Adaptive Clustering Method for Reclassifying Network Intrusions

Nehinbe Ojo Joshua

School of Computer Science and Electronic Engineering System
University of Essex, Colchester, UK
jnehin@essex.ac.uk

Abstract. The problems of classification and reporting of suspicious security violations often degenerate to other complex problems. However, efforts of system administrators to mitigate these flaws by reclassifying intrusive datasets so that realistic attacks can be substantiated are frequently unfruitful with swamped datasets. Also, the urgency required to process alerts has made validations of reduction criteria to be implemented with realistic attacks and unfortunately, these consistently endangering computer resources on the networks to more exposures. Consequently, the development of computer attacks that have been warned but still succeed is a classical problem in computer security. In this paper therefore, we have implemented a new clustering method to reduce these problems. Also, evaluation that we performed with synthetic and realistic datasets clustered alerts of each dataset to achieve a cluster of white-listed alerts. Moreover, the results obtained have indicated how system administrators could achieve prompt countermeasures to prevent realistic attacks.

Keywords: intrusion quarantining, intrusion blacklisting, intrusion whitelisting, probing attacks.

1 Introduction

Internet technology and network protocols have numerous vulnerabilities that are frequently exploited by hackers [1], [8], crime syndicates and terrorists to perpetrate illegitimate activities across the globe. These have been widely corroborated by frequent cases of computer attacks such as Distributed Denial of Service (DDoS), virus, spam, buffer-overflow, e-data pilfering, deletion of audit trails, e-spoofing of sensitive information, probing attacks, e-masquerading, password cracking, etc across [1] the globe. Accordingly, this has necessitated the need to adequately secure computer networks and to constantly audit the security of the networks at a frequency of 24 hours per day. Hence, preventive technology such as Network Intrusion Prevention System (NIPS), honey pots, firewalls and routers are frequently [9], [18], [21] deployed on the networks to 18] disallow intrusive packets [21] from migrating into the networks.

Nevertheless, since authorized activities often change over time and computer attacks are becoming sophisticated everyday [15], these devices are often manipulated, electronically masqueraded and circumvented by attackers. Consequently, intrusion detectors are deployed as [8] an additional network layer of defensive mechanism to

D. Weerasinghe (Ed.): ISDF 2009, LNICST 41, pp. 135–146, 2010.
© Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering 2010

complement preventive measures. In essence, the premise behind this concept is that the tendency of achieving substantial reduction in the cases of successful attacks on the networks would be very high if attacks that cannot be prevented are promptly detected.

Fundamentally, there are serious obstacles that militate against the realization of these security goals in realistic networks. For example, network intrusions detectors exhibit classification and reporting problems [7], [8], [17], [20], [23] because they cannot accurately address the problems of intrusion quarantining, intrusion blacklisting and intrusion white-listing. The concept of intrusion quarantining is a method to rigorously isolate already reported intrusions on the networks and store them in a separate repository while intrusion blacklisting is seen as a method to deliberately avoid noisy events or attacks by putting them on a blacklist and intrusion white-listing is perceived as a process of reporting a list of approved intrusions on the networks. For these reasons, network detectors excessively report and log [8], [17] unnecessary details [23] that always constitute impediments to prompt intrusion analyses and clarity of security breaches. Consequently, the clarity of security violations in a realistic network is often lost and hence countermeasures that would prevent them are wrongly implemented.

Nevertheless, the methodology of substantiating reported alerts is another challenge that stems from the aforementioned problems. Most system administrators still spend wasteful time [9] and efforts to select appropriate filtering criteria that are used to corroborate the existence of realistic attacks in the audit trails. In essence, they manually cluster sophisticated attacks and hence the tendency that erroneous actions are taken against legitimate network activities is extremely high [21]. Therefore, analyses of attacks that have been warned beforehand are unduly delayed and they are not forestalled at the end of the day.

Nevertheless, out of the aforementioned problems, attempts that have been made to address the problems of classifications and intrusion reporting collectively exhibit [17] low success rates. For instance, intrusion blacklisting has been moderately implemented as alerts tagging in Snort whereby rules are designed to sequentially log packets that are triggered by the same rule [16]. Similarly, the problems of classification and intrusion reporting are also proposed to be controlled by disabling noisy rules and to default some rules to drop some generic events [9] so that few quantity of alerts would be logged and warned [16]. Additionally, the quantity and quality of alerts to be reported can as well be controlled by limiting attributes that should be reported at a given time and by personalizing [16] the detection rules to specific kinds of traffics such as personalization by packet size.

