CONDOR: A Hybrid IDS to Offer Improved Intrusion Detection

David J. Day [#1], Denys A. Flores [*2], Harjinder Singh Lallie [%3]

[#] *Sheffield Hallam University, Furnival Building, 53 Arundel Street, Sheffield, S1 2NU, United Kingdom*
[1] d.day@shu.ac.uk

[*] *National Polytechnic School, Department of Informatics and Computer Sciences (DICC), Quito, Ecuador*
[2] denys.flores@epn.edu.ec

[%] *University of Warwick, International Digital Laboratory (WMG), Coventry, CV4 7AL, United Kingdom*
[3] h.s.lallie@warwick.ac.uk

Abstract— **Intrusion Detection Systems are an accepted and very useful option to monitor, and detect malicious activities. However, Intrusion Detection Systems have inherent limitations which lead to false positives and false negatives; we propose that combining signature and anomaly based IDSs should be examined. This paper contrasts signature and anomaly-based IDSs, and critiques some proposals about hybrid IDSs with signature and heuristic capabilities, before considering some of their contributions in order to include them as main features of a new hybrid IDS named *CONDOR* (*COmbined Network intrusion Detection ORientate),* which is designed to offer superior pattern analysis and anomaly detection by reducing false positive rates and administrator intervention.**

Keywords – Anomaly, signature, hybrid, IDS, NIDS, false positive, false negative, intrusion detection

I. Introduction

Intrusion Detection Systems (IDSs) monitor, and detect malicious activities against a computer or a set of computers. With the advance of cloud computing and evidence that traditional host based protection is not a pancea to security they are becoming a necessity [1]. However, they have some limitations which may lead to false positives and false negatives [2]. For instance, if a new service or application is installed, a heuristic-based IDS may define this behaviour as a false positive whereas if malicious code has been slightly modified, it can bypass the signature-based IDS detection as a false negative [3]. Therefore, detecting false positives/negatives is a hard task which, if unsuccessful, can compromise the entire network security.

This research explores a new mechanism to reduce false positives and negatives during attack detection. In doing so we have reviewed previous work to explore and analyze ways in which signature and anomaly-based IDSs can be combined to offer more efficient detection capabilities.

The rest of this paper is structured as follows. In Section 2, the deficiencies of both anomaly and signature-based detection methods are compared and discussed. In Section 3, four proposed models of IDSs with signature, heuristic, and hybrid capabilities are analysed along with their contributions so that a new hybrid IDS model, named as *CONDOR*, can be deployed to offer better detection against known and unknown attacks.

In Section 4, *CONDOR* is discussed to explain how it can reduce false positive/negative rates, and enhance the overall IDS performance without excessive administrator intervention.

Finally, conclusions and constraints in our proposal are offered for further research.

II. Contrast Between Signature and Anomaly Based IDSs

A. Deficiencies of Signature-Based IDSs

A signature/misuse/pattern-based IDS scans packets looking for a set of patterns that may constitute an attack in progress. In addition, this kind of IDS conducts a deep inspection of the packet sections looking for malicious patterns in the content of the header and payload. Even though this inspection method seems effective, there are two main disadvantages of these IDSs, namely inaccuracy of the signature detection against unknown attacks [4], and deficiencies in pattern analysis [5].

The first disadvantage is due to misuse-based IDSs being unable to detect unknown attacks, or variations of a previous attack pattern [6]. This is due to signature based IDSs rely on string comparison [7]; thus variants of the attack as well as unknown attacks can deviate from the comparison string, or signature, and avoid detection, leading to false negatives [8].

Secondly, misuse-based IDSs have deficiencies regarding pattern analysis, and rule writing techniques [3]. Writing rules using a root-cause analysis of the intrusion is reliable as it considers the context of the attack, and not only its characteristics [9]. However, it relies on the human ability to make a good analysis of the vulnerability, so if all the possible causes are not properly analyzed, the rule becomes ineffective. Additionally, the root-cause analysis is a time-consuming process that can take days, or even months to produce results [3]. Conversely, writing unique-pattern rules is both quick, and simple because it involves looking for data that is unique for either an exploit, or any other malicious traffic, without a deep understanding of how the vulnerability actually works [10].

