ISSN 2278-7763

# ONLINE INTRUSION ALERT AGGREGATION THROUGH MOBILE

**S.Magesh kumar[1], K.Mohan[2], G.Kadirvelu[3], S.Muruganandam[4]**

Department of Computer Science and Engineering

Thirumalai Engineering College, Kanchipuram, India

Email: mageshkumars@yahoo.com[1], jackmoh2000.2009@gmail.com[2], kadir.cse@gmail.com[3], murugansoft@hotmail.com[4]

## ABSTRACT:

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 approximative maximum likelihood parameter estimation techniques. Thus, the beginning as well as the completion of attack instances can be detected. In the proposed scheme of Online Intrusion Alert Aggregation, 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

**Keywords:** Alert Aggregation, Intrusion Detection, Data Stream

## I. INTRODUCTION

An intrusion detection system (IDS) is a device or software application that monitors network and/or system activities for malicious activities or policy violations and produces reports to a Management Station.[1] Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system.[1] Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, and reporting attempts.[1] 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.[1] IDPSes have become a necessary addition to the security infrastructure of nearly every organization.[1]

IDPSes typically record information related to observed events, notify security administrators of important observed events, and produce reports.[1] Many IDPSes can also respond to a detected threat by attempting to prevent it from succeeding.[1] 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.[1]

### A. TYPES

There are two main types of IDS:

- **Network intrusion detection system (NIDS)**

It is an independent platform that identifies intrusions by examining network traffic and monitors multiple hosts. Network intrusion detection systems gain access to network traffic by connecting to a network hub, network switch configured for port mirroring, or network tap. In a NIDS, sensors are located at choke points in the network to be monitored, often in the demilitarized zone (DMZ) or at network borders. Sensors capture all network traffic and analyzes the content of individual packets for malicious traffic. An example of a NIDS is Snort

- **Host-based intrusion detection system (HIDS)**

It consists of an agent on a host that identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability databases, Access control lists, etc.) and other host activities and state. In a HIDS, sensors usually consist of a software agent. Some application-based IDS

ISSN 2278-7763

are also part of this category. An example of a HIDS is OSSEC.

**Our approach has the following distinct properties:**

- It is a ***generative modeling approach*** [2] using probabilistic methods. Assuming that attack instances can be regarded as random processes "producing" alerts, we aim at modeling these processes using approximative 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 [3]. Thus, it can be applied online and under harsh timing constraints.

- It is a ***data stream approach***, i.e., each observed alert is processed only a few times [3]. Thus, it can be applied online and under harsh timing constraints.

## II. RELATED WORKS

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.

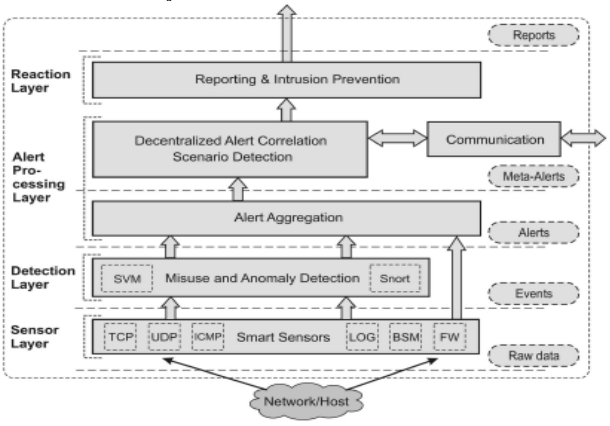



Fig.1. Architecture of an intrusion detection agent.

A completely different clustering approach is presented in [4]. There, the reconstruction error of an autoassociator neural network (AA-NN) is used to distinguish different types of alerts. Alerts that yield the same (or a similar) reconstruction error are put into the same cluster. The approach can be applied online, but an offline training phase and training data are needed to train the AA-NN and also to manually adjust intervals for the reconstruction error that determine which alerts are clustered together. In addition, it turned out that due to the dimensionality reduction by the AA-NN, alerts of different types can have the same reconstruction error which leads to erroneous clustering.

In our prior work, we applied the well-known c-means clustering algorithm in order to identify attack instances [5]. However, this algorithm also works in a purely offline manner.

**ALGORITHM FOR THE PROPOSED IDS**

*Misuse and Anomaly 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.

**ON LINE INTRUSION ATTACKS**

A "perfect" IDS should be situation aware 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 article we make an

ISSN 2278-7763

important step towards 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, 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.

