International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Online Intrusion Alert Aggregation with GDSM

Borhade Sushama R[1], Chandre P.R[2]

[1]Student, M.E. (Computer Engineering), Department of Computer Engineering,
Sharadchandra Pawar College of Engineering, Otur, Pune-412409, Maharashtra, India

[2]Professor, Guide & PG Coordinator, Department of Computer Engineering
Sharadchandra Pawar College of Engineering, Otur, Pune-412409, Maharashtra, India

Abstract: ***Online intrusion alert aggregation with generative data stream modeling uses generative modeling approach. It also uses probabilistic methods as one of the type of method. We assume that instances of an attack are similar as a alert producing process. This process may be a random process. This paper summarizes the process of collecting and modeling these attacks on some similar parameters such as source, destination etc., so that attack from beginning to completion can be identified. This collected and modeled alert is processed through different types of layers through generative data stream modeling. With some data sets, we will show that it is easy to decrease the number of alerts and count of missing meta alerts is also extremely low. Also we show that generation of meta alerts having delay of only few seconds even though first alert is produced already. Also we send these Meta alerts on registered mobile so that admin will get messages as soon as possible.***

Keywords: online intrusion detection system, data stream, alert aggregation, IDS, offline alert aggregation, online alert aggregation etc.

1. Introduction

Network security is one of the most important nonfunctional requirements in a system. Over the years, many software solutions have been developed to enhance network security and many papers provide an insight into one such solution which has become prominent in the last decade: Intrusion Detection System (IDS). Many papers have provided an overview of different types of intrusion Detection Systems, the advantages and disadvantages of the same. The need for IDS in a system environment and the generic blocks in IDS is also mentioned. The details of examples of intrusion detection systems proposed by other authors have been elaborated. The examples are as follows:

1) Misuse intrusion detection system that uses state transition analysis approach,
2) Anomaly based system that uses payload modeling and
3) Hybrid model that combines the best practices of Misuse and Anomaly based intrusion systems.

An intrusion detection system (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a management station. IDS come in a variety of “flavors” and approach the goal of detecting suspicious traffic in different ways. There are network based (NIDS) and host based (HIDS) intrusion detection systems. Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, and reporting attempts. In addition, organizations use IDPSes for other purposes, such as identifying problems with security policies, documenting existing threats and deterring individuals from violating security policies. IDPSes have become a necessary addition to the security infrastructure of nearly every organization.

IDPSes typically record information related to observed events notify security administrators of important observed events and produce reports. Many IDPSes can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g. reconfiguring a firewall) or changing the attack's content.
There are two types of IDS.

1.1 NIDS

Network intrusion detection systems NIDS are placed at a strategic point or points within the network to monitor traffic to and from all devices on the network. It performs an analysis for a passing traffic on the entire subnet, works in a promiscuous mode, and matches the traffic that is passed on the subnets to the library of known attacks. Once the attack is identified, or abnormal behavior is sensed, the alert can be sent to the administrator. Example of the NIDS would be installing it on the subnet where firewalls are located in order to see if someone is trying to break into the firewall. Ideally one would scan all inbound and outbound traffic, however doing so might create a bottleneck that would impair the overall speed of the network.

1.2 HIDS

Host intrusion detection systems run on individual hosts or devices on the network. A HIDS monitors the inbound and outbound packets from the device only and will alert the user or administrator if suspicious activity is detected. It takes a snapshot of existing system files and matches it to the previous snapshot. If the critical system files were modified or deleted, the alert is sent to the administrator to investigate. An example of HIDS usage can be seen on mission critical machines, which are not expected to change their configurations.

2. Related Work

The Intrusion Detection System which are already exists have very high accuracy in detecting the intrusions. But they

still have some drawbacks such as alerts are getting produced at very low level due to that number of alerts are very large. Due to low level production they are neither get not processed nor gets minimized. Due to large number of alerts produced users may get confused for their relative actions to block the intruders. Many authors have proposed their work to overcome these drawbacks. The related works of the authors are as follows.