Due to the outlined issues, writing rules by either analyzing their root causes, or simply crafting them for a specific malicious attack, often involves participation of security analyst communities [11]. These communities work together to look through reported attacks, e.g. Snort blog[1]; however, the nature of these attacks may be just isolated events instead of being a general security issue.

B. Deficiencies of Anomaly-Based IDSs

In case of anomaly/heuristic-based IDSs, some disadvantages are present in the behavioural model generated during the training phase [12], in the system performance [4], and in the administrator intervention [13]; the last two during the detection phase.

Initially, anomaly-based IDSs create a heuristic model to compare the current network activity with normal characteristics of traffic [3]. Then, if the current traffic deviates from the normal behaviour, the administrator is alerted to a possible attack in progress [14]. Although, this IDS is shown to be more effective at detecting unknown attacks [15], a lot of time must be invested during the training phase in order to create a database with normal operation thresholds to reduce the false alarm rate [12]. Moreover, some anomaly-based IDSs base their training time considering only one computer, which does not represent a real network operation where different computers and environments coexist [8]. The detection task is inaccurate as a result of false positives due to the lack of behavioural information generated in the comparison model.

In contrast, regarding system performance, much recent research has focused on *system call anomaly detection*, the analysis of which relies on techniques such as neural networks, and data mining [6]. Particularly, two prominent lines of knowledge in the field of anomaly based IDSs have been identified. These are mathematical [16], or neural network assisted [17]. Both of these, however, have shown limitations in performance due to high resource utilisation. For instance, the first implements statistical models and data mining to define normal behaviour [18] [19] which is highly resource intensive due to both time and processing requirements necessary for defining threshold levels and the analysis of what constitutes deviation. The second employs neural networks with the purpose of reducing false positives; however, its processing time is even longer [4] due to the recursive routines in neural networks[2]. Hence, even though both lines of knowledge are focused on reducing the false positive rate, both compromise the overall system performance as well; in particular, when neural networks are employed.

Finally, during the detection phase, some anomaly-based IDSs send alert messages asking for administrator intervention [13]. At this point, the IDS may warn of potential intrusions even when they are related to false positives. This is often due to a lack of information of normal behaviour where the IDS cannot differentiate legitimate operations from attacks [8]; e.g. if a new application is deployed, or an existing one is upgraded, the IDS may raise an alert if the event is defined as abnormal.

C. Comparison Between Signature and Anomal- Based IDSs

Given the deficiencies outlined above, we now proceed to exploring the key differences and contrasts between anomaly and signature-based IDSs.

- Pattern IDSs search for known attack signatures whilst anomaly IDSs look for deviations in normal operation behaviour to detect unknown attacks [20]. Then, if an attack is undetectable using signatures, it may be identified by anomaly-based IDSs [3]. Furthermore, since anomaly based IDSs build behavioural models for the detection of unknown attacks [12], they are not dependant on signature coding, or string matching criteria; therefore, the understandability, learnability and operability in an anomaly-based IDS may be higher than the correspondent characteristics of usability[3] in a signature based IDS because the former learns from the user behaviour, and the second requires a constant input of new signatures.
- Anomaly-based IDSs require a training stage to define normal behaviour thresholds; however, even though a signature-based IDS does not require training, it is more likely to report false negatives when an attack does not match a rule, e.g. when the attacker changes the malicious patterns of the attack to deviate from a preconfigured signature [21]. Correspondingly, when considering dynamic network activity, it is more likely to have false positives than that of anomaly-based IDSs [4] [22].
- Both IDSs are focused on detecting attacks in the network layer; however, at present, the tendency of attackers is to compromise systems at the application layer [20]. Thus, due to the unknown events that may occur at this layer via the differing nature of applications, it seems that anomaly-based detection could be more appropriate, although they may fail in the purpose of detecting specific forms of known attacks like SQL injections, where signature-based IDSs are shown to be more effective [20].

