Prahanthi, Radha Devi, K.Sandhya Rani / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 2,Mar-Apr 2012, pp.1483-1487

Analysis of Intrusion Detection System & Emergence of Online Alert Aggregation

***Prahanthi **Radha Devi ***K.Sandhya Rani**

Abstract — Network Intrusion detection is mainstream to identify alert aggregation and to cluster different alerts produced by low-level intrusion detection systems firewalls etc. Belonging to a specific attack instance which has been initiated by an attacker at a certain point in time, thus, meta-alerts can be generated for the clusters that contain all the relevant information whereas the amount of data (i.e., alerts) can be reduced substantially. Meta-alerts may then be the basis for reporting to security experts or for communication within a distributed intrusion detection system. We propose a novel technique for online alert aggregation which is based on a dynamic, probabilistic model of the current attack situation. Basically, it can be regarded as a data stream version of a maximum likelihood approach for the estimation of the model parameters. In addition, meta-alerts are generated with a delay of typically only a few seconds after observing the first alert belonging to a new attack instance.

Keywords – Network Dataset, Intrusion Detection System, Alert , Attacks, Metadata.

INTRODUCTION I

Intrusion and anomalies are two different kinds of abnormal traffic events in an open network environment. An intrusion takes place when an unauthorized access of a host computer system is attempted. An anomaly is observed at the network connection level. Both attack types may compromise valuable hosts, disclose sensitive data, deny services to legitimate users, and pull down network based computing resources [2]. The intrusion detection system (IDS) offers intelligent protection of networked computers or distributed resources much better than using fixed-rule firewalls. Existing IDSs are built with either signature-based or anomaly-based systems [18]. Signature matching is based on a misuse model, whereas anomaly detection is based on a normal use model. Since the seminal work by Denning in 1981 [1], many intrusion-detection prototypes have been created. Sobirey maintains a partial list of 59 of them. Intrusion-detection systems have emerged in the computer security area because of the difficulty of ensuring that an information system will be free of security flaws. Indeed, a taxonomy of security flaws by Landwehr et al. [3] shows that computer systems suffer from security vulnerabilities regardless of their purpose, manufacturer, or origin, and that it is technically difficult as well as economically costly (in terms of both building and maintaining such a system) to ensure that computer systems and networks are not susceptible to attacks. An intrusion detection system acquires information about an information system to perform a diagnosis on the security status is to discover breaches of security attempted breaches or open vulnerabilities that could lead to potential breaches.

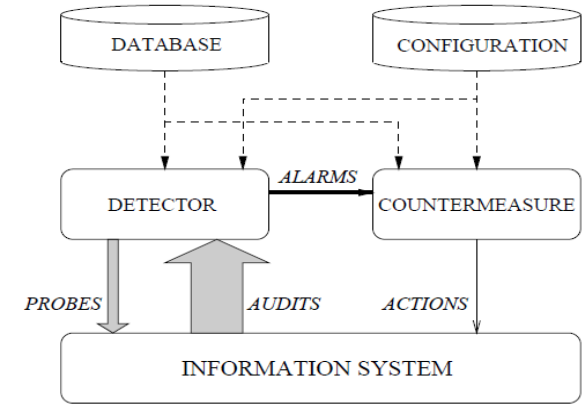


Figure 1 states the model of simple intrusion Detection system.

Intrusion detection system can be described at a very macroscopic level as a detector that processes information coming from the system to be protected. Detector can also launch probes to trigger the audit process such as requesting version numbers for applications using three kinds of information such as long-term information related to the technique used to detect [4] intrusions, configuration information about the current state of the system and audit information describing the events that are happening on the system. Role of the detector is to eliminate unneeded information from the audit trail presents either a synthetic view of the security related actions taken during normal usage of the system, or synthetic view of the current security state of the system.

SECTION II

2.Survey on Intrusion Detection System: Intrusion detection systems (IDS) process large amounts of monitoring data. As an example, a host-based IDS examines log files on a computer (or host) in order to detect suspicious activities. Network-based IDS, on the other hand, searches network monitoring data for harmful packets or packet flows.

2.1. Types of Intrusion Detection System

2.1.1 Network Intrusion Detection System: Network–based intrusion detection system (NIDS) [8] that tries to detect malicious activity such as denial of service attacks, port scan or even attempts to crack into computer by monitoring network traffic. NIDS does this by reading all incoming packets and trying to find number of TCP connection requests to a very large number of different ports is observed, one could assume that there is someone conducting a port scan of some or all of the computers in the network. It mostly tries to detect incoming shell codes in the same manner that an ordinary intrusion detection system does. Often inspecting valuable information about an ongoing intrusion can be learned from outgoing or local traffic and also work with other systems as well, for example update some firewalls blacklist with the IP address of computers used by suspected crackers.

