed by merging alerts with equivalent source and target attributes which occur in a certain temporary period. Two alerts can be considered as a match if the source and target attributes are equivalent.

Our system calculates two parameters and passes them to the IDS tuning module. One is the occurrence frequency of the specific attack alert in a certain period, and the other is the percentage value associated with the specific attack alert, compared against all other attack alerts. The occurrence frequency of the specific attack is determined by,

$$F_{specific} = \frac{T_{specific}}{|t_{start} - t_{end}|}, \tag{3}$$

where $T_{specific}$ is the total number of the specific attack alerts, t_{start} is the time that the alert first appears and t_{end} is the time that the specific attack alert appears at the end of the current round of examination.

The percentage value associated with the specific attack alert is calculated by,

$$P_{specific} = \frac{T_{specific}}{T_{total}} \tag{4}$$

where T_{total} is the total count of the attack alerts.

3.2.2. *Multi-source and target reconstruction.* The aim of the multi-source and target reconstruction components is to identify hosts that are either the source attacker or the target victim machine of a high number of attacks. We aggregate the alerts associated with single hosts attacking multiple victims, referred as the one2many scenario, and single victims that are targeted by multiple attackers, referred as the many2one scenario. The number of alerts caused by distributed denial-of-service (DDoS) attacks and large-scale scans is effectively reduced by this correlation component.

In the one2many scenario, the number of different targets is recorded for each attack source in the examination period. The system identifies attacks from the same attack source, and groups them into a single set. The system then calculates the number of different distensions in order to calculate the occurrence frequency and percentage value associated with the specific attack alert. The many2one scenario operates in a similar way, with the only difference being that the roles of the attacker and the victim are reversed. A many2one attack is classified as a DDoS attack when the attacks are against the same victim.

3.3. **The IDS rule tuning module.** A type-2 fuzzy logic-based rule tuning module is employed to determine the new lifetime for each temporary rule recorded in the historical database. The rule tuning module depends on three attributes, specifically: the occurrence frequency, the percentage value associated with the specific attack alert compared against the total of all attack alerts, and the time interval between the start and the end of an examination. The motivation behind using type-2 fuzzy logic here is that fuzzy logic has been successfully applied in many fields, from control theory to artificial intelligence. There are lots of variations on VLSI chips that allow fuzzy inferences to be hardware-computed, and high-speed/low cost fuzzy chips have been introduced recently, thus making the implementation of fuzzy logic via hardware feasible nowadays [21,22].

The concept of a type-2 fuzzy set was introduced by Zadeh as an extension of an ordinary fuzzy set [14,15]. In an ordinary fuzzy set, the membership value for each fuzzy membership function is a crisp number in $[0, 1]$, whereas the membership value for each element of a type-2 fuzzy set is also a fuzzy set in $[0, 1]$.

Figure 2 illustrates the architecture of a type-2 fuzzy logic inference system, which consists of five components, including a fuzzifier, rule control, inference engine, type reducer, and a defuzzifier. Notably, a type-2 fuzzy logic system [16] differs from the ordinary fuzzy set mainly in the insertion of the process of type reduction right before defuzzification in the ordinary fuzzy logic system. The basic functions of the components employed in the type-2 fuzzy logic inference system are described as follows.

- **Fuzzifier**

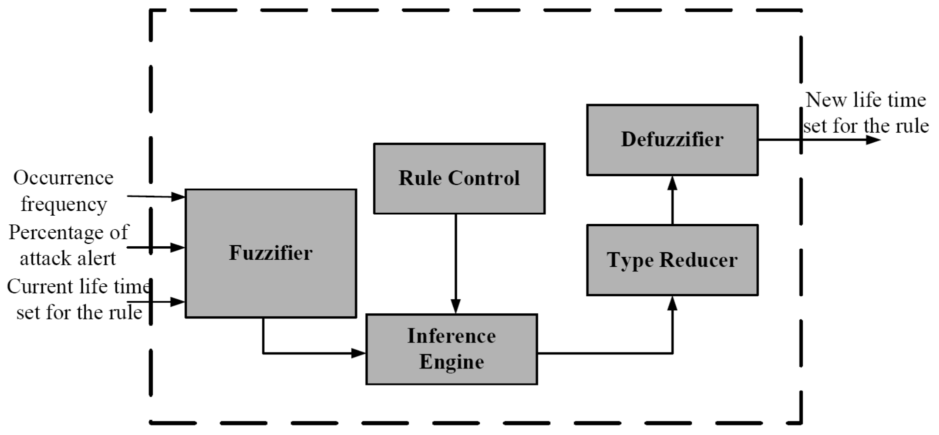
Occurrence frequency
Percentage of attack alert
Current life time set for the rule
Fuzzifier
Rule Control
Inference Engine
Type Reducer
Defuzzifier
New life time set for the rule