III. Combining Signature and Anomaly-Based IDSs for Superior Detection

As discussed in the previous section, since signature and anomaly-based IDSs may have some inherent deficiencies, combining them could be an important solution in order to harden the intrusion detection process [23]. This has been fairly discussed in previous models about hybrid IDSs, the features of which are analyzed in this section with a view to combining them into a hybrid IDS capable of detecting both known and unknown attacks, taking advantage of the signature accuracy, and the heuristic versatility.

In the work done by Hwang et al. [24], a hybrid IDS with heuristic capabilities, and automatic signature generation is proposed. First, Snort is connected in cascade with a novel Anomaly Detection System (ADS), in order to detect possible attacks. This proposal followed a serial production line to detect intrusions, generate rules, and update the Snort database. As first step, the standalone Snort installation detects known attacks using its raw signature database. Then, an episode-mining engine generates frequent episode rules (FERs) with different levels of support thresholds to enable unknown attack detection. It is an enhanced process to define anomalous

[1] Snort blog is available at http://blog.snort.org/

[2] This is the recursion required during the training phase cited in Yong, 2010, p.405

[3] The term and definition of usability is proposed in the ISO/IEC 9126 standard for every software-based product whose characteristics are the understandability, learnability, operability, among others.

episodes in which the FERs are evaluated with pre-computed frequent episodes from normal traffic. Hence, the FERs which thresholds levels either mismatch the frequency levels of these normal episodes, or match them with unusual high frequency are considered as anomalous. Subsequently, the anomalous FERs are used to generate signatures, and update the Snort database for further detection. Therefore, in terms of precision, this proposal seems to offer a proper intrusion detection solution with less administrator intervention; nonetheless the malicious sequences or episodes analyzed with the Snort database are based on standard pattern comparisons which omit the fact that this signature-based IDS may not detect attacks in different sections of the sequences. Moreover, this model updates the signature database simultaneously every time an anomaly is detected; as consequence, the processing time, and the whole system overhead may be high.

The work done by Zhang et al. [20] was also analyzed, in which a novel signature-based IDS for detecting database intrusion was suggested in order to enhance SQL injection detection. In this model, instead of directly using a signature-based IDS to detect intrusions, a sniffer is implemented to capture database communication packets received by the *n* database hosts in order to extract SQL commands. Later, these sequences are sent to a database signature-based detection system to match them with known attack patterns. Hence, the sniffer reduces the throughput in the database signature-based IDS by just sending the sequences with suspicious content. However, the disadvantage with this model is that it only detects database intrusions, so it cannot be used as a general signature-based IDS for monitoring an entire network with different services and behaviour.

In order to improve heuristic, and signature detection, a hybrid IDS proposed by Li et al. is analyzed [6]. First, this model captures system call sequences based on pattern matching by inspecting the names, parameters and returned values on each system call. Next, the captured sequences are sent to a sequence analysis and matching module in which they are divided in short sequences of *k* length using the *sliding window* algorithm. This step is performed to detect unknown patterns in the sequences that could not be detected by normal signature matching. Later, these sequences are used to create a heuristic model by defining short sequence correlation and matching coefficients. Finally, the sequence patterns defined as normal are used to generate a database of secure sequence patterns. As a result, the intrusion detection relies on both signature and anomaly detection systems, but unlike the prior hybrid IDS discussed, this one improves the intrusion detection by defining anomalous episodes based on mathematical models (correlation and matching models) instead of in just their frequency (frequent episode rules). Nonetheless, this model was tested on just one computer with its efficacy in a realistic network environment undetermined.