2.1.2. Host-based Intrusion Detection System: Host-based intrusion detection system (HIDS) [8] monitors parts of the dynamic behavior and the state of computer system, dynamically inspects the network packets. A HIDS could also check that appropriate regions of memory have not been modified, for example- the system-call table comes to mind for Linux and various v table structures in Microsoft Windows. For each object in question usually remember its attributes (permissions, size, modifications dates) and create a checksum of some kind (an MD5, SHA1 hash or similar) for the contents, if any, this information gets stored in a secure database for later comparison (checksum-database). At installation time- whenever any of the monitored objects change legitimately- a HIDS must initialize its checksum database by scanning the relevant objects. Persons in charge of computer security need to control this process tightly in order to prevent intruders making un-authorized changes to the database.

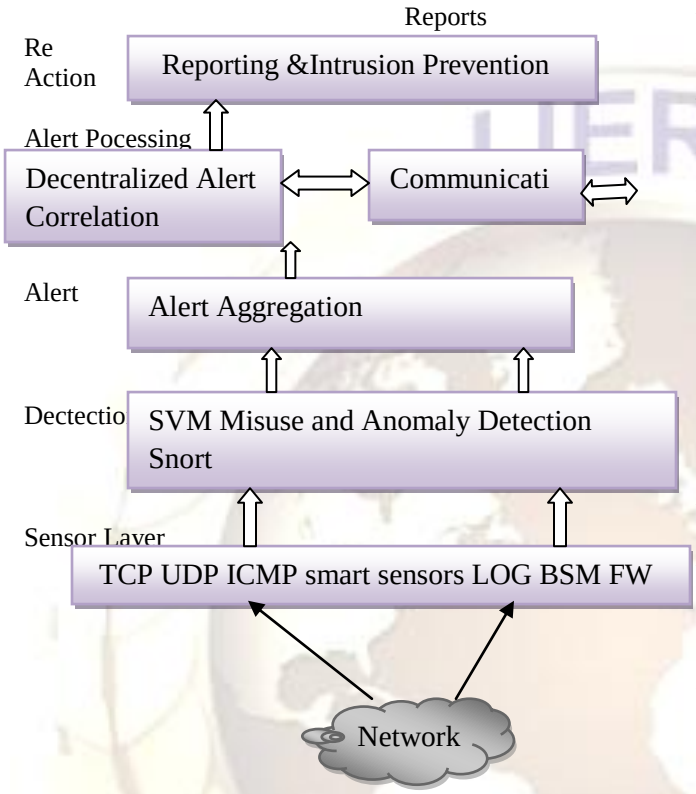
2.1.3. Protocol-based Intrusion Detection system: Protocol based intrusion detection system (PIDS) [8] typically installed on a web server, monitors the dynamic behavior and state of the protocol, and typically consists of system or agent that would sit at the front end of a server, monitoring the HTTP protocol stream. Because it understands the HTTP protocol relative to the web server/system it is trying to protect it can offer grater protection than less in-depth techniques such as filtering by IP address or port number alone, however this greater protection comes at the cost of increased computing on the web server and analyzing the communication between a connected device and the system it is protecting.

2.1.4. Application Protocol-based Intrusion Detection System: Application protocol-based intrusion detection system (APIDS) [16] will monitor the dynamic behavior and state of the protocol and typically consists of a system or agent that would sit between a process, or group of servers, monitoring and analyzing the application protocol between two connected
devices.

2.2 Approach of Existing IDS: Existing IDS are optimized to detect attacks with high accuracy still have various disadvantages that have been outlined in a number of publications and a lot of work has been done to analyze IDS. Correlation approach is attack thread reconstruction, which can be seen as a kind of attack instance recognition. No clustering algorithm is used, but a strict sorting of alerts within a temporal window of fixed length according to the source, destination, and attack classification (attack type). In [7], a similar approach is used to eliminate duplicates, i.e., alerts that share the same quadruple of source and destination address as well as source and destination port. In addition, alerts are aggregated (online) into predefined clusters (so-called situations) in order to provide a more condensed view of the current attack situation. The definition of such situations is also used in [8] to cluster alerts. In [9], alert clustering is used to group alerts that belong to the same attack occurrence. Even though called clustering, there is no clustering algorithm in a classic sense. The alerts from one (or possibly several) IDS are stored in a relational database and a similarity relation—which is based on expert rules—is used to group similar alerts together. Two alerts are defined to be similar, for instance, if both occur within a fixed time window and their source and target match exactly. As already mentioned, these approaches are likely to fail under real-life conditions with imperfect classifiers (i.e., low-level IDS) with false alerts or wrongly adjusted time windows.

