International Journal on Emerging Technologies (Special Issue on NCRIET-2015) **6**(2): 235-238(2015)

ISSN No. (Print) : 0975-8364
ISSN No. (Online) : 2249-3255

Online Intrusion Alert Aggregation with Generative Data Stream Modeling

Sushma Priyadarshini
Department of CSE, BKIT, Bhalki, Karnataka, INDIA

(Corresponding author: Sushma Priyadarshini)

(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Alert aggregation is an important subtask of intrusion detection. The goal is to identify 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.

I. INTRODUCTION

INTRUSION detection systems (IDS) are besides other protective measures such as virtual private networks, authentication mechanisms,or encryption techniques very important to guarantee information security. They help to defend against the various threats to which networks and hosts are exposed to by detecting the actions of attackers or attack tools in a network or host-based manner with misus or anomaly detection techniques [1].

At present, most IDS are quite reliable in detecting suspicious actionsby evaluating TCP/IP connections or log files, for instance. Once an IDS finds a suspicious action, it immediately creates an alert which contains information about the source, target, and estimated type of the attack (e.g., SQL injection, buffer overflow, or denial of service). As the intrusive actions caused by a single attack instance— which is the occurrence of an attack of a particular type that has been launched by a specific attacker at a certain point in time—are often spread over many network connections or log file entries, a single attack instance often results in hundreds or even thousands of alerts. IDS usually focus on detecting attack types, but not on distinguishing between different attack instances. In addition, even low rates of false alerts could easily result in a high total number of false alerts if thousands of network packets or log file entries are inspected. As a consequence, the IDS creates many alerts at a low level of abstraction. It is extremely difficult for a human security expert to inspect this flood of alerts, and decisions that follow from single alerts might be wrong with a relatively high probability. In our opinion, a "perfect" IDS should be situation-aware [2] in the sense that at any point in time it should "know" what is going on in its environment regarding attack instances (of various types) and attackers. In this paper, we make an important step toward this goal by introducing and evaluating a new technique for alert aggregation. Alerts may originate from low-level IDS such as those mentioned above, from firewalls (FW), etc. Alerts that belong to one attack instance must be clustered together and meta-alerts must be generated for these clusters. The main goal is to reduce the amount of alerts substantially without losing any important information which is necessary to identify ongoing attack instances. We want to have no missing meta alerts, but in turn we accept false or redundant meta-alerts to a certain degree.

This problem is not new, but current solutions are typically based on a quite simple sorting of alerts, e.g., according to their source, destination, and attack type. Under real conditions such as the presence of classification errors of the low-level IDS (e.g., false alerts), uncertainty with respect to the source of the attack due to spoofed IP addresses, or wrongly adjusted time windows, for instance, such an approach fails quite often.

Objective

(i) Online Intrusion Alert Aggregation with Generative Data Stream Modelling is a generative modeling approach using probabilistic methods.

Assuming that attack instances can be regarded as random processes "producing" alerts, we aim modeling these processes using approximate maximum likelihood parameter estimation techniques. Thus, the beginning as well as the completion of attack instances can be detected.

(ii) 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.

(iii) In the proposed scheme of Online Intrusion Alert Aggregation with Generative Data Stream Modelling, we extend our idea of sending Intrusion alerts to the mobile. This makes the process easier and comfortable.

(iv) 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.

(v) 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.

(vi) 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.

II. ALGORITHM FOR THE PROPOSED IDS MISUSE BASED DETECTION ALGORITHM

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.

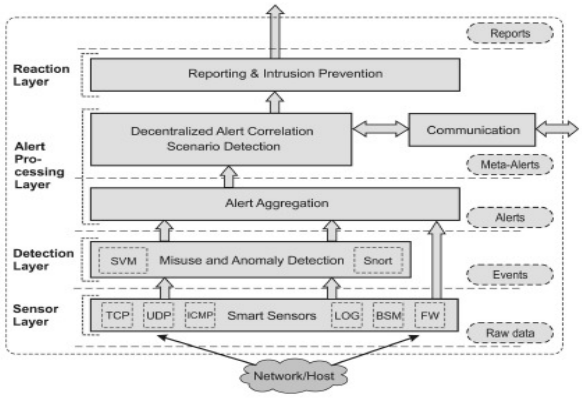

Fig. 1. Architecture of an intrusion detection agent.

III. LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next step is to determine which operating system and language can be used for developing the tool. Most existing 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 (cf. [5], for instance). Besides others, one drawback is the large amount of alerts produced. Recent research focuses on the correlation of alerts from (possibly multiple) IDS. If not stated otherwise, all approaches outlined in the following present either online algorithms or—as we see it—can easily be extended to an online version. Probably, the most comprehensive approach to alert correlation is introduced in [6]. One step in the presented 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 [10].

IV. IMPLEMENTATION

Implementation is the stage of the paper when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

MODULES

Server

Client

DARPA Datasets

Mobile

Attack Simulation

Server

Server module is the main module for this project. This module acts as the Intrusion Detection System.

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 P1 process. If the client tries to make more than 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.

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 Data set using the open file dialog box. Whenever the dataset is chosen based on the conditions specified the Intrusion Detection System works.

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.

Attack Simulation

In this module, the attack simulation is made for our self 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.

V. CONCLUSION

In this paper a novel technique for online alert aggregation. It also addressed the problem of accuracy and efficiency of Intrusion Detection System. It developed the presented architecture and tested the system with the misuse based anomaly detection technique. it also proposed a misuse based anomaly detection algorithm for our system. As our contribution, to make the system more efficient in identify the intrusion alerts and also it extend this work by sending the Alerts as Message to the Network Administrator who governs the Network or Intrusion Detection System. Most of the present existing Intrusion Detection System does not have a generalized framework.

Our proposed architecture is similar to layers, so according to the network environment, the network administrator can add or remove the layers. If a new updated version of detection comes in future, then it will be very easy to add the layer with our proposed system. We also tested our system by launching various attacks to the system, and we found how the system detects and reacts according to the developed IDS. As a future work, this work can be extended as not only to detect attacks and also to prevent attacks. As mentioned earlier, our proposed system allows adding new layers, the prevention layer functionality layer can also be added with our system, as a future work.

REFERENCE

[1]. Autonomous Agents for Intrusion Detection, http://www.cerias. purdue.edu/research/aafid/, 2010.

[2]. CRF++: Yet Another CRF Toolkit, http://crfpp.sourceforge.net/, 2010.

[3]. KDD Cup 1999 Intrusion Detection Data, http://kdd.ics.uci.edu/ databases/kddcup99/kddcup99.html, 2010.

[4]. Overview of Attack Trends, http://www.cert.org/archive/pdf/ attack_trends.pdf, 2002.

[5]. Probabilistic Agent Based Intrusion Detection, http://www.cse.sc. edu/research/isl/agentIDS.shtml, 2010.

[6]. SANS Institute—Intrusion Detection FAQ, http://www.sans.org/ resources/idfaq/, 2010.

[7]. T. Abraham, IDDM: Intrusion Detection Using Data Mining Techniques, http://www.dsto.defence./gov.au/publications/ 2345/DSTO-GD-0286.pdf, 2008.

[8]. R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proc. ACM SIGMOD, vol. 22, no. 2, pp. 207-216, 1993.

[9]. N.B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs. Decision Trees in Intrusion Detection Systems," Proc. ACM Symp. Applied Computing (SAC '04), pp. 420-424, 2004.

[10]. J.P. Anderson, Computer Security Threat Monitoring and Surveillance, http://csrc.nist.gov/publications/history/ande80.pdf, 2010. 11. R. Bace and P. Mell, Intrusion Detection Systems, Computer Security Division, Information Technology Laboratory, Nat'l Inst. of Standards and Technology, 2001.