FIGURE 2. The architecture of a type-2 fuzzy logic system

The function of the fuzzifier is to transform the set of crisp values which are taken as the input data of a fuzzy logic system into a set of fuzzy values. The Gaussian primary membership functions can be expressed by,

$$\mu_{\tilde{A}}(x) = \exp\left[-\frac{1}{2}\left(\frac{x-m}{\sigma}\right)^2\right], \quad m \in [m_1, m_2], \tag{5}$$

where m is mean and σ represents standard deviation.

A type-2 fuzzy set utilizes a Gaussian primary membership function. Notably, the embedded fuzzy sets of a type-2 fuzzy set are ordinary fuzzy sets, and the footprint of uncertainty (FOU) can be used to achieve blurring of the membership grade for each element of type-2 fuzzy set so that the linguistic value for each membership function is not restricted to a crisp value. A type-2 fuzzy set is completely characterized by its 2-D FOU that is bound by a lower membership function (LMF) and an upper membership function (UMF) expressed by the following two equations, respectively,

$$\underline{\mu}_{\tilde{A}}(x) = \begin{cases} N(m_2, \sigma; x), & x \le \dfrac{m_1+m_2}{2} \\ N(m_1, \sigma; x), & x > \dfrac{m_1+m_2}{2} \end{cases}, \tag{6}$$

$$\bar{\mu}_{\tilde{A}}(x) = \begin{cases} N(m_{1,}\sigma; x), & x < m_1 \\ 1, & m_1 \le x \le m_2 \\ N(m_2, \sigma; x), & x > m_2 \end{cases}, \tag{7}$$

where $N(m, \sigma; x) = \exp\left[-\frac{1}{2}(\frac{x-m}{\sigma})^2\right]$.

- **Rule Control**

It is assumed that the input data of the type-2 fuzzy system is $x_1 \in X_1$, $x_2 \in X_2$, ... and $x_p \in X_p$, and fuzzy output sets is $y \in Y$. The rules of the type-2 fuzzy system are denoted as follows:

$$R^i : \text{IF } x_1 \text{ is } \tilde{F}_1^i \text{ and } x_2 \text{ is } \tilde{F}_2^i \ldots \text{ and } x_p \text{ is } \tilde{F}_p^i \text{ then } y \text{ is } \tilde{G}^i$$

where $\tilde{F}_j^i = [\underline{\mu}_{\tilde{A}}(x), \bar{\mu}_{\tilde{A}}(x)]$ denotes the jth antecedent of rule i and $\tilde{G}^i$ indicates the consequent of rule i.

- **Inference Engine**

The inference engine is the key component of a fuzzy logic inference system. The inference engine calculates a firing level, which is the weight of each fired rule, based on the input and the antecedents of the rules, and then applies these firing levels to the consequent fuzzy sets. In a type-2 fuzzy logic system, each fuzzy rule has its own firing strength, and membership function values with different input parameters in a fuzzy rule proceed to t-norm operation to obtain firing strengths for the consequent. In general, minimum and product operators are the two most commonly used t-norm operations for a fuzzy logic inference system. The upper and lower firing levels for the consequent, $\tilde{G}^i$, can be expressed by,

$$\underline{f}^i(x') = \min\{\underline{\mu}_{\tilde{F}_1}(x_1), \underline{\mu}_{\tilde{F}_2}(x_2), \underline{\mu}_{\tilde{F}_3}(x_3)\}, \tag{8}$$

$$\bar{f}^i(x') = \min\{\bar{\mu}_{\tilde{F}_1}(x_1), \bar{\mu}_{\tilde{F}_2}(x_2), \bar{\mu}_{\tilde{F}_3}(x_3)\}, \tag{9}$$

where $\underline{\mu}_{\tilde{F}_i}(x_i)$ is the lower membership function value when $x = x_i$, and $\bar{\mu}_{\tilde{F}_i}(x_i)$ is the upper membership function value when $x = x_i$. Notably, a minimum operator is adopted in this work.

