NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE

DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

COMPUTING SYSTEMS: SOFTWARE AND HARDWARE(SYS)

M.Sc. THESIS

Complex Event Processing(CEP) for Intrusion Detection

Dimitris I. Tsitsigkos

Advisors: **Efstathios Hadjiefthymiades,** Assistant Professor NKUA
Kakia Panagidi, PhD Candidate NKUA

Athens

NOVEMBER 2016

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΥΠΟΛΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ : ΛΟΓΙΣΜΙΚΟ ΚΑΙ ΥΛΙΚΟ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σύνθετη Επεξεργασία Γεγονότων με στόχο την Ανίχνευση Επιθέσεων Ασφάλειας

Δημήτρης Ι. Τσιτσίγκος

Επιβλέποντες: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής ΕΚΠΑ
Κάκια Παναγίδη, Υποψήφιος Διδάκτωρ ΕΚΠΑ

ΑΘΗΝΑ

ΝΟΕΜΒΡΙΟΣ 2016

M.Sc. THESIS

Complex Event Processing for Intrusion Detection

Dimitris I. Tsitsigkos

A.M.: M1283

Advisors: **Efstathios Hadjiefthymiades,** Assistant Professor NKUA
Kakia Panagidi, PhD Candidate NKUA

Examiners: **Efstathios Hadjiefthymiades,** Assistant Professor NKUA
Lazaros Merakos, Professor NKUA

NOVEMBER 2016

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σύνθετη Επεξεργασία Γεγονότων με στόχο την Ανίχνευση Επιθέσεων Ασφάλειας

Δημήτρης Ι. Τσιτσίγκος

Α.Μ.: Μ1283

Επιβλέποντες: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής ΕΚΠΑ
Κάκια Παναγίδη, Υποψήφιος Διδάκτωρ ΕΚΠΑ

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής ΕΚΠΑ
Λάζαρος Μεράκος, Καθηγητής ΕΚΠΑ

ΝΟΕΜΒΡΙΟΣ 2016

ABSTRACT

In this thesis we deal with the usage of data analysis technologies to study the behavior of IoT **Error! Reference source not found.** networks. IoT devices are everywhere, and they're not going away any time soon, including wearable health, connected vehicles and smart grids. But what about security? These systems are able to gather and share huge quantities of sensitive user data. Consumers are constantly exposed to attacks and physical intrusions due to the use of a wide range of available IoT devices, such central control devices for home automation sensors. As we can imagine these devices are inherently insecure (and their users are often unaware of any impending threats), they're easy prey for hackers. In parallel IoT devices can be characterized as low cost, i.e. devices with limited processing power, battery and memory. This means that device-centric solutions for incorporating security and privacy components will be a challenge as well.

The proposed approach offers an application solution to the problem of security intrusions (anomaly-based detection) by using streams generated by IoT devices relevant to their network properties in order to detect abnormal behavior and notify the user via an alert. In our case, each device participating in a IoT network is handled as a sensor device that generates streams of network measurements by using Simple Network Management Protocol (SNMP) **Error! Reference source not found.**.

These measurements are provided as input to Complex Event Processing (CEP) **Error! Reference source not found.** framework, i.e. Esper **Error! Reference source not found.**. CEP listeners detect and analyze the sensor streams in real time based on thresholds related to the normal behavior. Such abnormal statistical behavior can be a clear indication of an event occurrence (e.g., intrusion). Typical measurements of the devices can be combined in order to more accurately observe the outbreak of various security incidents. The estimations of CEP engine will be based on statistical predictors including machine learning methods like ART [4]. We present a number of experiments for the proposed methodologies that show their performance.

SUBJECT AREA: Intrusion, IoT, Real Data Analytics, Machine Learning, Data Mining, SNMP, ESPER, CEP

KEYWORDS: Intrusion, IoT, Real Data Analytics, Machine Learning, Data Mining, SNMP, ESPER, CEP

ΠΕΡΙΛΗΨΗ

Σε αυτή την εργασία ασχολούμαστε με τη χρήση των τεχνολογιών ανάλυσης δεδομένων για τη μελέτη της συμπεριφοράς των δικτύων IoT **Error! Reference source not found.**. Οι συσκευές IoT βρίσκονται παντού γύρω μας και δεν πρόκειται να ξεπεραστούν σύντομα, οπως είναι τα έξυπνα βραχιόλια υγειας , έξυπνες συσκευές που συνδέονται με οχήματα και έξυπνα ενεργειακοί πάροχοι. Αλλά τι γίνεται με την ασφάλεια; Αυτά τα συστήματα είναι σε θέση να συγκεντρώνουν και να μοιράζονται τεράστιες ποσότητες ευαίσθητων δεδομένων του χρήστη. Οι καταναλωτές είναι συνεχώς εκτεθειμένοι σε επιθέσεις και φυσικές εισβολές επειδή χρησιμοποιουν ένα ευρύ φάσμα των διαθέσιμων συσκευών IoT, όπως κεντρικές συσκευές ελέγχου για αισθητήρες οικιακού αυτοματισμού. Όπως μπορούμε να φανταστούμε αυτές οι συσκευές είναι εγγενώς ανασφαλής (και οι χρήστες τους συχνά αγνοούν τις επικείμενες απειλές), και αποτελούν εύκολη λεία για τους επιτιθέμενους. Παράλληλα, οι συσκευές IoT μπορούν να χαρακτηριστούν ως χαμηλού κόστους, δηλαδή συσκευές με περιορισμένη επεξεργαστική ισχύ, μπαταρία και μνήμη. Αυτό σημαίνει ότι οι λύσεις που αφορούν την ασφάλεια των έξυπνων συσκευών, καθώς και τα προσωπικά δεδομένα των χρηστών αποτελουν πρόκληση.

Η προτεινόμενη προσέγγιση προσφέρει μια εφαρμογή που λύνει το πρόβλημα των εισβολών ασφαλείας με τη χρήση δεδομένων που δημιουργούνται από συσκευές IoT που σχετίζονται με τις ιδιότητες του δικτύου τους με σκοπό τον εντοπισμό μη φυσιολογικών συμπεριφορών και ενημερώνει τον χρήστη μέσω ειδοποιήσεων. Στην περίπτωσή μας κάθε συσκευή που συμμετέχει σε ένα δίκτυο IoT αντιμετωπίζεται ως μια συσκευή αισθητήρα που μετράει τα χαρακτηριστικα του δικτύου, χρησιμοποιώντας ένα πρωτόκολλο διαχείρισης δικτύου (SNMP).

Οι μετρήσεις αυτές παρέχονται ως είσοδος σε Σύνθετη Επεξεργασία Γεγονότων (CEP) που ονομάζεται Esper [1]. Οι αισθητήρες του CEP εντοπίζουν και να αναλύουν τα δεδομένα του αισθητήρα σε πραγματικό χρόνο με βάση τα κατώτατα όρια που σχετίζονται με τη φυσιολογική συμπεριφορά. Μια τέτοια διαφορετική συμπεριφορά μπορεί να είναι μια σαφής ένδειξη της εμφάνισης συμβάντος (π.χ. επίθεση). Οι μετρήσεις των συσκευών μπορούν να συνδυαστούν ώστε να μπορούμε να ανιχνεύσουμε διαφόρες επιθέσεις ασφάλειας με μεγαλύτερη σιγουριά. Οι εκτιμήσεις του προγράμματος CEP βασίζεται σε στατιστικούς προγνωστικούς παράγοντες, συμπεριλαμβανομένων των μεθόδων μηχανικής μάθησης όπως ο αλγόριθμος ART. Σας παρουσιάζουμε μια σειρά πειραμάτων για τις προτεινόμενες μεθοδολογίες που δείχνουν την απόδοσή τους.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Επιθέσεις, IoT, Πραγματική Ανάλυση Δεδομένων, Μηχανική Μάθηση, Εξόρυξη Δεδομένων, SNMP, ESPER, CEP

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Επιθέσεις, IoT, Πραγματική Ανάλυση Δεδομένων, Μηχανική Μάθηση, Εξόρυξη Δεδομένων, SNMP, ESPER, CEP

I would like to dedicate this thesis to my beloved family,
to my friends and my colleagues
for their support

CONTENTS

1. INTRODUCTION 3
2. Security Intrusions 4
2.1 Definition of intrusion 4
2.2 Types of Intrusions 4
2.3 Intrusion Detection and Prevention System IDPS 6
2.3.1 Signature-Based Detection 7
2.3.2 Anomaly-Based Detection 8
2.3.2.1 Statistical Anomaly Detection 9
2.3.2.2 Stateful Protocol Analysis Detection 10
2.3.3 Types of IDS 11
2.3.3.1 Host-Based IDS 11
2.3.3.2 Network-based IDS 13
2.3.3.3 Intrusion Prevention System 14
3.Simple Network Management Protocol SNMP 15
3.1 Definition of SNMP 15
3.2 General Architecture 16
3.2.1 SNMP Manager 17
3.2.2 SNMP agent 17
3.2.3 Management Information Base 18
3.3 SNMP messages-types 19
4. Intrusion Detection by using SNMP on IoT using Complex Event Processing 21
4.1 Problem Definition 21
4.2 Challenges 22
4.3 Related work 23
4.3.1 IoT Security Intrusions 23
4.4 What is Complex Event Processing – ESPER 24
4.5 Proposed Methods 24
4.5.1 SNMP 25
4.5.2 Esper Engine 26
4.5.2.1 First layer of the Esper Engine 26
4.5.2.2 Second layer of the Esper Engine - Network Manager 28
4.5.3 Rules: feature extraction 28
4.5.3.1 Shewhart Controller 29
4.5.3.2 Adaptive Reasonance Theory 30