However, all these techniques have inevitable weaknesses. For example, the risk classification of changing the default settings of the detection rules is high because such modification would endanger ability to detect attacks such as buffer overflow that are tracked by the size of packet payloads. Also, it is impossible and computationally resource intensive for detectors such as Snort to automatically cluster alerts on the basis of many attributes other than the conventional six [16] attributes that are currently implemented in Snort. This implies that Snort can only improve on the quality of reported alerts on the basis of generation-id, signature-id, one of its three types of threshold at a time, source-ip or destination-ip (and not source-ip and destination-ip), count and time in seconds. Although, these strategies are efficient in reducing some proportions of false positives in the audit logs however, they tend to be vendor's

specific or particularly restricted to Snort intrusion detector. In addition, Snort cannot cluster alerts [16] per port numbers or by combination of long sequence of selection criteria. Therefore, since different attacks tend to require different selection criteria to thorough ascertain their patterns, the ideas of limiting alerts or adopting restricted logging of security violations to interesting attributes are sources of potential dangers to the resources on the networks.

For these reasons, researchers [1], [3], [17], [23] over the years have automated some methods that indirectly reduced the problems of intrusion reporting. Consequentially, statistical methods have been proposed to reduce false positives [2], [22], automated strategies of attacks were proposed to preempt the intentions of attackers [14] and classification rules [4], [7], [11], [12] have been suggested to reclassify alerts into several clusters. However, each of these strategies has numerous [21] weaknesses as well. For example, they are frequently and widely criticized for poor detection capabilities [1], [8], [20], lack of adaptability [21] and poor performance tuning [18] to solve realistic network problems. Essentially, both the industrial and research designs are collectively flawed [15] and they have failed to enhance significant reduction of heightening cases of computer attacks across the globe. Therefore, the aim of this paper is to address these problems. We deployed Snort intrusion detector to sniff six different datasets that we have extracted from experimental, realistic and synthetic networks. After that, an automated clustering method was applied to succinctly cluster alerts of each dataset to expedite countermeasures that would prevent attacks that are signified by the alerts.

One of the essential contributions of this paper was an automated modeling of a practical approach that has bridged the wider gap that existed between research designs and industrial problems. We have been able to extensively validate the efficacies of our model in reducing the aforementioned problems with several kinds of datasets that broadly represented basic attacks that are encountered in everyday activities. Additionally, the results obtained have fully established the fact that the efficacy of our model is independent of the size of the dataset. Furthermore, our work has pointed out focal direction that stimulates future discussions in redesigning intrusion detection researches to be directly adaptable to solve realistic problems.

The remainder of this paper is organized as follows. Section 2 gives a review of closely related works. Section 3 gives an overview of experimental datasets that we have used to verify the performance of our model and section 4 gives an account of our model for reclassifying network intrusions. Section 5 gives an account of experimental results that we have conducted while section 6 gives conclusion and future research direction.

2 Related Works

The approaches for addressing the problems of classification and intrusion reporting [1], [3], [17], [19] and some of the associated problems that we have described above are diversified. For instance, consequence or collections of ordered alerts that are linked together by their respective duplicates have been implemented to reduce false positives. Examples of duplicates are alerts that originate from the same source IP address or source ports that migrate towards the same target IP address or target ports

and within the same timestamp. In [7] for instance, duplicate alerts are merged with their consequence using algorithm that uses implicit rules. The algorithm then creates a new attack thread each time a new attack is discovered. However, this method will wrongly match attacks such as ping flooding and ping of death that exhibit closely related consequence and duplicates together.

Furthermore, AprioriAll-like sequential pattern mining rules [11] that use behavioural classification model to build a decision support system has been implemented to reclassify online alerts into false, unknown and normal alerts. Nevertheless, one of the major flaws of every behavioural classification model is that it is practically unfeasible to model all attacks. Also, Apriori-based models always exhibit low performance in processing long sequences of candidate generation alerts. In addition, apart from lack of objective evaluations that were not done with standardized evaluative datasets, this model is completely resource intensive to implement in corporate networks.