The proper way to tackle the many intrusion alerts are explained in [1] by Kothawale Ganesh S., Borhade Sushama R., B. Raviprasad,. In this paper authors suggested to use of Generative Data Stream modeling to avoid repetition processing of alerts. the best suitable way for applying the relation between different types alerts is done in[6]. In this paper alert thread is reconstructed again. The alerts which are produced by IDS can be aggregated by using some fixed length window. But it can produce duplicates, which should be eliminated for proper working of IDS. So elimination of these above types of duplicates alerts is done in[7] by using the clustering process. the clustering of alerts are done online as well as offline. Firstly offline algorithm has been developed to eliminate the duplicate alerts and then extended to generate online algorithm. This online algorithm is totally made up from offline algorithm by using some of its property. The situation of current attack is done in[8]. The clustering process is used to group same attacks depends on some parameters is done in [9]. [10] suggests the another way to correlate between alerts instead of clustering of alerts. In this paper the process of combining two alerts is done on the basis of weighted and attribute wise similar operator. But from [11] and [12] this way has one of the biggest disadvantages that large numbers of parameters are needed to be set to correlate the types of alerts. [13] has similar disadvantage as[10]. To overcome this disadvantage [14] uses another clustering algorithm that uses user defied parameters such as source IP address and destination IP address. It uses strict sorting based on source and target i.e. destination IP addresses and ports in alerts. [22] uses fully different and unique way for clustering, AA-NN i.e. auto associator neural network's error is reconstructed and it helps to analyze different alerts. Alerts which are producing the same reconstruction error are grouped or placed into same type of cluster resulting into one single type of alert. The major advantage of this approach is it can be applied to offline as well as online. Offline training is required to do first of all and that can be extended to online training of AA-NN as said above.

3. Online Intrusion Alert Aggregation Technique

In this section, we will put forward our new approach for alert aggregation. As we have already stated that it uses a method called as probabilistic model of current situation of different types of attacks. First of all we describe the architecture of our system. The architecture is consists of the block diagram, which shows the detailed view and description about the layers in detail. Then we will describe about the process of generating the alerts and the alert formats i.e. what will be the contents of alerts. After that we discuss about the offline as well as online clustering algorithm. First we demonstrate offline algorithm for aggregation and how to extend it to apply it online aggregation algorithm. At last we prepare result. Analyze it to produce remark for generation of Meta alerts. Whatever Meta alerts has been produced we will send it to users registered mobile so that users can get intruder notification on their mobiles also.

3.1 Architecture

The figure 1 shows the architecture of above proposed system.

3.1.1 Sensor Layer

It is the first layer of the architecture. It is low level layer which acts as an interface between the network and host which is actually agent reside. It captures raw data from above both, filters it and takes out essential data i.e. parameters required to aggregate alerts to create an event. Sensor layer consists of sensors which captures traffic on network efficiently.

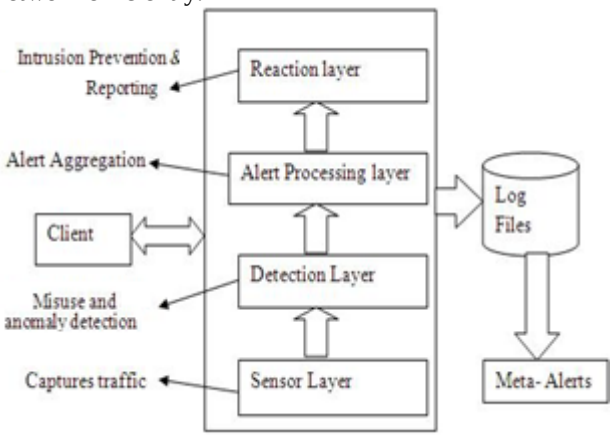

Figure 1: Detailed view of Layered Model of Proposed System

3.1.2 Detection Layer

This is second most layer which consists of different types of detectors e.g. Support Vector Machines, Snort. It takes input from sensor layer i.e. raw data collected from sensor layer. It looks for misuse detection and anomaly detection. If it finds suspicious behaviors, it generate alerts and forward to next layer of our proposed architecture i.e. alert processing layer for processing of alerts.

3.1.3 Alert Processing Layer

Whatever alerts has been received from detection layers that alerts are processed at this layer in such a way that meta alerts is generated by using GDSM i.e. Generative Data Stream Modeling. This generation of meta alerts is done on the basis of attack instance information which includes source IP address and destination IP address and possible type of attack.

3.1.4 Reaction Layer

It is something like Intrusion Prevention System which prevents intrusions. Relative and appropriate action is taken for Meta alerts produced by alert processing layer. This layer suggests the action to take for relative alerts.