Another approach to alert correlation is presented in an approach to alert correlation is presented in [10]. A weighted, attribute-wise similarity operator is used to decide whether to fuse two alerts or not. However, as already stated in and [5], this approach suffers from the high number of parameters that need to be set. The similarity operator presented in [6] has the same disadvantage there are lots of parameters that must be set by the user and there is no or only little

guidance in order to find good values. In [7], another clustering algorithm that is based on attribute-wise similarity measures with user defined parameters is presented. However, a closer look at the parameter setting reveals that the similarity measure, in fact, degenerates to a strict sorting according to the source and destination IP addresses and ports of the alerts.



SECTION III

3. Problem Definition: Network Security is an important issue in our current Task. Providing certain firewall, antivirus, security for password become hectic even though we cant control the network type of anomalies or misuse activities. To maintain the security issue at a time we introduce a novel method call online alert aggregation which is based on dynamic or probabilistic model of the current attack situation.

3.1. Application Specific for Alert Aggregation

3.1.1. Server: Server module is the main module for this project. This module acts as the Intrusion Detection System. This module consists of four layers viz. sensor layer (which detects the user/client etc.), Detection layer, alert processing layer and reaction layer. In addition there is also Message Log, where all the alerts and messages are stored for the references. This Message Log can also be saved as Log file for future references for any network environment.

3.1.2 Client: Client module is developed for testing the Intrusion Detection System. In this module the client can enter only with a valid user name and password. If an intruder enters with any guessing passwords then the alert is given to the Server and the intruder is also blocked. Even if the valid user enters the correct user name and password, the user can use only for minimum number of times. For example even if the valid user makes the login for repeated number of times, the client will be blocked and the alert is sent to the admin. In the process level intrusion, each client would have given a specific process only. For example, a client may have given permission only for P1process. If the client tries to make more then these processes the client will be blocked and the alert is given by the Intrusion Detection System. In this client module the client can be able to send data. Here, when ever data is sent Intrusion Detection System checks for the file. If the size of the file is large then it is restricted or else the data is sent.

3.1.3. DARPA Dataset: This module is integrated in the Server module. This is an offline type of testing the intrusions. In this module, the DARPA Data Set is used to check the technique of the Online Intrusion Alert Aggregation with Generative Data Stream Modeling. The DARPA data set is downloaded and separated according to each layers. So we test the instance of DARPA Dataset using the open file dialog box. Whenever the dataset is chosen based on the conditions specified the Intrusion Detection System works.

3.1.3. Mobile: This module is developed using J2ME. The traditional system uses the message log for storing the alerts. In this system, the system admin or user can get the alerts in their mobile. Whenever alert message received in the message log of the server, the mobile too receives the alert message.

3.1.3. Attack Simulation: In this module, the attack simulation is made for ours elf to test the system. Attacks are classified and made to simulate here. Whenever an attack is launched the Intrusion Detection System must be capable of detecting it. So our system will also be capable of detecting such attacks. For example if an IP trace attack is launched, the Intrusion Detection System must detect it and must kill or block the process.

3.2. Algorithm for Proposed Architecture IDS:

Step 1: Select the ‘n’ layers needed for the whole IDS.

Step 2: Build Sensor Layer to detect Network and Host Systems.

Step 3: Build Detection Layer based on Misuse and Anomaly detection technique.

Step 4: Classify various types of alerts. (For example alert for System level intrusion or process level intrusion)

Step 5: Code the system for detecting various types of attacks and alerts for respective attacks.

Step 6: Integrate the system with Mobile device to get alerts from the proposed IDS.

Step 7: Specify each type of alert on which category it falls, so that user can easily recognize the attack type.

Step 8: Build Reaction layer with various options so that administrator/user can have various options to select or react on any type of intrusion.

Step 9: Test the system using Attack Simulation module, by sending different attacks to the proposed IDS.

Step 10: Build a log file, so that all the reports generated can be saved for future references.

3.3. Alert Aggregation when the user is at OffLine: Assume that a host with an ID agent is exposed to a certain intrusion situation with one or several attacks launch several attack instances belonging to various attack types. The attack instances each cause a number of alerts with various attributes values.

1. False alerts are not recognized as such and wrongly assigned to clusters: This situation is acceptable as long as the number of false alerts is comparably low.

2. True alerts are wrongly assigned to clusters: This situation is not really problematic as long as the majority of alerts belonging to that cluster is correctly assigned. Then, no attack instance is missed.

3. Clusters are wrongly split: This situation is undesired but clearly unproblematic as it leads to redundant meta-alerts only. Only the data reduction rate is lower, no attack instance is missed.