In addition, a formalized model that is proposed in [13] uses adequate knowledge of the resources on the network to eliminate false positives. Basically, the idea of this model is to isolate false alerts by reclassifying them as alerts that are not commonly reported by all the detectors on the networks whenever the alerts of all the detectors are correlated. Essentially, this model has rigorously established sound theoretical concepts that can be adopted to solve some intrusion detection's problems but then, it has numerous fundamental weaknesses. One of these is that it undermines the possibility of having some attacks that deliberately elude detections. Additionally, formalized approach suffers from lack of objective evaluation and hence, its efficacy in reducing redundant false alerts cannot be validated.

In addition, some attributes of alerts have been incorporated into expert [4], [12] systems to generate elementary (low level) alerts. In further reclassifying online alerts, this model validates each of the incoming alerts with its expert rules to determine their similarities. Thereafter, clustering is applied to cluster alerts that are generated by the same attack scenario while the proportions of the false positives in the datasets are reduced by selecting an alert per cluster. The uniqueness of different designs that adopt this approach is the mode of updating the expert rules. For example, expert system that is implemented in [12] automatically creates attack scenarios to update its expert rules unlike in [4] whereby the expert rules are manually updated by the system administrators each time a novel attack is discovered.

Support Vector Machine (SVM) method and multi-layer perceptron (MLP) are two examples of neural network [25] classifiers that are implemented to reduce false positives. The model automatically constructs graphs of attack scenarios from temporal relationships of alerts while correlation matrix procedure is introduced to eventually eliminate false alerts from alerts that temporally related. However, this method cannot process alerts of attacks that occur in multiple and unrelated phases.

Implementation of Queue graph (QG) [20] to extract patterns of multistep attacks that occur within different timestamps has also been designed to reduce false positives. In this approach, QG algorithm is used to search for the most recent alerts of each type that prepare for future alerts and they are then merged together on the basis of their timestamps. Like the attack-graph model, this method visually represents the distributions of alerts graphically. However, the interpretations of the output graphs are too complex to understand despite the timeliness that is necessary to counter intrusions.

The reduction of false positives is also achieved [24] with an exponentially weighted Dempster-Shafer theory that uses an alert confidence fusion approach. In this method, colored Petri-Net is used as an add-on to compute the probability of occurrence of each token and a specified threshold is applied to filter false alerts. Like all statistically based approaches, this model always produces low data reduction rate on datasets that contain lots of unrelated attacks. Essentially, apart from the inherent flaws of each of the existing models, they have indicated potential research gaps.

3 Overview of Datasets

We evaluated our method with simulated, realistic and synthetic attacks that were mainly DARPA 2000[6], DEFCON-8 and DEFCON-10 [5] standard datasets. Essentially, our assumption was that a wider coverage of different datasets [21] would give a good picture of realistic network attacks. The DARPA-1 (LLDoS.1.0) and DARPA-2 (LLDoS.2.0.2) were two traces of DARPA 2000 datasets that represented Distributed Denial of Service (DDoS) that were launched by novice and experienced attackers within 5 and 8 seconds respective intervals. Also, DEFCON-8 lasted for about 34 hours while DEFCON-10 lasted for about 47 hours. In addition, attacks in DEFCON-8 dataset were ports scan and buffer overflow while that of DEFCON-10 dataset were some probing and non-probing attacks that included bad packet, ports scan, port sweeps, etc. The UNI-DATA was a trace file that was obtained from one of the perimeters of a University's networks within 21 hours real-time monitoring of realistic events. Besides, a trace file from our experimental network that contained some probing attacks that were mainly ping, Xmas tree, UDP, SYN stealth scan attacks and attacks that explored versions and list of operating systems in the target machines was also extracted and labeled as TEST-DATA. Also, these attacks lasted for about 27 hours. Throughout this paper, we would constantly refer to probing attacks as attacks that seek for vulnerabilities in the target systems and non-probing attacks as attacks that exploit vulnerabilities in the target systems.