- **Type Reducer**

In a type-2 fuzzy logic inference system, the output set corresponding to each rule is a type-2 set, as well. The center-of-sets type-reduction method [23] is employed to reduce a type-2 fuzzy set to a type-1 fuzzy set $[y_l, y_r]$ in order to achieve computational efficiency. The two end points, y_l and y_r, determine the interval set as shown above, and correspond to the centroid of the type-2 interval consequent set $\tilde{G}^i$ of the ith rule. It can be shown [24] that the computation of y_l and y_r is expressed by,

$$[y_l, y_r] = \left[\frac{\underline{y_l} + \min\left\{\frac{\sum_{i=1}^M \underline{f^i} y_l^i}{\sum_{i=1}^M \underline{f^i}}, \frac{\sum_{i=1}^M \overline{f^i} y_l^i}{\sum_{i=1}^M \overline{f^i}}\right\}}{2}, \frac{\overline{y_r} + \max\left\{\frac{\sum_{i=1}^M \overline{f^i} y_r^i}{\sum_{i=1}^M \overline{f^i}}, \frac{\sum_{i=1}^M \underline{f^i} y_r^i}{\sum_{i=1}^M \underline{f^i}}\right\}}{2}\right], \tag{10}$$

$$\underline{y_l} = \min\left\{\frac{\sum\limits_{i=1}^M \underline{f^i} y_l^i}{\sum\limits_{i=1}^M \underline{f^i}}, \frac{\sum\limits_{i=1}^M \overline{f^i} y_l^i}{\sum\limits_{i=1}^M \overline{f^i}}\right\} - \left[\frac{\sum\limits_{i=1}^M (\overline{f^i} - \underline{f^i})}{\sum\limits_{i=1}^M \overline{f^i} \sum\limits_{i=1}^M \underline{f^i}} \times \frac{\sum\limits_{i=1}^M \underline{f^i}(y_l^i - y_l^1) \sum\limits_{i=1}^M \overline{f^i}(y_l^M - y_l^i)}{\sum\limits_{i=1}^M \underline{f^i}(y_l^i - y_l^1) + \sum\limits_{i=1}^M \overline{f^i}(y_l^M - y_l^i)}\right], \tag{11}$$

$$\overline{y_r}(x) = \max\left\{\frac{\sum\limits_{i=1}^M \overline{f^i} y_r^i}{\sum\limits_{i=1}^M \overline{f^i}}, \frac{\sum\limits_{i=1}^M \underline{f^i} y_r^i}{\sum\limits_{i=1}^M \underline{f^i}}\right\} + \left[\frac{\sum\limits_{i=1}^M (\overline{f^i} - \underline{f^i})}{\sum\limits_{i=1}^M \overline{f^i} \sum\limits_{i=1}^M \underline{f^i}} \times \frac{\sum\limits_{i=1}^M \overline{f^i}(y_r^i - y_r^1) \sum\limits_{i=1}^M \underline{f^i}(y_r^M - y_r^i)}{\sum\limits_{i=1}^M \overline{f^i}(y_r^i - y_r^1) + \sum\limits_{i=1}^M \underline{f^i}(y_r^M - y_r^i)}\right] \tag{12}$$

where $\underline{f^i}$ and $\overline{f^i}$ are firing levels as given in Equations (8) and (9), $i = 1, \ldots, M$, M is the number of fired rules, and y_l^i and y_r^i denote the end points of the centroids of the consequent of the ith rule. Notably, y_l^i and y_r^i can be derived by using the KM algorithms [25].

- **Defuzzifier**

Each type-reduced set is defuzzified to get a crisp output from the type-2 fuzzy logic inference system. The defuzzified value is the average of type-reduction set, calculated as follows:

$$y(x) = \frac{y_l + y_r}{2}. \tag{13}$$

The fuzzy linguistic variables used for the input membership function are "low", "medium" and "high".

There are 27 inference rules generated in the Mamdani fuzzy inference system. The non-fuzzy output of the Mamdani fuzzy inference system can then be obtained by computing the centroid of area for the aggregated output membership function.

4. **Experimental Results.** To evaluate the effectiveness of the proposed algorithm, a series of simulations were conducted to compare the proposed work and a representative intrusion detection alert correlation scheme presented by Valeur et al. [12]. Snort [6] and the VRT certified rules for Snort were employed as the N-IDS sensor. The datasets adopted in the experiments include Defcon9 [17], MIT-LL1999, MIT-LL2000 [18] and Treasure Hunt [17].