4. Several clusters are wrongly combined into one: This situation is definitely problematic as attack instances may be missed.

3.4. Dataset Alert Aggregation: Assume that in the environement observed by an ID agent attackers initiate new attack instances that cause alerts for a certain time interval until this attack instance is completed at point in time the ID agent which is assumed to have a model of the current situation.

1. Component adaption: Alerts associated with already recognized attack instances must be identified as such and assigned to already existing clusters while adapting the respective component parameters.

2. Component creation (novelty detection): The occurrence of new attack instances must be stated. New components must be parameterized accordingly.

3. Component deletion (obsoleteness detection): The completion of attack instances must be detected and the respective components must be deleted from the model.

SECTION V

5. Comparative Study: Previous IDS are optimized to detect attacks with high accuracy. However, they still have various disadvantages that have been outlined in a number of publications and a lot of work has been done to analyze IDS in order to direct future research Besides others, one drawback is the large amount of alerts produced. Alerts can be given only in System logs. Existing IDS does not have general framework which cannot be customized by adding domain specific knowledge as per the specific requirements of the users or network administrators. Compare to already known application system our work states that Online Intrusion Alert Aggregation with Generative Data Stream Modeling is a generative modeling approach using probabilistic methods. Assuming that attack instances can be regarded as random processes “producing” alerts, we aim at modeling these processes using approximate maximum likelihood parameter estimation techniques. Thus, the beginning as well as the completion of attack instances can be detected. It is a data stream approach, i.e., each observed alert is processed only a few times. Thus, it can be applied online and under harsh timing constraints. In the proposed scheme of Online Intrusion Alert Aggregation with Generative Data Stream Modeling, we extend our idea of sending Intrusion alerts to the mobile. This makes the process easier and comfortable. Online Intrusion Alert Aggregation with Generative Data Stream Modeling does not degrade system performance as individual layers are independent and are trained with only a small number of features, thereby, resulting in an efficient system. Online Intrusion Alert Aggregation with Generative Data Stream Modeling is easily customizable and the number of layers can be adjusted depending upon the requirements of the target network. Our framework is not restrictive in using a single method to detect attacks. Different methods can be seamlessly integrated in our framework to build effective intrusion detectors.Our framework has the advantage that the type of attack can be inferred directly from the layer at which it is detected. As a result, specific intrusion response mechanisms can be activated for different attacks.

Refernces

[1] U.M. Fayyad and K.B. Irani, "Multi-Interval Discretization of Continuous-Valued attributes from Classification Learning," Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '93), pp. 1022-1027, 1993.

[2] K. Hwang, Y. Chen, and H. Liu, "Defending Distributed Computing Systems from Malicious Intrusions and Network Anomalies," Proc. IEEE Workshop Security in Systems and Networks (SSN '05) held with the IEEE Int'l Parallel & Distributed Processing Symp., 2005.

[3] Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi. A taxonomy of computer program security aws. ACM Computing Surveys, 26(3):211{254, September 1994.

[4] K. Hwang, Y. Kwok, S. Song, M. Cai, Y. Chen, and Y. Chen, "DHT-Based Security Infrastructure for Trusted Internet and Grid Computing," Int'l J. Critical Infrastructures, vol. 2, no. 4, pp. 412- 433, Dec. 2006.

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

[6] 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.

[7] 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 A. Imiya, eds. pp. 184-193, Springer, 2005.

[8] 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.

[9] 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—ASIAN2007, Computer and Network Security, I. Cervesato, ed., pp. 247-253, Springer, 2008.

[10] 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. 308-319, Springer, 2008.

Prahanthi M.Tech Computer Science & Engineering from JNTU Hyderabad B.Tech from Rajiv Gandhi Institute of Technology. Currently she is working at Samskruthi College of Engineering has guided many UG&PG students previously she worked in Tirumala Engineering College. Her research areas are Mobile Computing, Compiler Design, Computer Architecture.

K.Sandhya Rani B.Tech IT from Jayamukhi Institute of Science & Technology M.Tech Software Engineering from CVSR College of Engineering currently working as Asst Prof at Samskruti College of Engineering & Technology having five years of experience in academic has guided many UG students. Her research areas include Databases Web Technology Multimedia Application Development.

Radha Devi M.Tech Computer Science & Engineering from G.Narayanamma Institute of Technology & Science MSc Physics from St.Pious Degree and P.G. College. Currently she is Working at Samskruthi College of Engineering has guided many UG & PG students previously she worked in Tirumala Engineering College. Her research areas are Computer Networks, Network Security, Data Warehousing & Data Mining.