4.6 Experimental Results **31**
4.6.1 Simulation Environment 31
4.6.2 Evaluation Experiments 32
5. Conclusions **38**
References **41**

LIST OF FIGURES

Figure 1 : Definitions of Alerts 7

Figure 2: Signature-Based Detection 8

Figure 3: Anomaly-Based Detection 9

Figure 4: Statistical Anomaly Detection 10

Figure 5: Stateful Protocol Analysis Detection 11

Figure 6: Host-Based IDS 12

Figure 7: Network-Based IDS 13

Figure 8: Basic SNMP Communication [8] 15

Figure 9: Architecture of SNMP [9] 16

Figure 10: Communication Manager-Agent 17

Figure 11: MIB Tree Diagram [10] 19

Figure 12: Internet Of Things [11] 23

Figure 13: CEP Architecture 25

Figure 14 – ART Clustering in two dimensions 30

Figure 15 : All values of TCP in each dataset 33

Figure 16 : All values of Memory in each dataset 33

Figure 17 : All values of UDP in each dataset 33

Figure 18 : All values of Bandwidth in each dataset 34

Figure 19 : All values of CPU in each dataset 34

Figure 20 : All the metrics of the dynamic variables for each dataset 35

Figure 21 : Comparison between ART radius=50, η=1 and ART radius:50, η=0.5 for day 17 36

Figure 22 : Comparison between ART radius=100, η=1 and ART radius:100 , η=0.5 for day 17 36

Figure 23 : event counters for dynamic variables of all ART algorithms for each day 37

Figure 24 : Comparison of Algo1, Shewhart and ART algorithms on day 17 37

LIST OF TABLES

Table 1: Advantages and Disadvantages of Signature-Based Detection 8

Table 2: Advantages and Disadvantages of Statistical Anomaly Detection 10

Table 3: Advantages and Disadvantages of Stateful Protocol Analysis Detection 11

Table 4: Advantages and Disadvantages of Host-Based IDS 12

Table 5: Advantages and Disadvantages of Network-based IDS 13

Table 6: Key differences between IDS and IPS 14

Table 7 : All the counters of the dynamic variables for each dataset 35

PREFACE

This master thesis was carried out in the postgraduate studies program of "Computer Systems: Software and Hardware(SYS)" of the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens. The goal of my research is to study intrusions in IoT and to develop optimized methods in order to detect these attacks and provide a safer environment for IoT.

I would like at this point to express my warmest thanks to my thesis supervisor, Assistant Prof. Efstathios Hadjiefthymiades, for guidance and valuable contributions during my study of the master thesis. I would also like to thank the Kakia Panagidi, Ph.D. candidate of Department of Informatics and Telecommunications, for his helpful guidance and advice throughout the preparation of this work. Their continuous monitoring of the progress of the master thesis, their meaningful remarks and their help for resolving any problem have contributed in the final formation of this thesis.

1. INTRODUCTION

Our days are dominated by the rapidly evolution and progress of technology, and this has affected our everyday life. New devices such as smartphones, tablets and wearables entered in users' lives by helping them to stay motivate, to improve their lives, to monitor remotely their home devices, to inform about the traffic or the pollution of a city and a lot of other applications. The next think was to connect all these devices, in order to exchange data and operate more automated, without requiring human-to-human or human-to-computer interaction. This is called Internet of Things (IoT). The Internet of Things -IoT- is the internetworking of physical devices, vehicles (also referred to as "connected devices" and "smart devices"), buildings and other items embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. A "thing", in the Internet of Things, can be a person with a heart monitor implant, a farm animal with biochip transporter, an automobile that has built-in sensors to alert the driver when tire pressure is low, or any other man-made object that can be assigned an IP address and provided with the ability to transfer data over a network. The IoT is evolving very fast and is becoming an increasing growing topic in conversations about technology. Undoubtedly, IoT is a project that is designed and implemented to make human life easier and simpler, but this help comes with no cost? Evolution of IoT brings new problems and concerns, some technical and others social or environmental. Some of them are lack of security, lack of privacy, storage issues, energy demands and waste disposal.

The security issue is the first short term problem that arises in the new era of IoT. Nowadays, all network devices, such as routers, smart TVs, and smart mobiles are very easy to get hacked. Now you can imagine how huge problem will be created, if hackers can hack so easily billions of smart devices, that belong to a IoT network. Moreover, with the interconnection of the devices, a hacker can take the control of many important things of your life, such as the smart car, the smart home or the smart business. In this thesis, we analyze the security issue of IoT and we designed and implemented a framework for detect intrusion in smart devices. This thesis is organized as follows: in section 2 we explain and analyze the security intrusions in IoT networks. In section 3, we explain the architecture and way that Simple Network Management Protocol (SNMP) works. In the next section, we briefly discuss the related work of intrusion detection systems, knowledge discovery over heterogeneous wireless network, the studied problem with its challenges and the proposed framework is presented in detail with experiments. The last section includes the conclusions.

2. SECURITY INTRUSIONS

2.1 Definition of intrusion

The rapidly evolution in computing, networking, and technology, make the world seems to be more and more connected. Internet connects millions of computers in all over the world. The Internet is a network of networks and consists of billions of users across private, public, university, and government networks sharing information across the networks. The Internet uses TCP/IP protocol and the underlying physical media can be wire, optical, or wireless technologies. The Internet serves a huge range of applications, such as e-mail, the World Wide Web (www), and social networks. Each application may use one or more protocols. There is a large amount of personal, commercial, business, government, and military information being shared on the Internet and there are billions of users, both good and bad, accessing the Internet. The bad guys, known as hackers and such other persons with malicious intent are a concern.

With so many computers, networking devices, protocols, and applications on the network, it has become a serious threat to information security. It is very important to keep user information and network safe and reliable because any application, network device, or protocol can be vulnerable in hacker attacks. An intrusion is an event that can occur by taking advantage of any vulnerabilities that exist in the network and can provoke a lot of simple or significant problems, from losing personal data to take full control of your device.

2.2 Types of Intrusions

The primary classes of threats to network security are internal and external threats:

- **Internal Threats:** Internal threats are threats from someone inside access in the same network, and network resources, who understands the network infrastructure well, who understands the security applications and the security loop holes. Someone within the network can create and send out attacks by hiding his identity as he already knows enough inside information. These threats are very common and the easiest, because hacker has a lot of information about the network.
- **External Threats:** External threats are threats from another network. They do not possess authorized access to the network resources. They work by gaining unauthorized access to the network and network resources with the intention of damaging the resources or for profit. These attacks are more difficult than the internal attacks, because hacker needs to discover vulnerabilities and information of the network, in order to make an attack. These can be structured or unstructured:
    - **Structured:** Structured attacks come from technically competent hackers who belong to a class of highly motivated individuals. They understand vulnerabilities and develop sophisticated tools and techniques to penetrate without anyone knowing. **Unstructured:** These threats are from inexperienced individuals testing their skills using some of the tools available in the public domain. Sometimes, these can do serious damage to an operating system.

Attackers generally abuse the network “rules” established by security policies. The rules are broken in such a way that attackers send their traffic that appears to be normal traffic. Attacks can be classified into the following categories:

- Reconnaissance: For an effective attack, the hacker should have some knowledge about the hardware, the software and the topology, that are used in the network. So, before the attack, the hacker tries to collect as much information about the network, which is called reconnaissance. Reconnaissance is not an attack by itself, but is the preprocessing step for next possible attacks. Some reconnaissance methods are: sniffing, pinging, banner grabbing and port scanning. Information that attackers try to found are IP address range, server location, running OS, software version and types of devices.
- Denial of Service (DOS)/Distributed Denial of Service (DDoS) **DoS (DDoS)** attack is an explicit attack to block users from accessibility to the network and network services, such as flood the network, thereby preventing legitimate network traffic, target single device with too many requests thus bringing down the device, disrupt the connections between two legitimate devices thereby preventing access to a genuine service request, destruction or alteration of network configurations, consume the network bandwidth.
- Other network attacks
    - **Masquerade/Spoofing Attacks:** The network attacker masquerades the TCP/IP packet by an illegal IP address, giving a distorted source address. The intruder confuses the remote machine because sending a fake source address but with valid user access privileges. In an IP spoofing attack, a hacker from outside the network hacks into the network pretending to be a trusted user, of the network, and spoofs the source address of a legal inside user thus gaining access to the network resources. This attack can also cause a broadcast in the network causing high network traffic. If the attacker manages to alter the routing tables, then response from the network resource can go to the spoofed destination address.
    - **ARP Spoofing & DNS Spoofing:** The Address Resolution Protocol (ARP) spoofing is used to mix up the system to correspond incorrect MAC address to a particular IP address in the ARP table. Similarly, DNS (Domain Name Service protocol) spoofing is to change the mapping of DNS entries in the DNS cache. Mac Flooding attacks are also similar to this.
    - **HTTP Tunneling:** This method may be used by the attackers in order to pass the firewall controls and send confidential information to the outside world without anyone inside being aware of the same.
    - **SSH Tunneling:** These may be used to directly connect to a network stealthily and initiate attacks. This is an illegitimate use of a legitimate tool.
    - **Session Hijacking:** A session between the user and the server can be hijacked by the attacker, by using and intermediate session. Some of the methods used in this regard are session fixing and session prediction.
    - **Attacks on Network Equipment including Routers:** Routers and modems are traditionally prone to default password vulnerabilities because the administrators not taking sufficient care in resetting these passwords. The weakness of the network configurations of a router is a new point of vulnerability. Another serious problem is, that some vendors have a so-called “back-door” to their system for debugging purposes and to support the client in case an admin password is forgotten or lost, which can easily be exploited, if it is known to the attackers.