An improvement on this aspect is considered by Ohtahara et al. [12] with a distributed anomaly-based IDS named ADCOIN. It is a client/server-oriented IDS which creates normal behaviour threshold levels during its training phase by considering different environments on the network. One integrated database is created which is replicated on each of the hosts for local anomaly analysis. ADCOIN combines host and network intrusion detection into one model. Even though this approach increases the human interaction, it reduces the false alarm rate because the heuristic model is built considering similar behaviour in the entire network environment and not in just one machine.

In conclusion, although the discussed models have some disadvantages, they have shown interesting approaches that may be implemented into an improved hybrid IDS. In the following section, *CONDOR* which combines many of the positive aspects of the above systems, while mitigating many of the disadvantages, is proposed.

IV. Introducing CONDOR: an Improved Hybrid IDS

CONDOR (COmbined Network intrusion Detection ORientate) is focused on improving the performance of a hybrid IDS, reducing false positive rates, and administrator intervention as much as possible. Furthermore, *CONDOR* incorporates the main features of the models discussed in Section 3, into a new *architectural model* for *distributed anomaly detection* and *automated signature generation.* In essence, *CONDOR* balances the sequence capture using a *sniffer*, and a similar architecture suggested by Zhang et al. [20]. A *distributed anomaly-based IDS* approach is also used similar to that proposed by Ohtahara et al. [12], the *sliding window* algorithm used is the one suggested by Li et al. [6] to get the sequence patterns. Finally, the former approach suggested by Hwang et al. [24] is also considered in order to *generate new signatures* from the collected sequences, and therefore, reduce the administrator intervention. Thus, *CONDOR's three core modules are*: the *user interface*, the

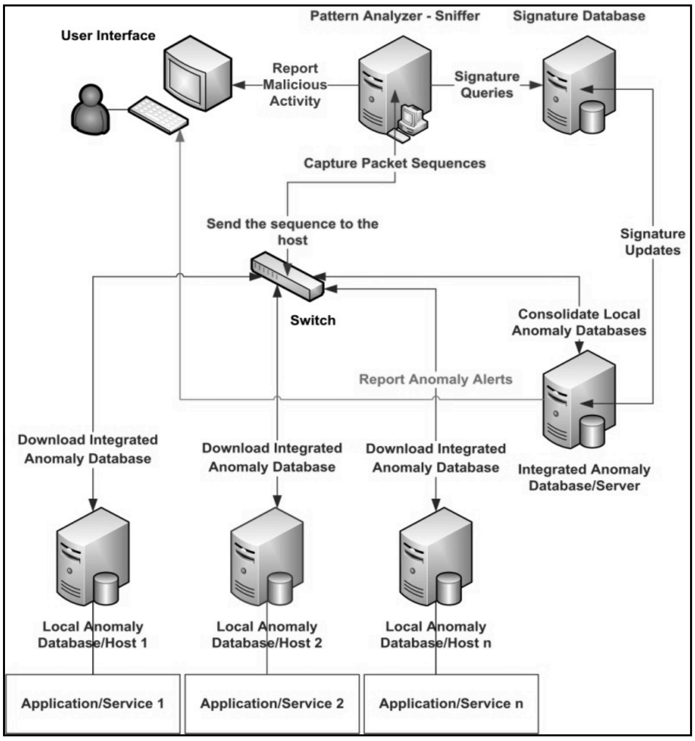
User Interface
Pattern Analyzer - Sniffer
Signature Database
Report Malicious Activity
Signature Queries
Capture Packet Sequences
Send the sequence to the host
Signature Updates
Switch
Consolidate Local Anomaly Databases
Report Anomaly Alerts
Download Integrated Anomaly Database
Download Integrated Anomaly Database
Download Integrated Anomaly Database
Integrated Anomaly Database/Server
Local Anomaly Database/Host 1
Local Anomaly Database/Host 2
Local Anomaly Database/Host n
Application/Service 1
Application/Service 2
Application/Service n

Figure 1. Proposed Structure of *CONDOR* combining a Signature Based IDS and a Distributed Anomaly Based IDS with Signature Generation and Anomaly Detection Capabilities

signature detection component (S), and the *anomaly detection component (A)*; the model structure is shown in Fig.1. The User Interface

As a main point of communication in *CONDOR*, the user interface is in charge of coordinating the communication between the administrator, the *signature detection component*, and the *anomaly detection component*. The alert messages are displayed on *CONDOR's* console where the administrator can also configure both components.