## III. A NOVEL ONLINE ALERT AGGREGATION TECHNIQUE WITH MOBILE

In this section, we describe our new alert aggregation approach which is—at each point in time—based on a probabilistic model of the current situation. To outline the preconditions and objectives of alert aggregation, we start with a short sketch of our intrusion framework. Then, we briefly describe the generation of alerts and the alert format. We continue with a new clustering algorithm for offline alert aggregation which is basically a parameter estimation technique for the probabilistic model. After that, we extend this offline method to an algorithm for data stream clustering which can be applied to online alert aggregation. Finally, we make some remarks on the generation of meta-alerts.

Along with this technique, we introduced mobile simulation software for java that is Wireless Tool Kit. It is mainly used to simulation of mobile. In our paper, we introduced all type of alerts send it through mobile so we implemented that process by using Wireless Tool Kit.

### A. COLLABORATING INTRUSION DETECTION AGENTS

Intrusion Detection agents through self-organized collaboration form a distributed intrusion detection system. The sensor layer provides the interface to the network and the host on which the agent resides. Sensors acquire raw data from both the network and the host, Filter incoming data, and extract interesting and potentially valuable (e.g, statistical) information needed to construct an appropriate event. At the detection layer, different detectors assess the attack events and search for known attack signatures (misuse detection) and suspicious behavior (anomaly detection).In case of attack suspicion, they create alerts and forwarded to the alert processing layer. Alerts may also be produced by firewalls (FW). At the alert processing layer, the alert aggregation combine alerts assumed to belong to a specific attack instance. Meta alerts are generated. [5]

### B. ALERT GENERATION

Comments on the information contained in alerts, the objects aggregated, and on their format. Sensors determine the values of attributes used as input for the detectors and for alert clustering. Attributes in an event that are independent of a particular attack instance are used for classification. Attributes are (or might be) dependent on the attack instance used in an alert aggregation process to distinguish different attack instances. Clearly dependent such as the source IP address which can identify the attacker. Clearly independent such as the destination port which usually is 80 in case of web based attacks. Both such as the destination port which can be a hint to the attacker's actual target service as well as an attack tool specifically designed to target a particular service only[6]

### C. OFFLINE ALERT AGGREGATION

Off-line algorithm for alert aggregation will be extended to a data stream algorithm for on-line aggregation one or several attackers launch several attack instances belonging to various attack types. The attack instances each cause a number of alerts with various attribute values. The task of the alert aggregation is to estimate the assignment to instances by using the unlabeled observations only and by analyzing the cluster structure in the attribute space. Reconstruct the attack situation. Meta-alerts generated are basically an abstract description of the cluster of alerts assumed to originate from one attack instance. The amount of data is reduced substantially without losing important information. Different potentially problematic situations: False alerts are not recognized as such and wrongly assigned to clusters. Acceptable as long as the number of false alerts is comparably low. True alerts are wrongly assigned to clusters (not really problematic as long as the majority of alerts belonging to that cluster is correctly assigned, no attack instance is missed).

ISSN 2278-7763

### D. DATA STREAM ALERT AGGREGATION

ID agent must be situation-aware and try to keep his model of the current attack situation permanently up to date. There is a trade-off between run-time (reaction time) and accuracy. It is possible to decide upon the existence of a new attack instance when only one observation is made. From a probabilistic viewpoint, state that overall random process is non-stationary regarded as mixing coefficients at certain points in time.

A mixing coefficient is either zero or the reciprocal of the number of active components (for the time interval of the respective attack instance).With appropriate novelty and obsoleteness detection mechanisms aim at detecting these points in time with both sufficient certainty and timeliness. Data stream algorithm with short run-times.

### E. META ALERT GENERATION AND FORMAT

With the creation of a new component, an appropriate meta-alert that represents the information about the component in an abstract way is created. Every time a new alert is added to a component, the corresponding meta- alert is updated incrementally too. The meta-alert evolves with the component. Meta-alerts is the basis for the task. Meta-alerts exchanged with other ID agents to detect distributed attacks such as one-to-many attacks. Meta-alerts used at various points in time from the initial creation until the deletion of the corresponding component

## IV. PROPOSED SYSTEM

In this proposed system, all the alerts passed through administrator by Short Message Service (SMS). If any unknown person used this system, we can easily find that by getting message from server.

The intrusion detection agents classified as four categories as shown in the fig.1.