2.3 Intrusion Detection and Prevention System IDPS

An intrusion is defined as an unwanted or unauthorized interference to network normally with bad intentions. The intention of an attacker is, at first to discover information related to the organization network such as the structure of the internal networks or software systems like operating systems, tools/utilities, or software applications used by the organization and then initiate connections to the internal network and execute attacks. Intrusions are normally provoked by attackers outside the organization. Sometimes, intrusions can be caused by internal authorized persons executing these attacks by misusing their authorization or by internal authorized persons who go beyond their area of authorization and such attacks also need to be protected against.

An Intrusion Detection System (IDS) is a combination of both hardware and software that discovers the attacks into a system or network. IDS complements a firewall by providing a thorough inspection of both the packets' header and its contents thus protecting against intrusions, which are otherwise perceived by a firewall as seemingly benign network traffic.

Firewalls look at the control rules and decide if a packet is either allowed or denied. A rule specifies whether a host or a network, or an application should be allowed into the trusted network. To check the rules, a firewall has to just inspect the header of the TCP/IP protocol such as FTP, HTTP, or Telnet. However, it does not control the data contents of the network packet, so if the data contains a malicious code, the firewall will allow this packet to pass through as the packet header has conformed to the rules configured in the firewall. Hence, a user can still have a firewall but your trusted network can be compromised.

IDS control each and every packet's content cross the network to discover any malicious behaviors. Every packet is peeled all the way down to the data content part and the data content is inspected for any malicious code and then the packet is reassembled back to its original form and then the packet is sent along

Before analyzing the detection methods, we should define some terms, which we will use **Error! Reference source not found.**:

- False Positives: These are alerts indicating that something is not right when actually it is right.
  Example: The IDS finds a packet as having malicious code but it is actually a genuine code.
- False Negatives: These are alerts that something is right when actually it is wrong. Example: The IDS finds that a packet does not have any malicious code but it actually does contain a malicious code, as found through investigation.
- True Positives: These are alerts that something is not right when it is actually not right. Example: The IDS finds a packet as containing malicious code and it was actually true that the packet had malicious code, as confirmed by investigation.
- True Negatives: These are alerts that something is right when it is actually right. Example: The IDS finds a packet as containing no issues and it actually had no issues.

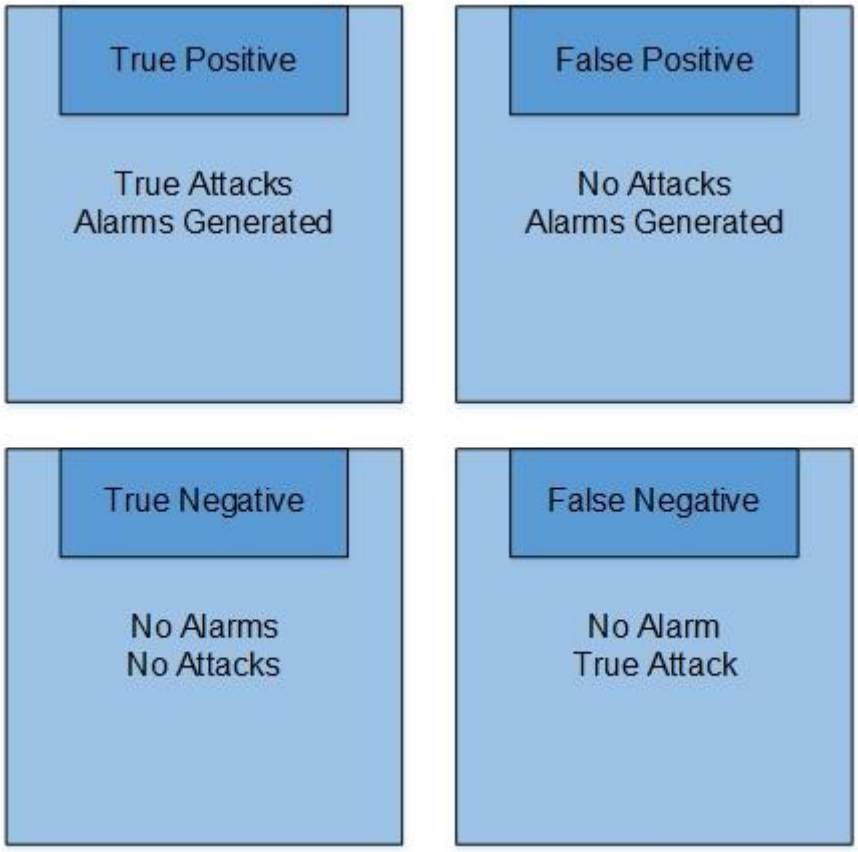

Figure 1 : Definitions of Alerts

2.3.1 Signature-Based Detection

Signature-based detection is the simplest type of detection because it just compares the traffic with the signature database. If the comparison returns a match an alert is generated, in every other case the traffic continues without any problem. In signature-based detection, detection is based on comparing the traffic with the known signatures for possible attacks. It can only discover known threats and so, are not effective in discovering threats, that are unknown. To discover an intrusion, the signature matching should be precise, otherwise, even if the intrusion has a small change from the known threat signature, then the system will not be able to detect. And as a result, it is very easy for the hackers to compromise and breach into the trusted network.

Signature database needs to be updated constantly, practically on a daily basis from the anti-virus labs such as McAfee, Symantec, and other security providers. If the signature is not the latest version, the IDS systems will fail to discover some of the intrusion attacks. The other disadvantage is that they have very little knowledge about the previous requests when processing the new ones.

Signature-based detection provides very concrete detection of known threats by comparing network traffic with the threat signature database. There is a possibility to increase the detection if the traffic inside the network can be made to learn specific patterns of the attacks. Signature detection engines intend to relegate performance over a time period as more and more signatures are populated to the database. It takes more and more time for the engine to do a pattern search as the signature database is always growing as more and more definitions are added to it. And as a result, a robust platform is needed for signature detection considering this growth. Table shows the pros and cons of signature-based detection technique.

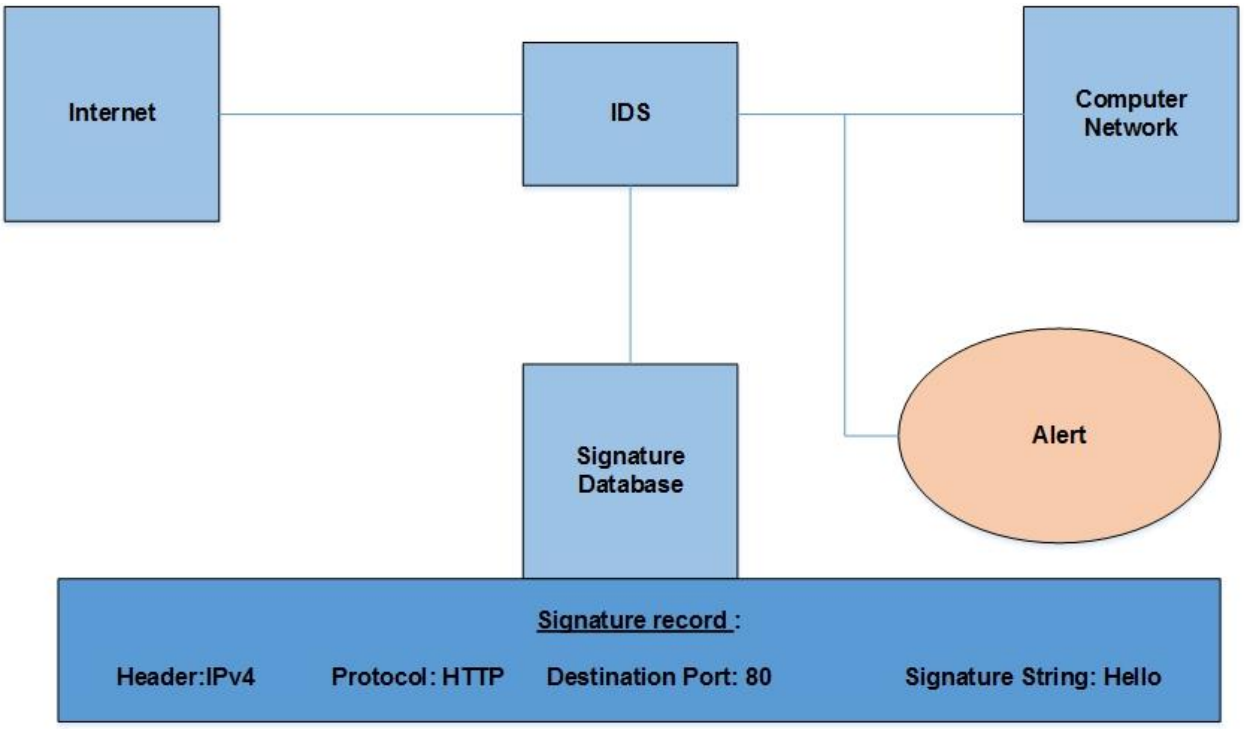

Figure 2: Signature-Based Detection

Advantages	Disadvantages
Simple method	Big number of signatures for a single vulnerability
Application across all protocols	High false positives rate
	High false negatives rate

Table 1: Advantages and Disadvantages of Signature-Based Detection

2.3.2 Anomaly-Based Detection

Anomaly-based detection prevents attacks from unknown threats. If any traffic is found to be abnormal from the threshold, then an alert is triggered by the IDS suspected of an attack. At first, Intrusion Detection and Prevention System(IDPS) creates a threshold profile that represents the normal behavior of the traffic. This profile is created by allowing the IDS system to follow the traffic of the network for a time period, this is like a preprocessing and testing step. After learning, the traffic collected over a period of time is statistically studied and the threshold profile created. Once the IDS is changed from learning mode to detection/prevention mode, it starts comparing the present traffic with the preprocessing profile, and if any abnormality or deviation is found, then an alert is triggered detecting the possible intrusion or the intrusion is prevented, depends on the mode. Customized profiles can also be created for specific traffic behavior such as the number of e-mails sent by a user and user access attempts. Some examples of anomalous behavior are HTTP traffic on a non-standard port and heavy SNMP traffic.