A. The Signature Detection Component

In *CONDOR,* this component is formed by a *pattern analyzer* and a *signature database*.

This *pattern analyzer* is essentially a *sniffer* which captures and processes the system call sequences that it can identify in the packets received and sent by *n* hosts, each of which provides *n* applications/services. This structure is like the one proposed by Guo-Song and Zhi-Chao [22], in which a group of detectors monitor all the traffic from and to the *n* hosts. Then, as suggested by Zhang et al. [20], this analyzer compares the sequences according to the patterns stored in the *signature database*. Subsequently, if an attack is detected, the *analyzer* sends an alert message to the *user interface*, and a *kill* message is sent to the host in order to terminate the process detected as malicious. The risk of having a false positive here is low because the process is killed only if the sequence matches a malicious signature. It is expected then, that the false positive rate in the entire network will be reduced.

If an attack is not detected, the *pattern analyzer* sends the sequence back to the host in which the anomalous sequence was originated. Afterwards, assuming the sequence as a false negative, it is analyzed by the *anomaly detection component* in order to define whether it is a normal behaviour or not.

B. The Anomaly Detection Component

CONDOR's anomaly detection component performs heuristic analysis on the sequences assumed as false negatives. As previously highlighted by Ohtahara et al. [12], it comprises of one *integrated anomaly database* stored in an *anomaly-based server*, and a set of *local anomaly databases* stored in the *n hosts* or *anomaly-based clients*. The purpose of having an integrated anomaly database is to collect anomalous behaviour from the different local anomaly-based clients so that a very accurate behavioural model can be created considering the entire network. Hence, *CONDOR's employs* both the server and the clients to form one single anomaly-based IDS, as shown in Fig.1. Three engines exist within this structure to assist the heuristic detection, they are: the *local anomaly analyzer*, the *anomaly database integration engine*, and the *signature generation engine*.

1) Local Anomaly Analyzer.

In order to perform the local anomaly analysis, each *CONDOR* client retrieves the *integrated anomaly database* from the server. The client waits for a sequence to arrive; then, each sequence is analyzed against the replicated database. The client in charge of analyzing a given sequence is the one that receives it from the *pattern analyzer*. If the sequence is not a subset of the replicated database, a message is sent to *CONDOR's user interface* asking for administrator intervention. The administrator has to decide whether the new sequence is anomalous or normal. If the sequence is defined as normal, it is added to the replicated database which now becomes different from the integrated database, given that a new sequence has been added, and thus, becoming a *local database* which is active only in the current client's environment. As the sequence was defined as normal, the process continues its normal operation. If the sequence is not normal, the client kills the process.

The problem with this form of anomaly detection, (as previously suggested by Gui-Xiang and Wei-Min [12]), is the possibility of having false positives due to the client performing a very subtle sequence analysis. In *CONDOR,* the analysis of anomalous sequences is enhanced by using the *sliding window* algorithm, as suggested by Li et al. [6]. In this algorithm, the sequence is divided in *short sequences* of *k* length. For explanation purposes, it is assumed that each one of the *short sequences* is a subset of the entire sequence, as shown in (1):

$$S = \bigcup_{i=1}^{n} Si \qquad (1)$$

Where:

- *S* is the original sequence comprising of S_i subsets
- *n* is the number of subsets in which the original *S* sequence was divided, using the *sliding window* algorithm.

It is assumed that these subsets are exclusive.

Each one of the subsets is compared against the *replicated database* in the client, and its matching percentage is the *short sequence matching result* [6].