Specifically, the attacks in TEST-DATA occurred between 2009-04-28 at 12:45:45 pm and ended on 2009-04-29 at 15:21:38 pm while that of the UNI-DATA occurred between 2009-06-21 at 14:41:55 pm and stopped on 2009-06-22 at 11:38:05 am. Also, DARPA-1 occurred between 2000-03-07 at 16:27:51 pm and ended 2000-03-07 at 16:27:56 pm while DARPA-2 occurred between 2000-04-16 at 21:06:15 pm and stopped on 2000-04-16 at 21:06:23 pm. In addition, DEFCON-8 started on 2000-07-28 at 16:39:03 pm and lasted for 2000-07-30 at 02:11:15 am while DEFCON-10 that started on 2002-08-03 at 00:57:17 am stopped on 2002-08-04 at 20:56:00 pm.

4 Reclassification of Intrusions

Network intrusion detectors are not specifically designed to solve the problems of intrusion quarantining, intrusion blacklisting and intrusion white-listing. Instead, they moderately provide supportive information that can be adapted to lessen the problems of intrusion quarantining and intrusion white-listing. Essentially, we have identified three classes of suspicious network packets that commonly migrate across realistic

networks (fig 1 below). These are packets that completely elude detections (i), packets that are blocked (ii) by the defensive devices (D) on the networks and packets that are reported by the detector (iii) that is positioned on the networks. Although, categories ((i) and (ii)) are beyond the scope of this paper, however they raise lots of questions [21]. For instance, it is imperative to understand the kinds of packets that have been blocked by the network defensive system to confirm whether they are legitimate or intrusive packets. Also, packets that elude detections are potential dangers to the network.

Hence, it is imperative to know what they are [21] and the impacts that they have on the networks. Nevertheless, the third category of suspicious packets (type-iii) is the main focus of this paper and hence, the security of a network is usually compromised by the existence of type-i and type-iii on every computer network. Basically, there are two classifiers in a compromised network that usually classify type-iii packets and they are mainly the higher level and the lower level classifiers. The higher level classifier (A) is a network detector such as Snort that classifies network packets at the packet level into normal and abnormal traffics during intrusion detections.

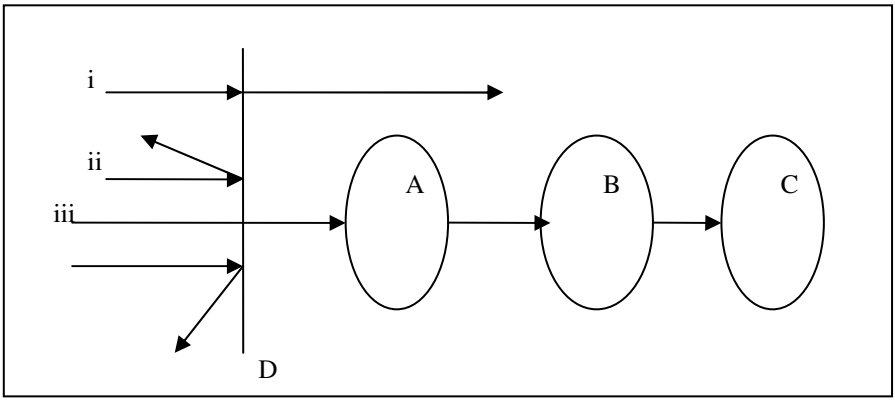

Fig. 1. Taxonomy of suspicious network packets

We have noticed that Snort is frequently regarded as the de facto standard for intrusion detection technology. The tool is an open source-code intrusion detector that uses rule-matching to perform protocol and packets' analyses, [16] logging and alerting suspicious events and it is a widely preferred choice for conducting most Network intrusion detection researches.

Accordingly, Snort validates each of the incoming packets that migrate across the networks with a database of its expert rules. Usually, a packet that does not match any of its rules is discarded while a match is appropriately tagged with reference identities or attributes [10] that have been extracted from the IP packet header. Thereafter, the packet is logged into two audit repositories. Some of these reference identities [16] are source-ip and source ports of the attacks, destination-ip and destination ports of the targets of the attacks and other attributes such as TOS, IPflag, TTL, IPprotocol, Timestamp, Priority, description, etc.