For effective intrusion detection, IDS must have a robust threshold profile which covers the entire organization's network and its segments. It should cover normal traffic

behavior of all the components which aiming to be covered by the IDPS. Threshold profile can vary in complexity from a simple to a comprehensive content, depending on the specifications of the network and its components

The most important step of the anomaly-based detection method is creating an efficient profile. The initial profile, sometimes referred to as the training profile, is generated by studying and analyzing the traffic pattern over a period of time. The time factor depends from the size and variety of data. Once this profile is created, IDS is put into detection mode and every time there is a packet, a pattern is matched against the threshold profile. This threshold can be changed as and when required based on the traffic behavior. If any malicious activity already exists while building the threshold profile, this activity will also become part of it and such type of activity will pass the detection. So, anomaly detection does not necessarily detect each and every unknown attack. The quality and the limitations of the detection, depends on the threshold profile.

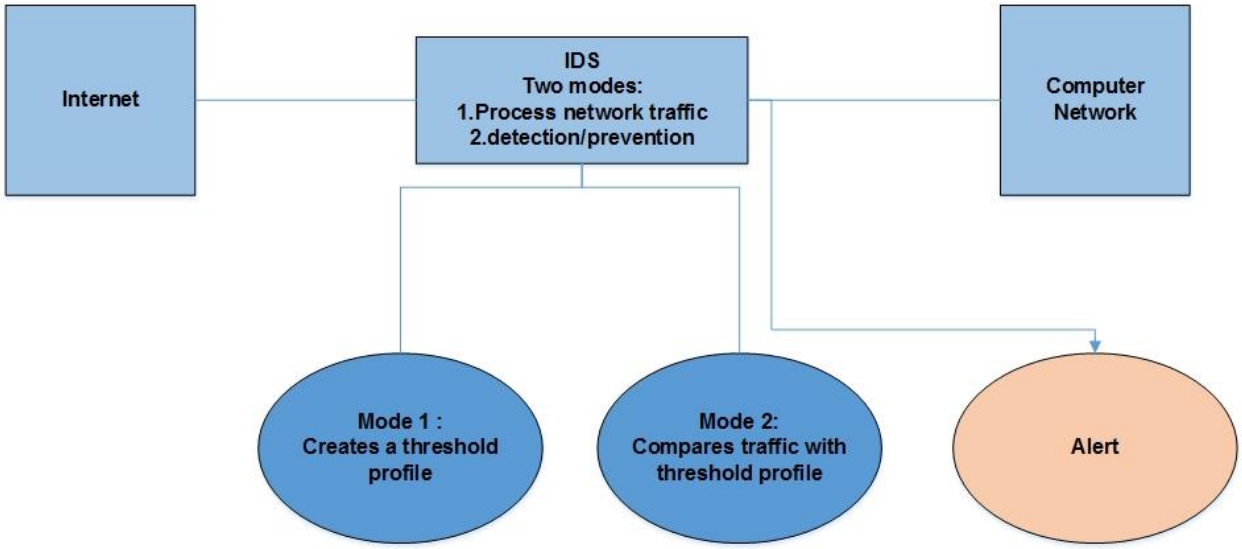

Figure 3: Anomaly-Based Detection

2.3.2.1 Statistical Anomaly Detection

Denial of Service (DoS) and Distributed Denial of Service (DDoS) results in an increasing of traffic on the network which may provoke a lot of dysfunctionalities. To deal with this intrusion, threshold profiles are created on the normal flow of traffic, as described above, based on a statistical model, such as Naïve Bayes, to decide if packets are malicious or not. While knowing about the network traffic behavior, the statistical model calculates the probability score for each of the data packets with normal traffic. The scores are calculated considering as base the sampled data over a period and stored in a threshold profile. Protocols and users are limited from an upper bound. When the IDS is in monitoring mode, the packets are compared against the baseline and the upper bound. Whenever an anomalous packet is detected and the scores are above upper limit, then an alert is triggered. If the data is found to be anomalous for a capable period of time, then will be reported, else the IDPS will ignore it. Profiles based on the statistical measures can discover some of the DoS anomalies based on long- and short-term distributions or in an increasing traffic. When the system is in detection mode, the normal threshold profiles continues to learn the network and they are re-created, in order to adapt the differences in traffic pattern to avoid false positives. By creating different profiles, DoS attacks can be prevented

Advantages	Disadvantages
Detects Unknown Attacks	Prone to false positives
Prevents DoS attacks, Buffer Overflows	Longer detection time
	Analyzing Intrusion may be difficult with Anomaly
	Difficulty in creating threshold profile

Table 2: Advantages and Disadvantages of Statistical Anomaly Detection

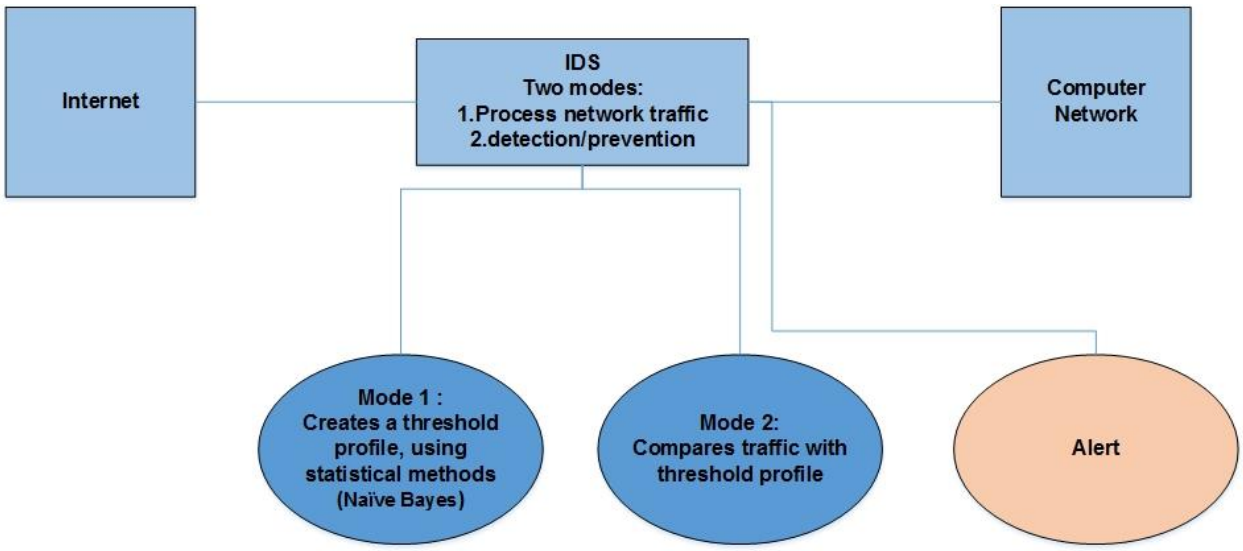

Figure 4: Statistical Anomaly Detection

2.3.2.2 Stateful Protocol Analysis Detection

This method is similar to the anomaly-based detection, except that the profiles are created by the vendors who provide the sensor equipment (IDPS). The profiles are determined in advance and constitute of the generally accepted benign network traffic activity as defined by the standards. Stateful means that the IDPS has the ability to keep track of the state of the protocol both in network layer and application layers. For example, when we have a TCP connection establishment state, the IDS should remember all the states of the connections. After an exchange of some information between the client and the server, the user is authenticated and got access to the network. During this period, the traffic is benign and the IDPS should remember the state.

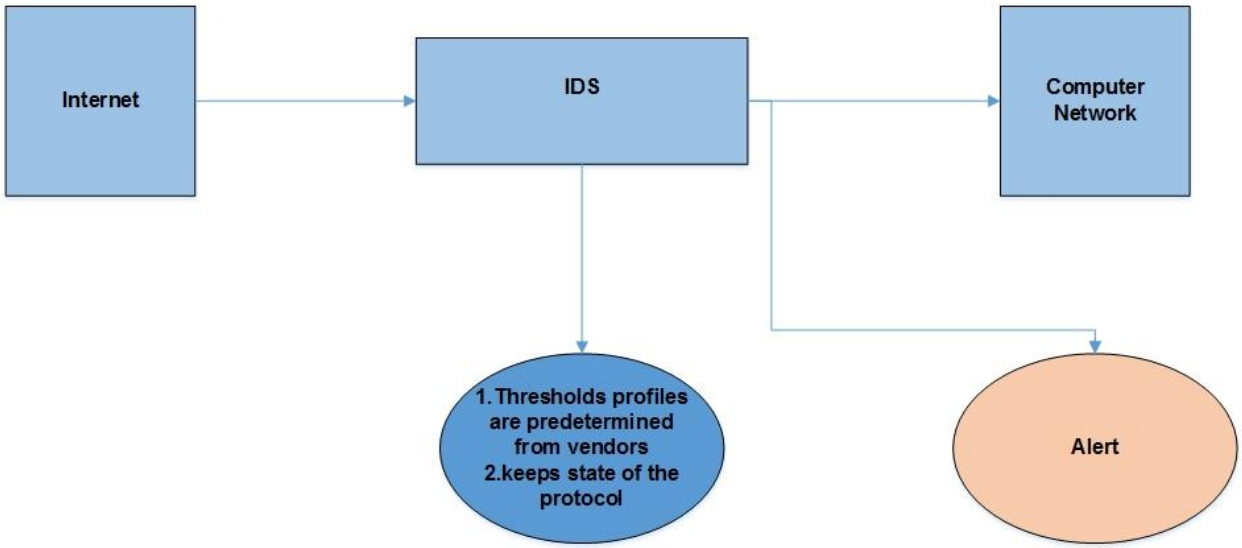

Figure 5: Stateful Protocol Analysis Detection

The stateful protocol anomaly detection method uses profiles that have been created based on standards and specifications defined by the vendor who in general conforms with the variety of the protocols from the standard bodies. If any vendor has implemented protocols, with variation to the standards, for IDPS it would be difficulty in discovering and analyzing the states. In such cases, IDPS protocol models also need to be up to date for the customized protocol changes.