The Defcon9 dataset is commonly used in IDS evaluations. Defcon is a yearly underground hacking conference. The conference includes a hacker competition called Capture the Flag (CTF). During the competition, all network traffic was recorded and then made publicly available. The Defcon9 data set has several properties that make it very different from “real world” network traffic; specifically, it contains an artificially high amount of attack traffic, no background traffic, and only a small number of IP addresses.

DARPA has sponsored a series of intrusion detection evaluation efforts for the past six years, including the evaluation carried out by the MIT Lincoln Laboratory in 1998, and a more systematic evaluation conducted by Durst et al. [19]. In both evaluations, a number of attacks, including port scans, remote compromises, local privilege escalation attempts, and denial-of-service attacks, were carried out on a testbed network. The relevant auditing events, such as network traffic and OS audit records, were collected at a number of critical points. The data sets they produced became a reference point in the evaluation of intrusion detection functionality [20].

The Treasure Hunt data set was generated during a Cyber Treasure Hunt competition [17]. The intention of the Treasure Hunt was to break into a simulated payroll system and perform unauthorized money transfer transactions. The tasks and the execution of this exercise were designed to support the creation of event streams that contain a sequence of attacks that are part of an overall plan to achieve a specific goal.

The reduction rate for the four data sets achieved by the proposed scheme is given in Table 1. Apparently, the reduction rate is heavily dependent on the data set. Several properties of a data set play an important role in determining how much reduction is achieved. For example, the characteristics of the attacks in the Treasure Hunt data set reflect the fact that the brute-force password guessing attack caused a very high reduction rate in the thread reconstruction step. The reduction rates for Defcon9, MIT-LL1999 and Treasure Hunt are significant in the proposed model, because our algorithm is capable of capturing the properties of these three data sets. On the other hand, the reduction rate of the MIT-LL2000 data set is the lowest.

Table 1. The reduction rates for four data sets

	Defcon9	MIT-LL1999	MIT-LL2000	Treasure Hunt
Input Alerts	6209601	40053	1206	2680652
Output Alerts	4033	1730	508	212
Reduction Rate	99.94%	95.68%	57.88%	99.99%

The comparison of the achievement between the proposed work and the scheme presented by Valeur et al. [12] is listed in Table 2. Our proposed scheme adopted both H-IDS and N-IDS sensors in the initial phase to detect the possible alerts in all domains, allowing false positives and non-relevant positives to be decreased significantly. Contrast this with the intrusion detection alert correlation presented by Valeur et al. which has some limitations that need improving. The latter relies heavily not only on a priori expert knowledge, but also on each IDS to trigger an alarm for each step of the attack. In the process, Valeur et al.’s correlation may encounter the problems of insufficient ability to exploit mutations, false negatives for novel attacks, decreased performance owning to bad sensor placement, and packet loss caused by sensor overload. As it stands, the type-2 fuzzy logic-based rule tuning module proposed in our work can improve these limitations, in terms of providing a comprehensive view of the network environment and allowing for an effective response and real-time protection. As a result, our scheme exhibits better performance in terms of the reduction rate for the four data sets, compared with Valeur et al.'s scheme.

Table 2. The reduction rate comparison between two schemes

	Defcon9	MIT-LL1999	MIT-LL2000	Treasure Hunt
Our scheme	99.94%	95.68%	57.88%	99.99%
Valeur et al.'s scheme	96.81%	80.87%	53.00%	99.96%

5. Conclusion and Future work.

Our experimental results show that our system can effectively reduce the number of correlated attack alerts. Besides the aforementioned improvement when compared against Valeur et al.'s work, another major advantage of the proposed work is the ability of the model to alleviate the loading of network equipment in processing the correlative alerts. Temporary rules are added to the IDS sensors to effectively drop attack packets and block attack alerts. The chance of launching the correlation module is thus significantly decreased. Notably, the attack alerts are summarized in a statistical report, including the attack categories, starting times and duration, in our alert correlation module. There is, however, still something we could improve in future work. For example, the N-IDS being used to detect alerts happening in the network domain may increase the system computation complexity. Hence, we may design a smarter N-IDS sensor to determine and analyze the risk of each attack category against the victim machines, and thus produce more comprehensive information about the monitored network area. By doing so, the management complexity of the network domain could be restricted to within well-defined margins.

Acknowledgment. The authors would like to thank the National Science Council of Taiwan for financially supporting this research under Contract No. NSC 99-2511-S-259-008-MY3 and 99-2220-E-259-002.

REFERENCES

[1] C. Lin and J. Leneutre, A game theoretical framework on intrusion detection in heterogeneous networks, *IEEE Transactions on Information Forensics and Security*, vol.4, no.2, pp.165-178, 2009.