Simultaneously, an alert is sent to notify the system administrator about the presence of such event on the networks to complete the intrusion detection cycle at the higher level. Fundamentally, this is the origin of lots of issues that are facing intrusion detection technology since inception. However, we refer interesting reader to [10] and [16] for extensive discussions about network packets, alerts and their respective attributes.

In addition, the lower level classification is basically implemented by the system administrators (B). Ordinarily at this level, all notified warnings (output from higher level classifier) are reclassified and clustered into true and false positives (C).

S/N	Sequence
1	<timestamp>
2	<source-ip, destination-ip, timestamp>
3	<tos, ipflag, ttl, ipprotocol>
4	<source-ip, tos ,ipflag ,ttl, ipprotocol>
5	<destination-ip, tos, ipflag, ttl, ipprotocol>
6	<source-ip, destination-ip, timestamp, tos, ipflag ,ttl, ipprotocol>

Fig. 2. Selection criteria

Unfortunately, there are numerous challenges that disrupt these processes in reality and hence, to investigate these problems, we have used six sequences of alerts' attributes that are shown in fig 2 above as selection or filtering criteria.

4.1 Automated Reclassification of Alerts

We automated a clustering program that was implemented in C++ language to cluster alerts of each dataset on the basis of each of the six selection criteria in fig 2 above and the arrival time of the attacks in each dataset. Since each dataset has different arrival time, we standardized the clustering intervals in an increasing order of 5mins, 10mins, 20mins, 40mins, 80mins, 160mins, 320mins, 640mins, 1,280mins, 2,560mins, etc. Hence, the time to cluster each dataset in each experiment was computed as the summation of each interval and the starting time of the attacks in the dataset. For instance, in DEFCON-8, the attacks started on 2000-07-28 at 16:39:03 pm and lasted for 2000-07-30 at 02:11:15 am.

So, the first arrival time was 16:44:03 (i.e. 16:39:03 + 5mins later), the second arrival time was 16:49:03 (i.e. 16:39:03 + 10mins later), the third was 16:59:03 (i.e. 16:39:03 + 20mins later), etc until the last arrived time which was 02:11:15 on 2000-07-30. Furthermore, the automation processes were divided into two basic phases. In the first phase, for each dataset at each arrival time and a selection criterion, our model quarantined each alerts within the arrival time and clustered them into clusters of repeated alerts and white-listed alerts. Subsequently, an advanced clustering procedure was then applied to further reclassify both intrusive categories into a meaningfully condensed dataset to achieve succinct white-listed alerts. The above procedures were repeated for each of the six selection criteria and for every evaluative dataset.

Fig. 3. Analysis process of defco8alerts

Furthermore, fig 3 above is an example of a stage of our model that quarantined DEFCON-8 dataset during the process of clustering on the basis of <timestamp>. However, it is not feasible to individually report other processes for each of the dataset due to space limitations. In addition, all the results that we have obtained in the entire experiments before and after the automation were recorded and they are presented in section 5 below.

5 Experimental Results

The original quantity of alerts per evaluative dataset is as follows. TEST-DATA generated 67,469 alerts, UNI-DATA generated 16,087, DARPA-1 generated 834, DARPA-2 generated 816, DEFCON-8 generated 917,221 while DEFCON-10 generated 5,372 alerts.