Later, each short sequence is compared with the normal thresholds in the *replicated database*. The probability of having them on this database is the *mathematical expectation E,* the maximum of which is the *correlation coefficient,* as shown in (2).

$$\max\left(E(S_i)_{i=1}^{n}\right) \qquad (2)$$

Therefore, as stated by Li et al. [6], if the *short sequence matching result* is >= 80%, the sequence is normal. If the *short sequences matching* result is >=60% and the *correlation coefficient* is >=60%, the sequence is normal. Otherwise, the sequence is considered as abnormal in which case: the process is killed, the sequence is sent to the *CONDOR's signature generation engine*, and the alarm is sent to the *user interface* to notify the administrator. In all the cases where the sequence is normal, it is registered on the *replicated database* which becomes the *local database* because it is different from the former *integrated database*. In *CONDOR,* this method of sequence analysis reduces the false negative rate because the matching spectrum is considered using *short sequences* rather than the original sequence sent by the *pattern analyzer.* I.e. the sequence is added to the *local anomaly database* only if both the *matching result* and *correlation coefficient* defined it as normal.

2) *Anomaly Database Integration Engine.*

Once the sequences have been analyzed, the result is a local database with a set of new sequences. In order to merge these events in *CONDOR's* integrated anomaly database, the *majority algorithm* suggested by Gui-Xiang and Wei-Min [12] is used.

First, all the collected sequences on each *anomaly-based client* represent various normal events, which may be normal in the other clients as well. Then, *CONDOR's anomaly-based server* retrieves each *local database*, and defines how many clients have reported the same sequences. Next, the number of clients is assigned as a sentinel value to each set of sequences. Once all the databases have been assessed, if the sentinel value of a given sequence is much smaller than the others, the server judges it as anomalous, and the sequence is added to *CONDOR's integrated database*.

Thus, next time a client needs to perform an analysis, it will retrieve the last version of the *integrated database*, and at the same time, each client will contribute to detect anomalies, defining normal behaviour thresholds locally which will be included later as a general network behaviour model.

3) *Signature Generation Engine.*

In order to tackle the excessive administrator intervention due to the alert messages in *CONDOR's* console, the *signature database* has to be updated. The current model follows the same proposal suggested by Hwang et al. [24], modified in such way that *CONDOR's signature generation engine* is part of the *integrated anomaly server* instead of being an isolated component. This is done with the purpose of avoiding the heuristic detection and signature generation in parallel. I.e., *CONDOR's anomaly-based server* uses the *signature generation engine* only if it receives abnormal system call sequences from any of the *anomaly-based clients*. Therefore, new signatures are generated by this engine to update *CONDORS's signature database,* once the abnormal sequences have gone through the whole anomaly detection process as a result of a false negative which bypassed the signature detection. Thus, next time a similar sequence is sent to the *signature-based component,* it will be detected immediately reducing both the processing time of sequences in the *anomaly-based component*, and the administration intervention attending alert messages in the *console* due to false positives.

V. CONCLUSIONS

Signature and anomaly based IDSs are security devices that must be deployed to harden the security in a network infrastructure. Moreover, as previously discussed, combining these *IDSs* is important in order to enhance the detection of unknown attacks and pattern analysis in *misuse-based IDSs*, as well as mitigating the disadvantages in the behavioural model generation, system performance, and high administrator intervention in anomaly-based IDSs. As consequence, improved protection against attacks can be achieved, maximising the chances of intrusion detection through signature-based capabilities to detect known attacks, and the anomaly-based IDS's ability to detect suspicious behaviour using heuristics. By this means, malicious packet sequences that may have bypassed the signature control can be detected.

In our proposal of hybrid IDS, *CONDOR's* design (Fig. 1) proposes a *new architectural model* for *distributed anomaly detection* and *automated signature generation*, incorporating the main features of the models discussed in Section 3. In fact, *CONDOR* reduces the user intervention to update signatures and thresholds; the first in misuse-based IDSs, and the second in heuristic-based IDSs during training phase. Additionally, the enhancements on the *short sequence* definition in the model suggested by Ohtahara et al. [12] enables *CONDOR* to generate a more accurate behavioural model. However, as the short sequence analysis is still relying on string matching comparison, there may still be a chance to have a false negative which may have been apparent over the larger sequence.