[2] R. C. Chen and S. P. Chen, Intrusion detection using a hybrid support vector machine based on entropy and TF-IDF, *International Journal of Innovative Computing, Information and Control*, vol.4, no.2, pp.413-424, 2008.

[3] H. Zhu, H. Kai, K. Eguchi, Z. Guo and J. Wang, Application of BPNN in classification of time intervals for intelligent intrusion detection decision response system, *International Journal of Innovative Computing, Information and Control*, vol.4, no.10, pp.2483-2491, 2008.

[4] *Snort – The Open Source Network Intrusion Detection System*, http://www.snort.org, 2004.

[5] Z. Jiong, M. Zulkernine and A. Haque, Random-forests-based network intrusion detection systems, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol.38, no.5, pp.649-659, 2008.

[6] *Nessus Vulnerabilty Scanner*, http://www.nessus.org/, 2004.

[7] J. K. Hu, X. G. Yu, D. Qiu and H. H. Chen, A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection, *IEEE, Network*, vol.23, no.1, pp.42-47, 2009.

[8] P. Ning, Y. Cui and D. S. Reeves, Constructing attack scenarios through correlation of intrusion alerts, *Proc. of ACM Conf. Computer and Comm. Security*, pp.245-254, 2002.

[9] D. Andersson, M. Fong and A. Valdes, Heterogeneous sensor correlation: A case study of live traffic analysis, *Proc. of the 3rd Ann. IEEE Information Assurance Workshop*, 2002.

[10] D. Tian, C. Hu, Q. Yang and J. Wang, Hierarchical distributed alert correlation model, *Proc. of the 5th Int'l Conf. Information Assurance and Security*, vol.2, pp.765-768, 2009.

[11] P. Yi, H. Xing, Y. Wu and J. Cai, Alert correlation through results tracing back to reasons, *Proc. of WRI Int'l Conf. Communications and Mobile Computing*, vol.3, pp.465-469, 2009.

[12] F. Valeur, G. Vigna, C. Kruegel and R. A. Kemmerer, Comprehensive approach to intrusion detection alert correlation, *IEEE Transactions on Dependable and Secure Computing*, vol.1, no.3, 2004.

[13] D. Curry and H. Debar, Intrusion detection message exchange format: Extensible markup language (XML) document type definition, *draft-ietf-idwg-idmef-xml-10.txt*, 2003.

[14] L. A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning, *Information Science*, vol.8, pp.199-249, 1975.

[15] T. Wang, Y. Chen and S. Tong, Fuzzy reasoning models and algorithms on type-2 fuzzy sets, *International Journal of Innovative Computing, Information and Control*, vol.4, no.10, pp.2451-2460, 2008.

[16] T. C. Lin, Analog circuit fault diagnosis under parameter variations based on type-2 fuzzy logic systems, *International Journal of Innovative Computing, Information and Control*, vol.6, no.5, pp.2137-2158, 2010.

[17] UCSB Reliable Software Group, *Collection of Intrusion Detection Data Sets*, http://www.cs.ucsb.edu/rsg/datasets/, 2004.

[18] MIT Lincoln Laboratory, *Lincoln Lab Data Sets*, http://www.ll.mit.edu/IST/ideval/data/data_index.html, 2000.

[19] R. Durst, T. Champion, B. Witten, E. Miller and L. Spagnuolo, Addendum to testing and evaluating computer intrusion detection systems, *Comm. ACM*, vol.42, no.9, pp.15, 1999.

[20] J. McHugh, Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln laboratory, *ACM Trans. Information and System Security*, vol.3, no.4, 2000.

[21] K. Daijin, An implementation of fuzzy logic controller on the reconfigurable FPGA system, *IEEE Trans. Industrial Electronics*, vol.47, no.3, pp.703-715, 2000.

[22] M. N. Uddin, T. S. Radwan and M. A. Rahman, Performances of fuzzy-logic-based indirect vector control for induction motor drive, *IEEE Trans. Industry Applications*, vol.38, no.5, pp.1219-1225, 2002.

[23] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice-Hall, Upper-Saddle River, NJ, 2001.

[24] J. M. Mendel, Type-2 fuzzy sets and systems: An overview, *IEEE Computational Intelligence Magazine*, vol.2, no.1, pp.20-29, 2007.

[25] N. N. Karnik and J. M. Mendel, Centroid of a type-2 fuzzy set, *Information Sciences*, vol.132, pp.195-220, 2001.