The most important disadvantage of this method is that they are analyzing many protocols, and the IDPS has to keep a list of their states of all of them simultaneously. Furthermore, if an intrusion is within the generally acceptable protocol behavior, then it can get access to the network. Finally, If the protocol implementation is different from operating system to operating system then IDPS may not perform well in discovering the intrusions.

Advantages	**Disadvantages**
Stateful Inspection	Resource intensive
Reasonable checks on the standard protocol before an alert	Cannot detect variations to the generally acceptable protocol behavior policy
	Cannot detect any conflict between the standards and how they are implemented

Table 3: Advantages and Disadvantages of Stateful Protocol Analysis Detection

2.3.3 Types of IDS

2.3.3.1 Host-Based IDS

Host-Based Intrusion Detection System (HIDS) relates to the discovery of attacks on a single system. This is normally a software-based deployment where an agent, is installed on the local host that monitors and reports the application activity. HIDS watches the access to the system and its application and sends alerts for any abnormal activities. It continuously monitors event logs, system logs, application logs, user policy enforcement, rootkit detection, file integrity, and other intrusions to the system. Using this information creates a threshold. HIDS checks, any new log entries appear, against the threshold and if any entries are found outside of this threshold, HIDS triggers an

alert. If any activity, that is not authorized, is detected, HIDS can alert the user or prevent the activity or perform any other decision based on the standards that is configured on the system.

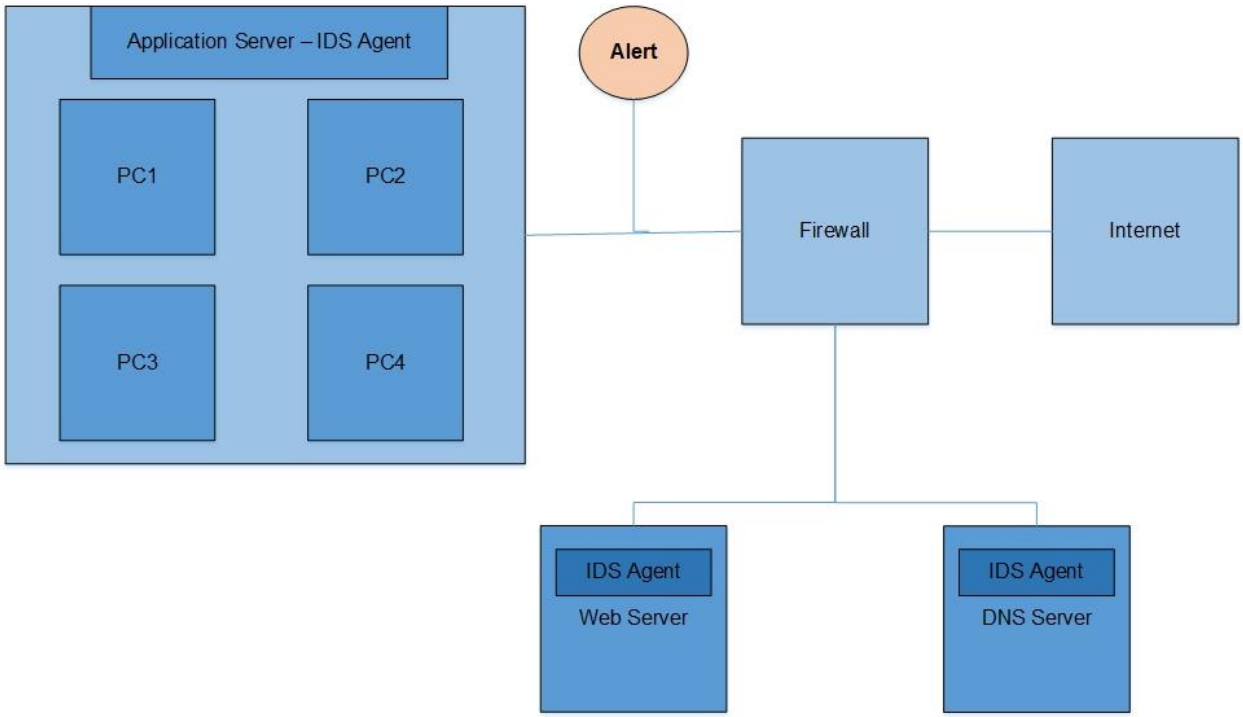

Figure 6: Host-Based IDS

Most of the HIDS systems also, have the ability to prevent attacks. HIDS depends on the logs generated by the system and the fact that the intruders leave evidence of their activities. In general, attackers get access to the system and establish malicious softwares so that future access becomes more easy. If these softwares transform the operating system configurations, or entries of some windows registry, it is logged in the systems/event log, so triggering an alert by the HIDS system. IDS is generally installed on servers, or end point devices to protect the system from intrusion. The operate of HIDS individually depends on the audit trails generated by the system. If hackers achieve to turn off these logs, even if you have a HIDS agent running, it may not find any abnormal activity and no alert will trigger. This is the biggest disadvantage of HIDS.

Advantages	**Disadvantages**
System level protection. Protects from attacks directed to the system	HIDS functionality works only if the systems generate logs and match against the pre-defined policies. If for some reason, systems do not generate logs, HIDS may not function properly
Any unauthorized activity on the system, such as configuration changes, file changes, registry changes, are detected and an alert is generated for further action	If hackers bring down the HIDS server, then HIDS is of no use. This is true for any vulnerability protection software

Table 4: Advantages and Disadvantages of Host-Based IDS

2.3.3.2 Network-based IDS

A Network-Based Intrusion Detection System (NIDS) monitors and discovers any activity on a network that looks suspicious. It checks every packet that is taking access to the network to make sure that it does not comprise any malicious content which would provoke damage to the network or to the end system. NIDS sniffs the network traffic constantly. The traffic is matched against known signature profiles and if there are any abnormal behaviors found in the traffic, then a NIDS triggers an alarm to the management console. A single sensor deployed in promiscuous mode or inline mode can monitor/protect several hosts in the network.

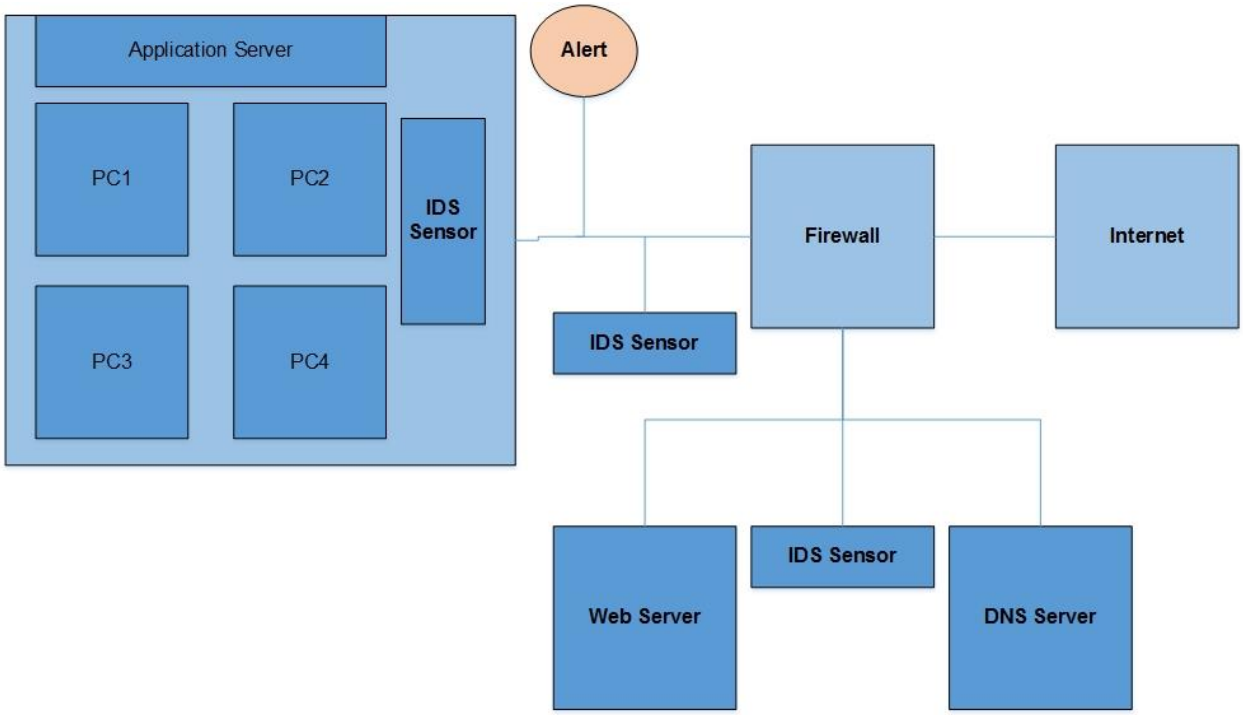

Figure 7: Network-Based IDS

NIDS preserve the network and its resources from the network perspective. For example, network IDS can discover reconnaissance attacks, Denial of Service attacks right at the network level. NIDS triggers alerts when it detects these intrusions. NIDS is a hardware/software solution located near the firewall as an independent device/sensor which running a network operating system. Sensors have interfaces to monitor the network (monitoring interface) and a management interface. These interfaces are used for controlling and receiving alerts, and then send these alarms to the central management controller.