3.2 Alert Generation and Format

We have discussed above the detailed functions of each layer in proposed system. Now we will discuss process of generation of alerts in detail. Sensors in sensor layer captures traffic over the internet as a raw data. It also decides attributes as an input for detector in detection layer. This attribute can be used for differentiation of attack instances in alert processing later. They may be dependent or independent. Attributes generated by the detectors are source IP address, target IP address, and possible type of attack which includes denial of service, buffer overflow and SQL injection etc. The format for alert (A) is as follows:

It has N number of attributes. Out of these N attributes let us suppose that N_m are categorical and remaining i.e. N_{m+1} are continuous.

$A = (A_1,\ldots\ldots, A_{Nm}, A_{Nm+1},\ldots\ldots.., A_N)$
Where, $A_1,\ldots\ldots, A_{Nm}$ are categorical attributes and $A_{Nm+1},\ldots\ldots.., A_N$ are continuous attributes.

3.3 Offline Alert Aggregation

In this section we briefly explain offline alert aggregation which will be extended to generative data streaming for online alert aggregation algorithm. We can show that different attacks are done on TCP/UDP traffic. Some alerts are false positive and some alerts are false negative. All information then get analyzed and finally offline alert can be generated. Though they have some drawbacks as follows.

1. Splitting of cluster may be wrongly done
2. Many different clusters may get combined wrongly into one single cluster.
3. Some of false alerts are not identified and they may get assigned to cluster
4. Wrong assignment of true alerts to cluster may happened.

Algorithm: Offline alert aggregation algorithm
Input: set of alerts (A), number of components C
Output : $\mu_c, \sigma_c^2, \rho_c$ parameters
Assignment of alerts to components
1. $\Pi c = 1/C$
2. Initiate σ_c^2 , ρ_c
3. While stopping not done do
   // E step : assign alerts to components
4. For all alerts $A^{(p)} \varepsilon$ A do
   C* := argmax H(a(p) | μc, σc2 , ρc)
5. c ε {1,……,C}
6. Assign alert $a^{(p)}$ to C*
   // M step : updating of model parameters.
7. For all component c ε {1,……,C} do
8. N_c := No. of alerts assigned to C
9. For all attributes n ε {1,……,N_m} do
10. $\rho_{cn} := 1 / N_c \Sigma a_n^{(p)}$
11. for all attribute n ε { $N_{m+1},\ldots\ldots.., N$ } do
12. $\mu_{cn} := 1 / N_c \Sigma a_n^{(p)}$
13. $\sigma_{cn}^2 := 1 / N_c \Sigma (a_n^{(p)} - \mu_{cn})^2$

From the above algorithm we can conclude that this algorithm performs following steps.

1. initialization of model parameters
2. assignment of alerts to components
3. updating of model parameters stopping process
4. coefficient mixing
5. Next it adds alerts to components slowly.

3.4 Online Alert Aggregation:

The offline aggregation algorithm is extended to generate online alert aggregation algorithm. For this IDS should have component adaption, component creation and component detection. In component adaption attack instances must be identified and should get assigned to proper cluster. In component creation new attack should created and parameters should set. In component detection attack instances should be detected.

Algorithm: online alert aggregation algorithm
Input : buffer B, Partition P, cluster number j
Output : $\mu_j, \sigma_j^2, \rho_j$ parameters
Assignment of alerts to components.
1. B : = Φ
2. While new alert do
3. If P : = Φ then
4. P_1 : = { a}
5. P : = { P_1 }
6. Initiate parameters like μ, σ^2, ρ
7. Else
8. P' : = P
9. J* : = argmax $H(\mu_j, \sigma_j^2, \rho_j)$
10. $\rho_j^* := \rho_j U \{ a \}$
11. $O_j := | C_j^* |$
12. For all attributes n ε {1,……,N_m} do
13. $\rho_{jn} := 1 / O_{j(n)} \Sigma a_n^{(p)}$
14. For all attributes n ε { $N_{m+1},\ldots\ldots.., N$ } do
15. $\mu_{j\,n} := 1 / O_{j(n)} \Sigma a_n^{(p)}$
16. $\sigma_{jn}^2 := 1 / O_{j(n)} \Sigma (a_n^{(p)} - \mu_{j\,n})^2$
17. if Ω(p) < Θ
18. P : = P'
19. B : = B U {a}
20. If novelty (a) then
    P : ALG3 (C, J*, B)
    B : = Φ
    For j ε { 1, ... | C|} do
    If obsoleteness (Pj) then
    P : = P/ Pj