Finally, one advantage of using the *distributed anomaly-based IDS,* proposed by Ohtahara et al. [12], inside the *CONDOR's anomaly detection component* is the enhancement in the heuristic detection rate due to the consideration of different environments. Nonetheless, the suggested structure (Fig. 1) is proportional to the number of hosts in the network because the anomalous component combines host and network anomaly-based detection, which might increase the cost of deploying *CONDOR*, even though the false positive rate may decrease due to the mathematical techniques suggested by Li et al. [6].

VI. CONSTRAINTS IN OUR PROPOSAL

CONDOR enhances the detection of system penetration via system calls, but it cannot stop or detect polymorphic worm propagation due to the mutant nature of this kind of malware [25]. Since some worms use heap overflows and dynamic allocation to gain system control [26] [27], it would be required to train the *CONDOR's* anomaly-based component under worm propagation conditions to assess whether the behavioural model could be able of detecting this kind of intrusion.

ACKNOWLEDGMENT

We wish to acknowledge the sponsorship of the National Polytechnic School as well as the National Secretariat of Higher Education, Science, Technology, and Innovation of Ecuador which have made it possible to undertake this research.

References

[1] D.J. Day and Z. Zhao, "Protecting Against Address Space Layout Randomization (ASLR) Compromises and Return-to-Libc Attacks Using Network Intrusion Detection Systems.," *International Journal of Automation and Computing*, vol. 8, no. 4, pp. 472-483, Dec. 2011.

[2] W. R. Cheswick, S. M. Bellovin, and A.D. Rubin, "Intrusion Detection," in *Firewalls and Internet Security: Repelling the Wily Hacker*, 2nd ed. Boston: Addison-Wesley, 2003, pp. 279-283.

[3] Ryan Trost, "Intrusion Detection Systems," in *Practical Intrusion Analysis: Prevention and Detection for the Twenty-First Century*, Karen Gettman, Ed. Boston, USA: Addison-Wesley, 2010, ch. 3, pp. 53-85.

[4] P. M. Mafra, V.Moll, J. da Silva Fraga, and A.O.Santin, "Octopus-IIDS: An Anomaly Based Intelligent Intrusion Detection System," in *IEEE Symposium on Computers and Communications*, Riccione, Italy, 22-25 June 2010, pp. 405-410.

[5] S. Jajodia, *Intrusion Detection Systems*, R.Di Pietro and L.V. Mancini, Eds. New York, US: Springer, 2008.

[6] W.Li, Z.Li, H.Shi, and W.Li, "A Novel Intrusion Detection System for E-Commerce System," in *International Conference on Management of e-Commerce and e-Government*, Nanchang, China, 16-19 September 2009, p. 454.

[7] Z.Trabelsi and R.Mahdy, "An Anomaly Intrusion Detection System Employing Associative String Processor," in *Ninth International Conference on Networks*, Menuires, France, 11-16 April 2010, p. 220.

[8] C.C.Lo, C.C.Huang, and J.Ku, "A Cooperative Intrusion Detection System Framework for Cloud Computing Networks," in *39th International Conference on Parallel Processing Workshops*, San Diego, USA, 13-16 September 2010, p. 281.

[9] J.Yang, X.Chen, X.Xiang, and J.Wan, "HIDS-DT: An Effective Hybrid Intrusion Detection System Based on Decision Tree," in *International Conference on Communications and Mobile Computing*, Shenzhen, China, 12-14 April 2010, p. 70.

[10] J.Mallery et al., "Intrusion Detection and Response," in *Hardening Network Security*, Jane K. Brownlow, Ed. Emeriville: McGraw-HIll, 2005, pp. 365-386.

[11] E.Flior et al., "A Knowledge-Based System Implementation of Intrusion Detection Rules," in *IEEE Seventh International Conference on Information Technology*, Las Vegas, USA, 12-14 April 2010, pp. 738-739.