Advantages	**Disadvantages**
Protects network and network resources	Sensor hardware is process intensive
Protects against DoS attacks	Prone to false positives.

Table 5: Advantages and Disadvantages of Network-based IDS

2.3.3.3 Intrusion Prevention System

An Intrusion Prevention System (IPS) is used to prevent the attacks. It is an expansion of IDS. IDS only detects whereas IPS preserve the network from intrusion by dropping the packet, denying entry to the packet or blocking the connection. IPS and IDS together monitor the network traffic for abnormal activities and IPS is considered as just an explanation of IDS. The main difference is that the IPS are placed in-line to prevent intrusions and IPS can take decisions like dropping the packet, or resetting the connection along with sending alarms to the management console. An IPS can also detect/correct fragmented packets, Cyclic Redundancy Check (CRC) errors, or TCP sequencing issues.

Table summarizes the key differences between the IDS and IPS. Today, most of the network-based intrusion systems combine both detection and prevention – Intrusion Detection and Prevention Systems (IDPS).

Intrusion Detection System (IDS)	**Intrusion Prevention System (IPS)**
Passively monitors network behavior and detects attacks	Actively analyzes network behavior and “prevents” attacks in real-time
Supports both Network and Host level detection	Supports both network and host level detection
Passive monitoring, does not sit in the data path	Active monitoring, deployed in-line mode
Key measure is detection accuracy	Key measure is lesser number of false positives
NIDS: ISS, Cisco, Enterasys, Symantec	NIPS: McAfee Intrushield, NetScreen, Tippingpoint.
HIDS: ISS, Symantec, Enterasys	HIPS: Cisco, McAfee (Entercept). Snort – an open source Network IDS/IPS developed by Sourcefire

Table 6: Key differences between IDS and IPS

3.SIMPLE NETWORK MANAGEMENT PROTOCOL SNMP

3.1 Definition of SNMP

Simple Network Management Protocol (**SNMP**) is a protocol that follows Internet-standard. Its functionality is to collect and organize information from managed devices on IP networks and to modify that information in order to switch device functionality. Routers, switches, servers, workstations, printers, modem racks and more are devices that support SNMP.

SNMP is widely used in network management for network monitoring. SNMP exposes management data in the form of variables on the managed systems organized in a management information base which describe the system status and configuration. These variables can then be remotely queried (and, in some circumstances, manipulated) by managing applications.

There are multiple versions of the SNMP protocol. SNMPv1 is the original and initial version of the protocol, which is the most widely used version, but it has security problems. Its popularity largely stems from its ubiquity and long time in the wild. Unless you have a strong reason not to, we recommend you use SNMPv3, improve the performance, flexibility and security.

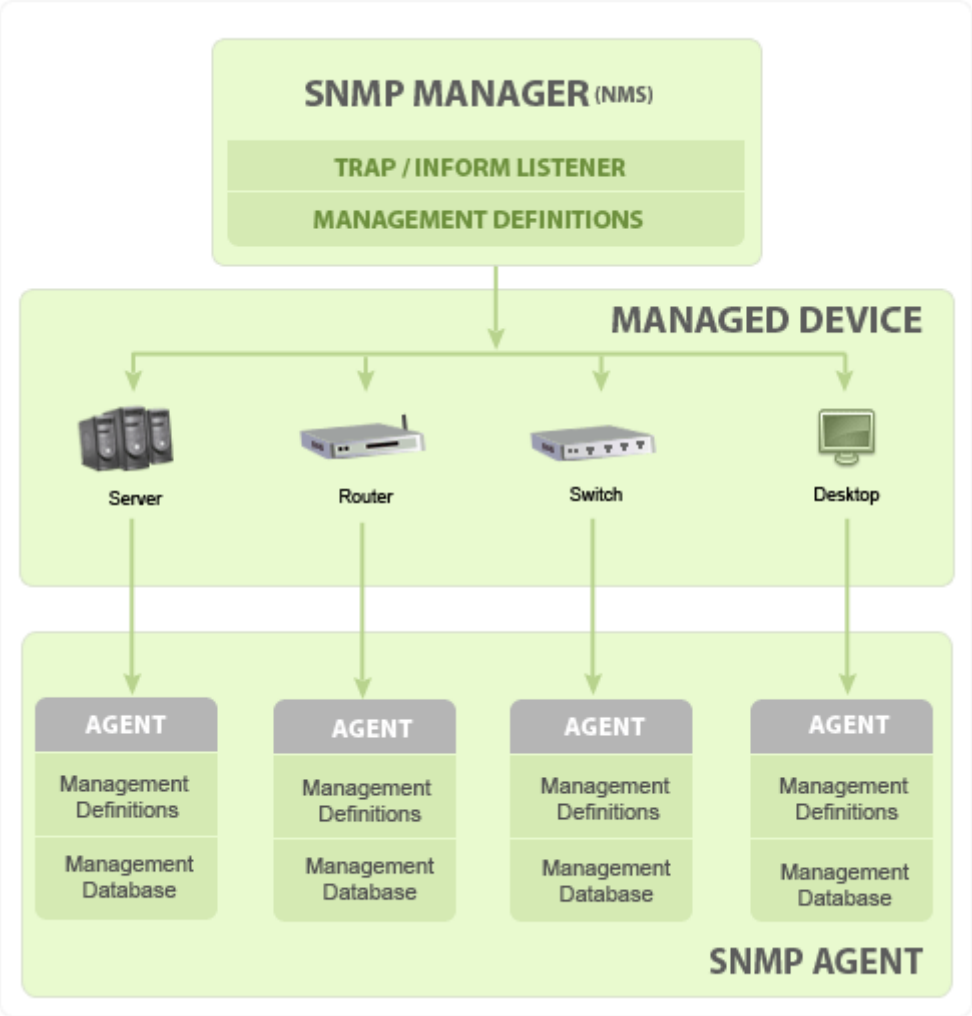

Figure 8: Basic SNMP Communication [8]

3.2 General Architecture

SNMP is the application layer protocol, which is part of the TCP / IP protocol stack and is designed for exchanging management information between devices connected to the network. The SNMP uses UDP, a transport protocol for exchanging information between devices. The UDP was chosen over TCP, because works better, especially in cases of dysfunction (e.g. congestion) network. The SNMP uses port 161 to send and receive messages, and port 162 to receive trap messages (trap messages). The protocol gathers information from a big variety of system in a consequent manner. It can also be used to connect a lot of different systems, the method of querying information and the paths to the relevant information are standardized. If we want to watch a network using SNMP, we have to employ network, devices that contains SNMP **agents**. An agent is a program that collects information about a hardware, organize it into predefined insertions, and reply to queries using the SNMP protocol. The component that makes questions to the agents for information is called an SNMP **manager**. These managers have data about all of the SNMP devices, that are enabled, in their network and can issue requests to collect information and set certain properties.

For a stand-alone management station, an application (process) management controls access to a MIB, located in the management station and provides an interface with the network administrator (human). The management application achieves network management using SNMP, which is implemented over the UDP protocol, IP and dependents from the network protocols (ethernet, ATM, FDDI, X25, etc.).

Each agent should implement the SNMP, UDP and IP. Furthermore, there is an application (process) agent that translates SNMP messages and controls the MIB agent.

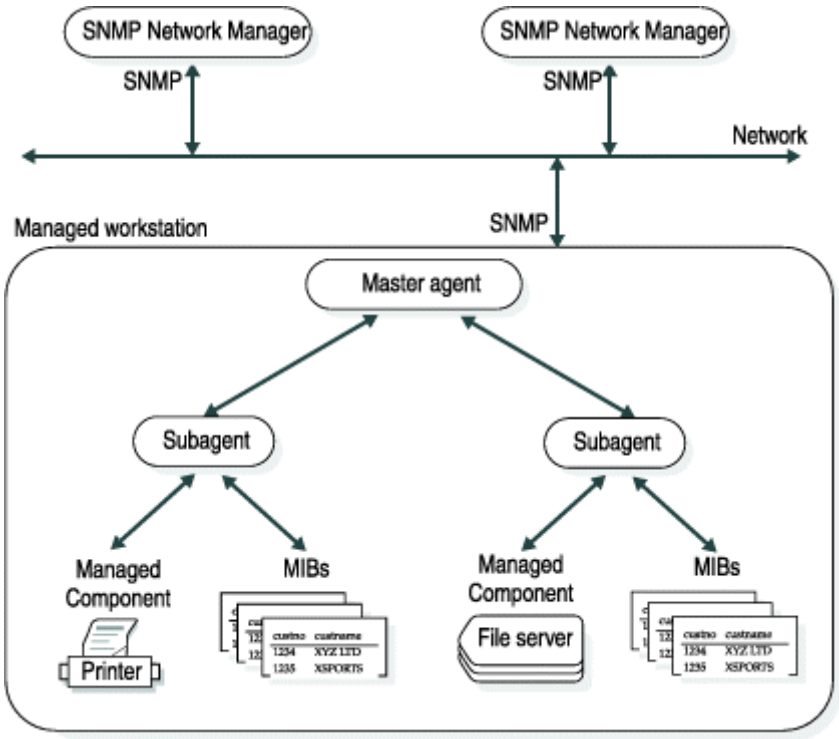

Figure 9: Architecture of SNMP [9]

3.2.1 SNMP Manager

The Network Management Station (NMS) or administrator (manager) provides all the necessary tools - applications for network management. The management component, when only discussing its core functionality, is actually a lot less complex than the client configuration, because the management component simply requests data. It can be performed on one or more servers. The ability to be distributed allows the system to be extended in parallel makes it more flexible (even if a host fails, the system will still work). Almost all of the commands defined in the SNMP protocol are designed to be sent by a manager component. These include GetRequest, GetNextRequest ,GetBulkRequest, SetRequest, InformRequest, and Response. The operator is responsible for making queries (polls) to the agent for an information, receiving notifications from the agent in cases of events, and then perform the appropriate action.