4. Implementation and Results

We have implemented custom simulator by using java programming language. System requirement to do the implementation is JDK 1.6, Eclipse or Net beans, J2ME. The operating system used to do the implementation is Windows XP. We have developed graphical user interface by using swing application programming interface. Following are some user interface of attack simulation

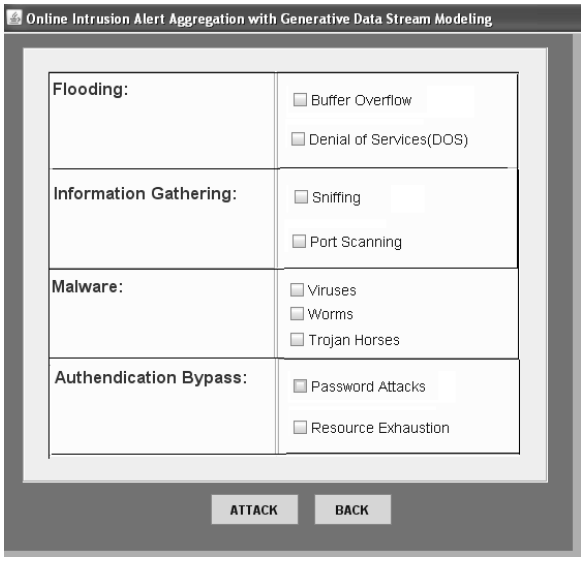

Figure 2: Different Types of Attacks may detected

As shown in figure different attacks can be simulated into information gathering, authentication failure, malwares, and flooding of data. Following is the GUI for alert aggregation

Figure 3: Simulation of Alerts

As shown in above figure there is separate space for each and every layers aggregation messages. When attack is done the relevant or appropriate action or message is displayed as shown in figure

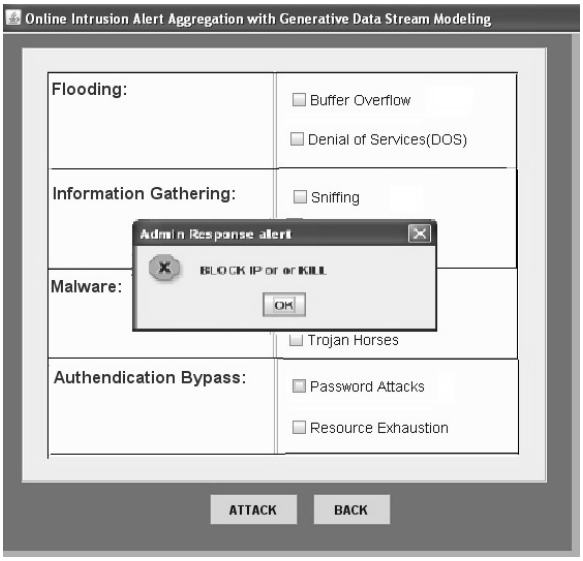

Figure 4: Response when attack is done

Alerts can be sent to users registered mobile as shown in figure.

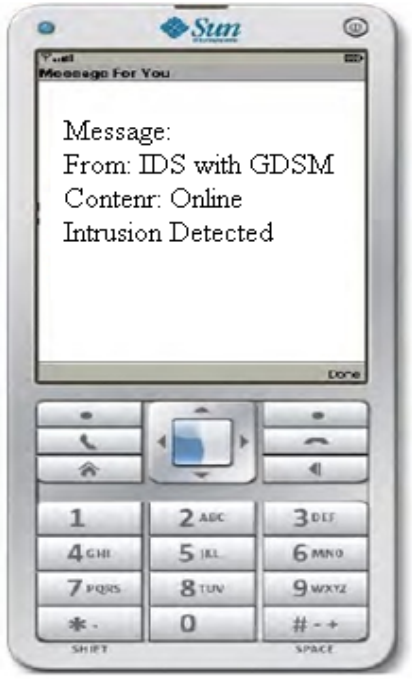

Figure 5: GUI of Mobile Alert

Once the alert is received on the mobile it can be processed by alert processing layer and reaction layer will suggest the way to handle the intruder. The action is also can be send to registered mobile as shown in below figure.