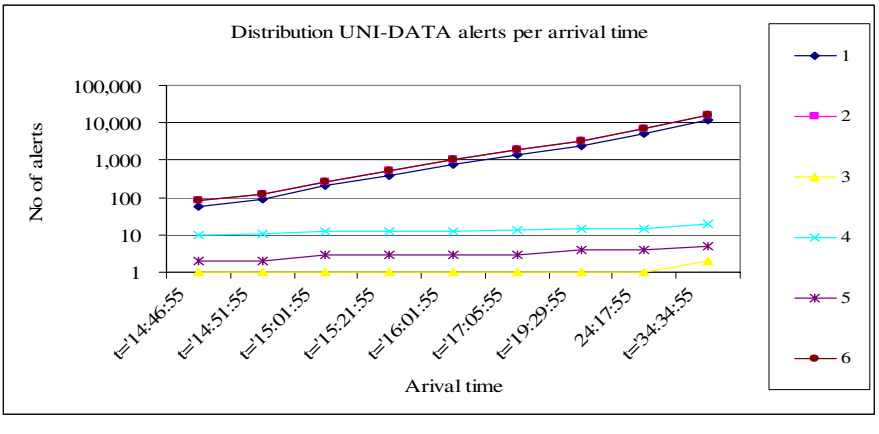

Fig. 4. Evaluation of UNI-DATA datasets

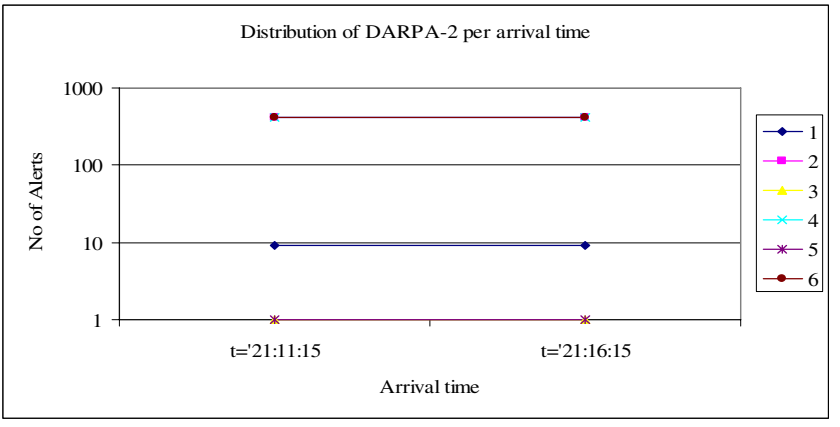

Fig. 5. Evaluation of DARPA-1 datasets

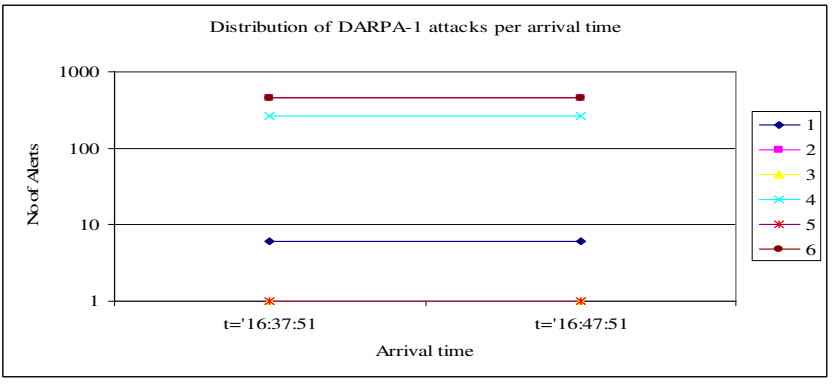

Fig. 6. Evaluation of DARPA-2 datasets

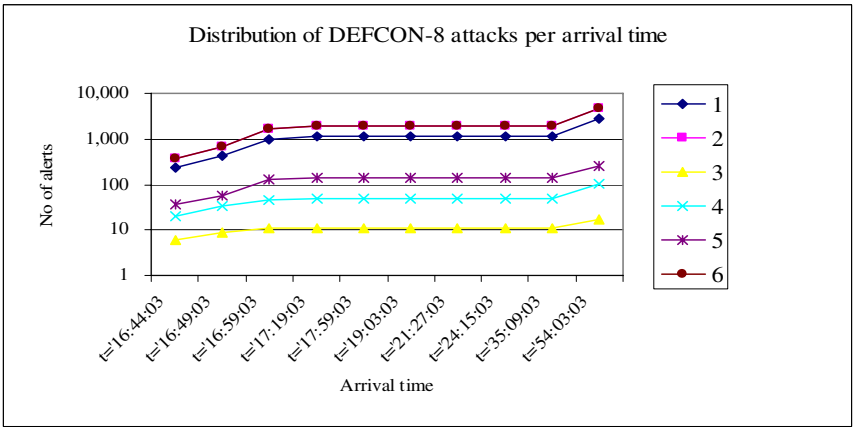

Fig. 7. Evaluation of DEFCON-8 datasets

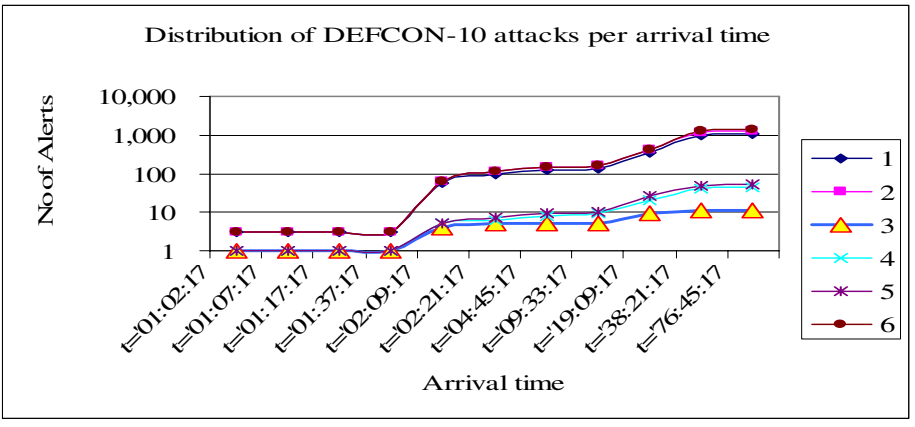

Fig. 8. Evaluation of DEFCON-10 datasets

5.1 Summary of Experimental Results

Figs 4 - 8 above were the results of various experiments per dataset. The results substantiated the significant impacts of each of the six selection criteria on realistic and synthetic datasets. The results also indicated that in investigating DDoS attacks for example, the efficacies of selection criteria 1, 3 and 5 have demonstrated high reduction capabilities to process swamped datasets that are characterized of DDoS attacks. On the other hand, selection criteria 3, 4 and 5 were only valid for investigating specific cases of probing and non-probing attacks such as buffer overflows and unauthorized network scanning especially if system administrators intend to achieve massive data reduction. Also, the results have substantiated the evidence that criteria 3 and 5 considerably reduced all the datasets by substantial proportions. For instance, criteria 3 and 5 respectively reduced UNI-DATA in the range of 1-2 and 2-5 white-listed alerts while TEST-DATA was respectively reduced in the range of 9-10 and 11-15 white-listed alerts. Also, DARPA-1 and DARPA-2 datasets were respectively reduced by criteria 3 and 5 to a white-listed alert in both cases. Similarly, in DEFCON-8, criteria 3 and 5 respectively reduced 917,221 alerts in the range of 6-17 and 36-25 white-listed alerts while DEFCON-10 was reduced in the range of 1-11 and 1-54 white-listed alerts respectively.

6 Conclusion and Future Research

This paper has extensively investigated four realistic problems that are commonly faced by system administrators in administering intrusion detectors. However, there have been several arguments that most research designs were not directly useful to solve industrial cases of computer attacks. Hence, this paper has bridged this wider gap by modelling system administrators in a simplified and automated approach.

Specifically, this paper has objectively addressed the problems of intrusion reporting, methodology of intrusion review, inability and inappropriateness of filtering criteria by automatic clustering of repeated alerts to synthesize a cluster of white-listed

alerts that enhanced prompt countermeasures. Hence, our central focus was to also extend intrusion detection researches to add value and to be operationally relevant in achieving immediate solutions to some fundamental problems in real-world.

However, we have not been able to extend this model to address some other critical intrusion detection problems such as intrusion that elude detections and intrusion blacklisting that we have briefly discussed in this paper. Hence, our opinions about them are to detect them and subsequently isolate them from migrating within the networks. One of the methods of achieving these goals is to extend our ideas about intrusion quarantining, intrusion white-listing and intrusion blacklisting into cooperative expert modules. The implementation would incorporate functionality that has the ability to expel subsequent white-listed attacks that do not provide additional information to the system administrators from the networks. Hence, these are potential research areas that we plan to extensively investigate in our future research.