3.2.2 SNMP agent

An agent is a network-management software module that resides on a managed device. An SNMP agent is any computer or other network device that monitors and responds to queries from SNMP managers. SNMP agents do the bulk of the work. Their responsibility is to collecting information about the local system and save them in a database called the "management information base", or **MIB** and using a format that can be queried. The agent computer specifies which managers should have access to its information. It can also play the role of an intermediary to report information on devices that it can connect to but are not configured for SNMP traffic. And as a result, is more flexible in getting your components online and SNMP accessible. SNMP agents respond to most of the commands defined by the protocol. These include GetRequest,GetNextRequest, GetBulkRequest, SetRequest and InformRequest. In addition, an agent is designed to respond to the administrator's requests and inform the asynchronous for various events (trap - trap message).

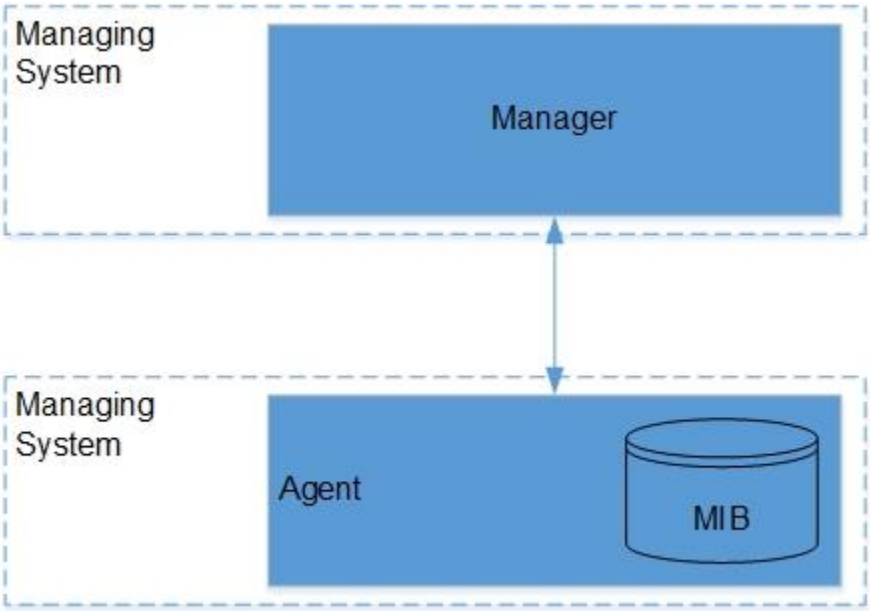

Figure 10: Communication Manager-Agent

3.2.3 Management Information Base

Management Information Base (MIB) is a collection of Information for managing network characteristics. The MIBs contains managed objects identified by the name Object Identifier (OID). Each of these Identifiers is unique and implies specific characteristics of a managed device. The return value of each identifier could be different e.g. Text, Number, Counter, etc. In short, MIB files are the set of questions that a Manager can ask the agent. Agent gathers these data locally and saves them, as defined in the MIB. So, the Manager should be aware of these standard and private questions for every type of agent. The MIB hierarchy can be represented in a hierarchical tree structure with individual variable identifier. The MIB of SNMPv1 characterized as MIB-STANDARD, the MIB of SNMPv2 as SNMPv2-mib. A typical object ID will be a dotted list of integers. For example, the OID in RFC1213 for “ipForwarding“ is 1.3.6.1.2.1.4.1.

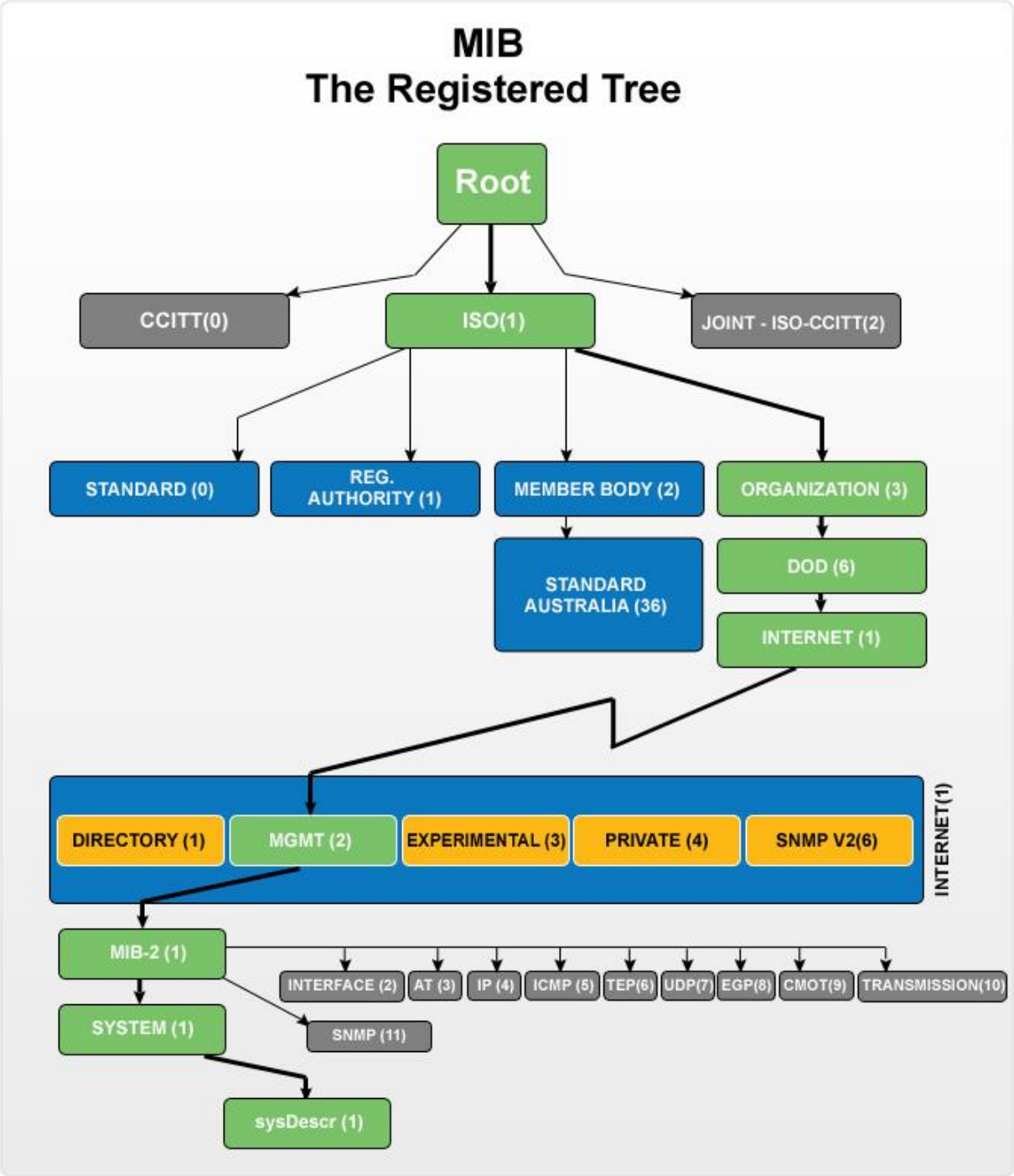

Figure 11: MIB Tree Diagram [10]

3.3 SNMP messages-types

One of the reasons that SNMP ıs wide known and it is used very much is the simplicity of the commands available. There are very few operations to implement or remember, but they are flexible enough to address the utility requirements of the protocol.

The following Protocol Data Units(PDU), or protocol data units, describe the exact messaging types that are allowed by the protocol:

- **Get**: A Get message is sent by a manager to an agent to request the value of a specific OID. This request is answered with a Response message that is sent back to the manager with the data.

- **GetNext**: A GetNext message allows a manager to request the next sequential object in the MIB. This is a way that you can traverse the structure of the MIB without worrying about what OIDs to query.
- **Set**: A Set message is sent by a manager to an agent in order to change the value held by a variable on the agent. This can be used to control configuration information or otherwise modify the state of remote hosts. This is the only write operation defined by the protocol.
- **GetBulk**: This manager to agent request functions as if multiple GetNext requests were made. The reply back to the manager will contain as much data as possible (within the constraints set by the request) as the packet allows.
- **Response**: This message, sent by an agent, is used to send any requested information back to the manager. It serves as both a transport for the data requested, as well as an acknowledgement of receipt of the request. If the requested data cannot be returned, the response contains error fields that can be set with further information. A response message must be returned for any of the above requests, as well as Inform messages.
- **Trap**: A trap message is generally sent by an agent to a manager. Traps are asynchronous notifications in that they are unsolicited by the manager receiving them. They are mainly used by agents to inform managers of events that are happening on their managed devices.
- **Inform**: To confirm the receipt of a trap, a manager sends an Inform message back to the agent. If the agent does not receive this message, it may continue to resend the trap message.