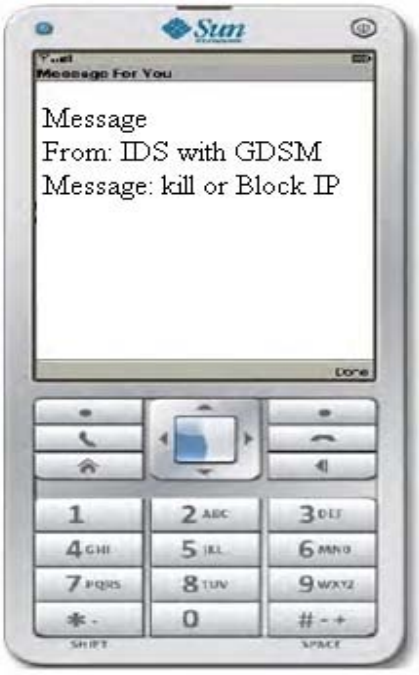

Figure 6: Relative action send by Reaction Layer

5. Conclusion

This paper suggests impressive way for online alert aggregation generation with generative data stream modeling. It has been implemented and it found that meta alerts can be generated effectively. Missing false positive rate gets decreased as it uses property of data streaming i.e. it executes a few times only before processed. The experimental result shows that it is very effective and helpful when it gets implemented in real time application. Also IDS accuracy gets increased very much. More alerts can be detected but compare to number of attacks detected very few false positive alerts gets introduced. So online intrusion alert aggregation with data streaming system is extremely efficient in information technology field to provide security to information.

References

[1] Kothawale Ganesh S., Borhade Sushama R., B. Raviprasad, "Online Intrusion Alert Aggregation with Generative Data Stream Modeling", International Journal of Modern Engineering Research IJMER | ISSN: 2249–6645 Vol. 4 | Iss.7| July. 2014 | 88.

[2] M. Hanock, K. Srinivas, A. Yaganteeswarudu, "Online Intrusion Alert Aggregation with Generative Data Stream Modeling",International Journal of Electronics and computer Science Engineering, ISSN-2277-1956

[3] S. Mangesh kumar, K. Mohan, G. Kadirvelu, S. Muruganandam, "Online Intrusion Alert Aggregation Through Mobile", International journal of advancement in Research and Technology, volume 1, issue3, August-2012

[4] Ravindra Bhat,"Intrusion Detection System with Data Stream Modeling using Conditional Privilages", International journal of computer science and technology, vol.3 no.7july 2012 ISSN:2299-3345

[5] Rupali Shewale, Yugandhar Pandey, Maheshkumar A. Sali, " Distributed Intrusion Alert Aggregation with Data Stream Modeling", International journal of electronics, communication and soft computing science and engineering, ISSN:2277-9477 March-2012

[6] Alexander Hofmann, Bernhard Sick, "Online Intrusion Alert Aggregation with Generative Data Stream Modeling", IEEE transaction on dependable and secure computing, vol 8 No. 2 March-April 2011.

[7] S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," Technical Report 99-15, Dept. of Computer Eng., Chalmers Univ. of Technology, 2000.

[8] M.R. Endsley, "Theoretical Underpinnings of Situation Awareness: A Critical Review," Situation Awareness Analysis and Measurement, M.R. Endsley and D.J. Garland, eds., chapter 1, pp. 3-32, Lawrence Erlbaum Assoc., 2000.

[9] C.M. Bishop, Pattern Recognitin and Machine Learning. Springer, 2006.

[10] M.R. Henzinger, P. Raghavan, and S. Rajagopalan, Computing on Data Streams. Am. Math. Soc., 1999.

[11] A. Allen, "Intrusion Detection Systems: Perspective," Technical Report DPRO-95367, Gartner, Inc., 2003.

[12] F. Valeur, G. Vigna, C. Kru¨ gel, and R.A. Kemmerer, "A Comprehensive Approach to Intrusion Detection Alert Correlation," IEEE Trans. Dependable and Secure Computing, vol. 1, no. 3, pp. 146-169, July-Sept. 2004.

[13] H. Debar and A. Wespi, "Aggregation and Correlation of Intrusion-Detection Alerts," Recent Advances in Intrusion Detection, W. Lee, L. Me, and A. Wespi, eds., pp. 85-103, Springer, 2001.