[12] S.Ohtahara, T.Kamiyama, and Y.Oyama, "Anomaly-based Intrusion Detection System Sharing Normal Behavior Databases among Different Machines," in *Ninth IEEE International Conference on Computer and Information Technology*, Xiamen, China, 11-14 October 2009, pp. 217-219.

[13] D.L. Prowse, "Computer Security," in *CompTIA Security+ SY0-201 Cert Guide*. Indianapolis, USA: Pearson Certification, 2011, ch. 2, p. 35.

[14] L.Gui-Xiang and G.Wei-Min, "Research on Network Security System Based on intrusion Detection," in *International Conference on E-Business and E-Government*, Guangzhou, China, 7-9 May 2010, p. 2096.

[15] F.Haddadi, S.Khanchi, M.Shetabi, and V.Derhami, "Intrusion Detection and Attack Classification Using Feed-Forward Neural Network," in *Second International Conference on Computer and Network Technology*, Bangkok, Thailand, 23-25 April 2010, p. 262.

[16] A.Jamdagni, Z.Tan, P.Nanda, X.He, and R.Liu, "Intrusion Detection Using Geometrical Structure," in *Fourth International Conference on Frontier of Computer Science and Technology*, Shanghai, China, 17-19 December 2009, p. 328.

[17] H.Yong and Z.X.Feng, "Expert System Based Intrusion Detection System," in *Third International Conference on Information Management, Innovation Management and Industrial Engineering*, Kunming, China, 26-28 November 2010, p. 404.

[18] S.Naiping and Z.Genyuan, "A study on Intrusion Detection Based on Data Mining," in *International Conference of Information Science and Management Engineering*, Xi'an, Shaanxi, China, 7-8 August 2010, p. 135.

[19] D.Zhao, Q.Xu, and Z.Feng, "Analysis and Design for Intrusion Detection System Based on Data Mining," in *Second International Workshop on Education Technology and Computer Science*, Wuhan, Hubei, China , 6-7 March 2010, p. 339.

[20] Y.Zhang, X.Ye, F.Xie, and Y.Peng, "A Practical Database Intrusion Detection System Framework," in *IEEE Ninth International Conference on Computer and Information Technology*, Xiamen, China, 11-14 October 2009, pp. 342-347.

[21] C.P. Pfleeger and S.L.Pfleeger, "Intrusion Detection Systems," in *Security in Computing*. Boston, USA: Pearson Education, 2007, pp. 484-490.

[22] J.Guo-song and Y.Zhi-Chao, "Intrusion Detection Models analysis and study of a new structure," in *International Symposium on Intelligence Information Processing and Trusted Computing*, Wuhan, Hubei China, 29-29 October 2010, pp. 676, 677.

[23] Z.Csajbók, "Simultaneous Anomaly and Misuse Intrusion Detections Based on Partial Approximative Set Theory," in *19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing*, Ayia Napa, Cyprus, 9-11 February, 2011, pp. 651-655.

[24] K.Hwang, M.Cai, Y.Chen, and M.Qin, "Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 1, pp. 41-55, January-March 2007.

[25] E.Chien and P.Ször. (2002) Blended Attacks Exploits,Vulnerabilities and Buffer-OverflowTechniques in Computer Viruses. [Online]. http://www.symantec.com/avcenter/reference/blended.attacks.pdf [Accessed 3 August 2011]

[26] J.Pincus and B.Baker, "Beyond Stack Smashing: Recent Advances in Exploiting Buffer Overruns," *IEEE Security & Privacy*, pp. 20-27, July 2004.

[27] C.H.Yau, Y.Y.Tan, A.S. Fong, and P.L.Mok, "Embedded Architectural Design Using Protection Logics to Defend Attack of Buffer Overflow and Unauthorized Access of Code," in *IEEE 8th International Conference on Computer and Information Technology Workshops*, 8-11 July 2008, p. 265.