References

1. Aleksandar, L., Vipin, K., Jaidep, S.: Intrusion detection: A survey, Computer Science Department, University of Minnesota (2005)
2. Alfonso, V., Keith, S.: Probabilistic alert correlation. In: Lee, W., Mé, L., Wespi, A. (eds.) RAID 2001. LNCS, vol. 2212, pp. 54–68. Springer, Heidelberg (2001)
3. Chyssler, T., Burschka, S., Semling, M., Lingvall, T., Burbeck, K.: Alarm Reduction and Correlation in Intrusion Detection Systems, Department of Computer Science, Linkoping University, Sweden (2004)
4. Cuppens, F., Miege, A.: Alert correlation in cooperative intrusion detection framework. In: Proceedings of IEEE symposium on security and privacy (2002)
5. Capture The Flag Contest-Defcon datasets (2009), `http://cctf.shmoo.com/data/`
6. DARPA.: Intrusion Detection Scenario Specific Data Sets (2009), `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/2000data.html`
7. Debar, H., Wespi, A.: Aggregation and correlation of intrusion detection alerts. In: Proceedings of international symposium on recent advances in intrusion detection, Davis, CA, pp. 85–103 (2001)
8. Fatima, L.S., Mezrioui, A.: Improving the quality of alerts with correlation in intrusion detection. International Journal of Computer Science and Network Security 7(12) (2007)
9. Hartsein, B.: Intrusion Detection Likelihood: A Risk-Based Approach SANS Institute (2008)
10. Internet Protocol: Internetworking Technology overview (1999) (2009), `http://www.cisco.com/en/US/docs/internetworking/technology/handbook/Internet-Protocols.pdf`
11. Jan, N.Y., Lin, S.C., Tseng, S.S., Lin, N.P.: A decision support system for constructing an alert classification model. Journals of Expert Systems with Applications (February 2009)
12. Kabiri, P., Ghorbani, A.A.: A Rule-Based Temporal Alert Correlation System. International Journal of Network Security 5(1), 66–72 (2007)
13. Morin, B., Me, L., Debar, H., Ducass, M.: M2D2: A formal data model for IDS alerts correlation. In: Wespi, A., Vigna, G., Deri, L. (eds.) RAID 2002. LNCS, vol. 2516, pp. 115–137. Springer, Heidelberg (2002)

14. Ning, P., Cui, Y., Reeves, D.S.: Constructing Attack Scenarios through correlation of alerts, department of computer science, NC state University, USA (2002)
15. Paxson, V.: Considerations and Pitfalls for Conducting Intrusion Detection Research, International Computer Science Institute and Lawrence Berkeley National Laboratory Berkeley, California USA (2007)
16. Roesch, M.: Snort Manual version 2.8.4 (2009), http://www.snort.org/assets/82/snort_manual.pdf
17. Sadoddin, R., Ghorbani, A.: Network Security Laboratory, University of New Brunswick, Fredericton, Canada (2006)
18. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS), Recommendations of the National Institute of Standards and Technology, Special Publication 800-94, Technology Administration, Department of Commerce, USA (2007)
19. Urko, Z., Roberto, U.: Intrusion Detection Alarm Correlation: A Survey, Computer Science Department, Mondragon University, Gipuzkoa Spain (2004)
20. Wang, L., Liu, A., Jajodia, S.: Using attack graph for correlating, hypothesizing, and predicting intrusion alerts. Science Direct, pp. 2917–2933. Elsevier, Amsterdam (2006)
21. Valeur, F., Vigna, G., Kruegel, C., Kemmerer, R.A.: A Comprehensive approach to Intrusion Detection Alert Correlation. IEEE Transactions on Dependable and Secure Computing 1(3) (2004)
22. Xinzhou, Q., Wenke, L.: Discovering Novel Attack Strategies from INFOSEC Alerts, College of Computing Georgia Institute of Technology, Atlanta, GA 30332, USA (2004)
23. Yusof, R., Sulamat, S.R., Sahib, S.: Intrusion Alert Correlation Technique Analysis for Heterogeneous Log. International Journal of Computer Science and Network Security 8(9) (September 2008)
24. Yu, D., Deborah, F.: Alert Confidence Fusion in Intrusion Detection Systems with Extended Dempster-Shafer Theory, Department of Computer Science, University of Idaho (2005)
25. Zhu, B., Ali, A.G.: Alert Correlation for Extracting Attack Strategies, Faculty of Computer Science, University of New Brunswick, Fredericton, New Brunswick, Canada (2006)