[14] D. Li, Z. Li, and J. Ma, "Processing Intrusion Detection Alerts in Large-Scale Network," Proc. Int'l Symp. Electronic Commerce and Security, pp. 545-548, 2008.

[15] F. Cuppens, "Managing Alerts in a Multi-Intrusion Detection Environment," Proc. 17th Ann. Computer Security Applications Conf. (ACSAC '01), pp. 22-31, 2001.

[16] A. Valdes and K. Skinner, "Probabilistic Alert Correlation," Recent Advances in Intrusion Detection, W. Lee, L. Me, and A. Wespi, eds.pp. 54-68, Springer, 2001.

[17] K. Julisch, "Using Root Cause Analysis to Handle Intrusion Detection Alarms," PhD dissertation, Universita¨ t Dortmund, 2003.

[18] T. Pietraszek, "Alert Classification to Reduce False Positives inIntrusion Detection," PhD dissertation, Universita¨ t Freiburg, 2006.

[19] F. Autrel and F. Cuppens, "Using an Intrusion Detection Alert Similarity Operator to Aggregate and Fuse Alerts," Proc. Fourth Conf. Security and Network Architectures, pp. 312-322, 2005.

[20] G. Giacinto, R. Perdisci, and F. Roli, "Alarm Clustering for Intrusion Detection Systems in Computer Networks," Machine Learning and Data Mining in Pattern Recognition, P. Perner and Imiya, eds. pp. 184-193, Springer, 2005.

[21] O. Dain and R. Cunningham, "Fusing a Heterogeneous Alert Stream into Scenarios," Proc. 2001 ACM Workshop Data Mining for Security Applications, pp. 1-13, 2001.

[22] P. Ning, Y. Cui, D.S. Reeves, and D. Xu, "Techniques and Tools for Analyzing Intrusion Alerts," ACM Trans. Information Systems Security, vol. 7, no. 2, pp. 274-318, 2004.

[23] F. Cuppens and R. Ortalo, "LAMBDA: A Language to Model a Database for Detection of Attacks," Recent Advances in Intrusion Detection, H. Debar, L. Me, and S.F. Wu, eds. pp. 197-216, Springer,2000.
[24] S.T. Eckmann, G. Vigna, and R.A. Kemmerer, "STATL: An Attack Language for State-Based Intrusion Detection," J. Computer Security, vol. 10, nos. 1/2, pp. 71-103, 2002.
[25] A. Hofmann, "Alarmaggregation und Interessantheitsbewertung in einem dezentralisierten Angriffserkennungsystem," PhD dissertation, Universita¨t Passau, under review.
[26] M.S. Shin, H. Moon, K.H. Ryu, K. Kim, and J. Kim, "Applying Data Mining Techniques to Analyze Alert Data," Web Technologies and Applications, X. Zhou, Y. Zhang, and M.E. Orlowska, eds. pp. 193-200, Springer, 2003.
[27] J. Song, H. Ohba, H. Takakura, Y. Okabe, K. Ohira, and Y. Kwon, "A Comprehensive Approach to Detect Unknown Attacks via Intrusion Detection Alerts," Advances in Computer Science—ASIAN 2007, Computer and Network Security, I. Cervesato, ed., pp. 247-253, Springer, 2008.
[28] R. Smith, N. Japkowicz, M. Dondo, and P. Mason, "Using Unsupervised Learning for Network Alert Correlation," Advances in Artificial Intelligence, R. Goebel, J. Siekmann, and W. Wahlster, eds. pp. 308-31 , Springer, 2008.
[29] A. Hofmann, D. Fisch, and B. Sick, "Identifying Attack Instances by Alert Clustering," Proc. IEEE Three-Rivers Workshop Soft Computing in Industrial Applications (SMCia '07), pp. 25-31, 2007.
[30] M. Roesch, "Snort—Lightweight Intusion Detection for Networks," Proc. 13th USENIX Conf. System Administration (LISA '99), pp. 229-238, 1999.
[31] O. Buchtala, W. Grass, A. Hofmann, and B. Sick, "A Distributed Intrusion Detection Architecture with Organic Behavior," Proc. First CRIS Int'l Workshop Critical Information Infrastructures (CIIW '05), pp. 47-56, 2005.