***The sensor layer*** provides the interface to the network and the host on which the agent resides. Sensors acquire raw data from both the network and the host, filter incoming data, and extract interesting and potentially valuable (e.g., statistical) information which is needed to construct an appropriate event. At the ***detection layer***, different detectors, e.g., classifiers trained with machine learning techniques such as support vector machines (SVM) or conventional rule-based systems such as Snort [9], assess these events and search for known attack signatures (misuse detection) and suspicious behavior (anomaly detection). In case of attack suspicion, they create alerts which are then forwarded to the alert processing layer. Alerts may also be produced by FW or the like. At the ***alert processing layer***, the alert aggregation module has to combine alerts that are assumed to belong to a specific attack instance. Thus, so called meta-alerts are generated. Meta-alerts are used or enhanced in various ways, e.g., scenario detection or decentralized alert correlation. An important task of the ***reaction layer*** is reporting.

### A.WIRELESS TOOL KIT

The Sun Java Wireless Toolkit (formerly known as Java 2 Platform, Micro Edition (J2ME) Wireless Toolkit) is a state-of-the-art toolbox for developing wireless applications that are based on J2ME's Connected Limited Device Configuration (CLDC) and Mobile Information Device Profile (MIDP), and designed to run on cell phones, mainstream personal digital assistants, and other small mobile devices. The toolkit includes the emulation environments, performance optimization and tuning features, documentation, and examples that developers need to bring efficient and successful wireless applications to market quickly.

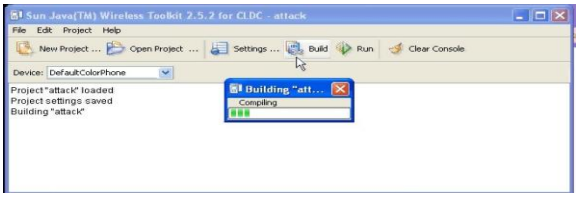



Fig.2. Wireless tool kit

Add the code in this tool then build and run this, you can see simulation of mobile (device).

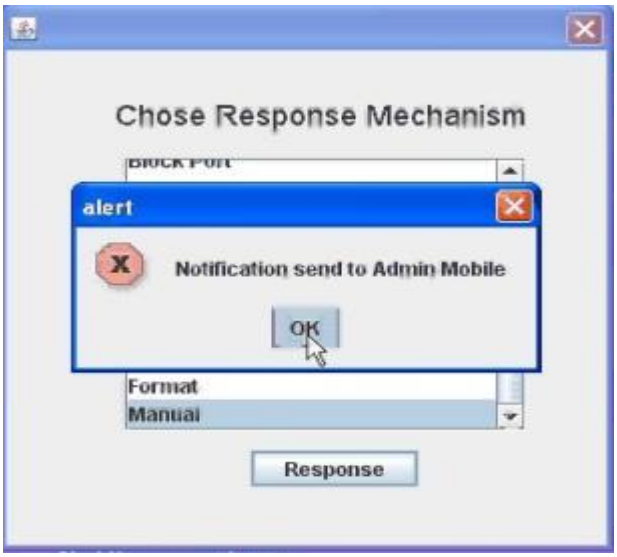
Fig.3. Simulation of Mobile

ISSN 2278-7763

### B.MOBILE ALERT

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.




### C. ATTACK SIMULATION

The attack simulation is made for ourself 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 it 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

Intrusion alert aggregates varies subtask of intrusion detection. Different alerts produced by low-level intrusion detection systems, firewalls, etc is evaluated to make the system more foolproof. The experiments demonstrated for our proposed online alert aggregation through mobile approach. We can see that sample output in this simulation by using sun java wireless tool kit.

## VI. REFERENCES

[1] Scarfone, Karen; Mell, Peter (February 2007). "Guide to Intrusion Detection and Prevention Systems (IDPS)".*Computer Security Resource Center (National Institute of Standards and Technology) (800-94 Retrieved 1 January 2010.*

[2] C.M. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.

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

[4] 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.

[5] 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.

[6] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Chalmers University of Technology, Department of Computer Engineering, Tech. Rep. 99-15, 2000.

[7] A. Allen, "Intrusion detection systems: Perspective," Gartner Inc., London, UK, Tech. Rep. DPRO-95367, 2003.

[8] F. Autrel and F. Cuppens, "Using an intrusion detection alert similarity operator to aggregate and fuse alerts," in 4th Conference on Security and Network Architectures, Batz sur Mer, France, 2005, pp. 312–322.

[9] M. Roesch, "Snort —Lightweight Intrusion Detection for Networks," Proc. 13th USENIX Conf. System Administration (LISA '99), pp. 229-238, 1999.