With these seven data unit types, SNMP is capable of querying for and sending information about your networked devices.

4. INTRUSION DETECTION BY USING SNMP ON IOT USING COMPLEX EVENT PROCESSING

4.1 Problem Definition

There are a few times, that significant information gets sent outside your network, without knowing it. Sometimes this can be happened by an unsuspecting employee clicking on a link in an email or entering on a malicious site while navigating on the internet. Unfortunately, there are a lot of harmful behaviors, that can provoke problems to your organization, like an unauthorized use of IT resources to advance racists, harassing or extremist issues.

How you can monitor these activities without violating the privacy of users? The best way deal with this problem is to use and anomaly detection software that flags abnormal user activity. In other words, this software is responsible for discovering the intrusions, the abnormal activity to your network and eliminate the possibility for attackers to harm your company. Its function is very simple. At first, you choose a data source that is like to provide the most useful information, such as the gateway logs for external traffic and access logs for internal resources. You could focus on URLs, users addresses, data volumes and HTML error codes from internal resources. The software looks across all users and then building, in order to create a baseline profile for the collective behaviors of the population.

As it relates to security, the majority of the employees and resources follow the rules and maintain a predictable behavior. On the other hand, infected users are going to behave differently. By comparing the behaviors of each individual against the baseline profile, abnormal activities can be identified. These abnormal activities would be rare behaviors, behaviors that are outliers to the normal population or behaviors that suddenly change.

While there are a thousand of anomalies events in a normal working day, such activities that are not necessarily defined as problems. This is happened because software aims to discover behaviors that could potentially provoke damage to your organization security. It is implemented to display you an accurate picture of the activity in your organization, concentrating more on events or likelihood of an event that could harm your company, rather than on minor activities that is ultimately harmless.

For example, a single day spike of 300 outbound emails from a single employee would not necessarily alarm you about the event, unless the email account belonged to an employee that was fires the prior days. Or if an address that in a normal condition sends and receives 300 mails per day suddenly spikes 30,000 outbound emails, then there is an obvious problem. Discovery anomalies is about cross-correlating multiple types of activities and weighting them by how rare they are or by how many other anomalies they generate. It is not just pointing out activities that are different.

For the most part, rogue users in an organization are in the minority, they stand out because their behavior is different that the most users. Technology in nowadays can expose these individuals by building a joint probabilistic model(baseline profile) and then individually comparing each member of the population against the whole, making it easier for your IT team to discover any individuals who are looking to harm your organization.

The rise of the IoT requires that we consider several fundamental issues pertaining to performance and management:

- **Security:** Concerns have been raised that the IoT is being developed rapidly without consideration of the profound security challenges involved and the

regulatory changes that might be necessary. Security is a big vulnerability in all devices, which are connected to the network. This is a major problem, because we don't have manage to find solution for existed network problems, think how easy is to hack basic devices, as routers, network storage and smart TVs. Now, you can imagine what will be happen if you have billions of devices, which are easily to hack. Network Security problems are getting bigger with the evolution of IoT and smart devices, in order to deal with them, we must invent and discover solutions that have to be smart, efficient and effectual.

- **Privacy:** Smart devices rule our lives today. Whatever you need to do, there is a smart device, that can cover your needs. This thing leads to a big variety of datasets, such as health, gps traces, car instructions, etc., some of them are private and sufficient. So we must keep them safe and private.
- **Data Storage:** Another big problem has to do with the storage space. All smart devices are keeping logs from their activities, some of them are important and some other not, so in the near future the demand for bigger repositories may arise, in order to store and processing the dataset.
- **Energy:** A concern regarding IoT technologies pertains to the environment impacts of the manufacture, use, and eventual disposal of all smart devices. Modern electronics are replete with a wide variety of heavy metals and toxic synthetic chemicals, which make recycling very difficult. At last, the need for energy will be increase dramatically.

In this thesis we are trying to apply a Complex event processing method in order to detect events (intrusions) of any mobile device is compatible with SNMP protocol participating in an IoT network

4.2 Challenges

The evolution of IoT constitutes a big and very interesting subject in the scientific community, because it has open a whole new world of challenges. In this thesis, we try to deal with some challenges concerning with security issues. The first challenge is heterogeneity, i.e. to face the highly dynamic structure of IoT network, incoming horde of connected IoT devices become similarly endpoints. Every day a new smart device is created, which must be secure connected to the network and more specific with the existing devices, so the device-center solutions for incorporating security components cannot integrate possible future changes. Our framework monitors any device can be registered via snmp.

The second challenge is connected with dynamic environment of IoT because suspicious anomalies of device performance need to be discovered quickly in order to prevent problems in real time. In our framework we created some event listeners, which are linked with metrics devices, in order to detect any abnormal activity. Listeners also learn dynamically the normality of each individual device and play and important role in intrusion detection. Smart devices also should not count only as unique units as they can join and leave networks stochastically. Moreover, using their spatial position, we can group them into smaller sub-nets. A possible intrusion to smartphones will affect the whole subnet, so it is important to group the devices into teams and the training phase handle the data from each group to a cooperative way.

Figure 12: Internet Of Things [11]

4.3 Related work

4.3.1 IoT Security Intrusions

Researchers especially in the past decade have applied techniques of Machine Learning(ML) in order to take the advantage of keeping the attack knowledge databases up-to date and improve the detection rate. Zhenghong Xiao proposed a Machine Learning (ML) based anomaly detection scheme , where Bayesian classification algorithm is used to detect anomalous nodes in Wireless Sensor Networks (WSN) [12]. An alternative approach that uses fuzzy logic is proposed in [13]. Jonatan Gomez and Dipankar Dasgupta proposed a technique to generate fuzzy rules that are able to detect anomalies and some specific intrusions, in order to predict the normal and the abnormal behaviors in networks. Moreover, decision trees classifiers have been proposed by [14]. Decision boosted tree classifiers [15],[16],[17] are trained with different sample sets drawn from the original training set in order to achieve an Intrusion Detection System. All hypotheses, produced from each of these classifiers, are combined to create a new tree and calculate total learning error, thereby arriving at a final composite hypothesis.

Another technique, that has been proposed is Markov Model classifiers. They proposed a simple data preprocessing approach to speed up a hidden Markov model(HMM) training for system-call-based anomaly intrusion detection [18]. Jiong Zhang and Mohammad Zulkernine [19] implemented a framework of anomaly based network intrusion detection, where patterns of network services are built by random forests algorithm over traffic data. Intrusions are detected by determining outliers related to the build patterns.

4.3.2 Knowledge Discovery in heterogeneous wireless networks

The most known knowledge discovery and data mining toolboxes nowadays are Weka, Rapidminer and IBM-SPSS , but all these tools are not designed for IoT cases. These tools are not distributed so you have to load all the data to a single machine and they use a very similar data analysis scenario. According to this scenario, at first they import data, the most of the time form static data sources, defined by the data analyst. Then, provide the analyst with the means to design Data Mining workflows, composed of data mining operators, dimensionality reduction, classification model building, instance

selection and sampling operators. At last, apply these workflows to achieve the knowledge discovery. A toolbox that somehow deviates in terms of the nature of the data that is able to handle from the existing mining toolboxes is MOA [20]. Massive Online Analysis(MOA) is a software environment for implementing algorithms and running experiments for online learning from evolving data streams. It also contains collection of offline and online for both classification and clustering as well as tools for evaluation. However, MOA is better a benchmarking tool for testing rather than a tool for developing and deploying models for IoT.

CEP context detection capability is among the key characteristics of IoT middleware [21]. The CEP architecture is far more versatile than existing IoT platforms [22] as it allows full customization and support of the application domain requirements. Existing IoT platforms impose a certain(static) processing in the collected data on their route from the data collection layer to the application or focus on different aspects of IoT(e.g., network support [21]).

Most of the Data Mining and Machine Learning algorithms assume that the data are identically independently distributed. On the other hand, IoT the data generation process can have a strong spatio-temporal dimension, which needs to be taken into account during the data modeling process if one wants to perform reliable knowledge extraction.

In addition, extensive research has been performed over the last years on learning over multi-source and heterogeneous data sources, that led to the development of numerous methods on kernel learning, [23], [24]. Recently, [25], such methods have been adapted for the on-line learning scenario, which is of direct use in the multiple data streams produced in

sensor networks. In addition, in a real world IoT scenario, nodes can regularly fail, e.g. limited battery life-time, resulting in incomplete data availability. Any knowledge discovery algorithm that is going to be used in such an environment for model building as well as its models should be able to work with incomplete and missing information. Most of the Data Mining and Machine Learning algorithms assume that the data will be collected in some repository and then analyzed offline. The algorithms are not able to detect, manage, and accommodate concept drift, i.e. changes in the distributions of the learning data, in other words they are meant for batch processing of static data.

4.4 What is Complex Event Processing – ESPER

Event Processing is a method of tracking, analyzing and processing streams, in order to discover events, that are important to our system. Complex Event Processing or CEP is the use of technology to predict high-level events and patterns from different data sources. The goal of complex event processing is to identify meaningful events and respond them as quick as possible. Esper is an open-source java-based framework for Complex Event Processing(CEP) and Event Stream Processing, that processes series of data to detect significant events and deal with them. Esper provides a rich Event Processing Language(EPL) to express filtering, aggregation and joins, possibly over sliding windows of multiple event series. It also includes pattern semantics to express complex temporal causality among events.

4.5 Proposed Methods

Our goal was to implement a framework for event detection in real time input streams. In more details, we track information from smart devices and feed them to our

framework, in order to detect critical events and inform the smart devices for the possibility of an intrusion. Our framework consists of three parts:

- SNMP
- Esper Engine
- Rules: feature extraction

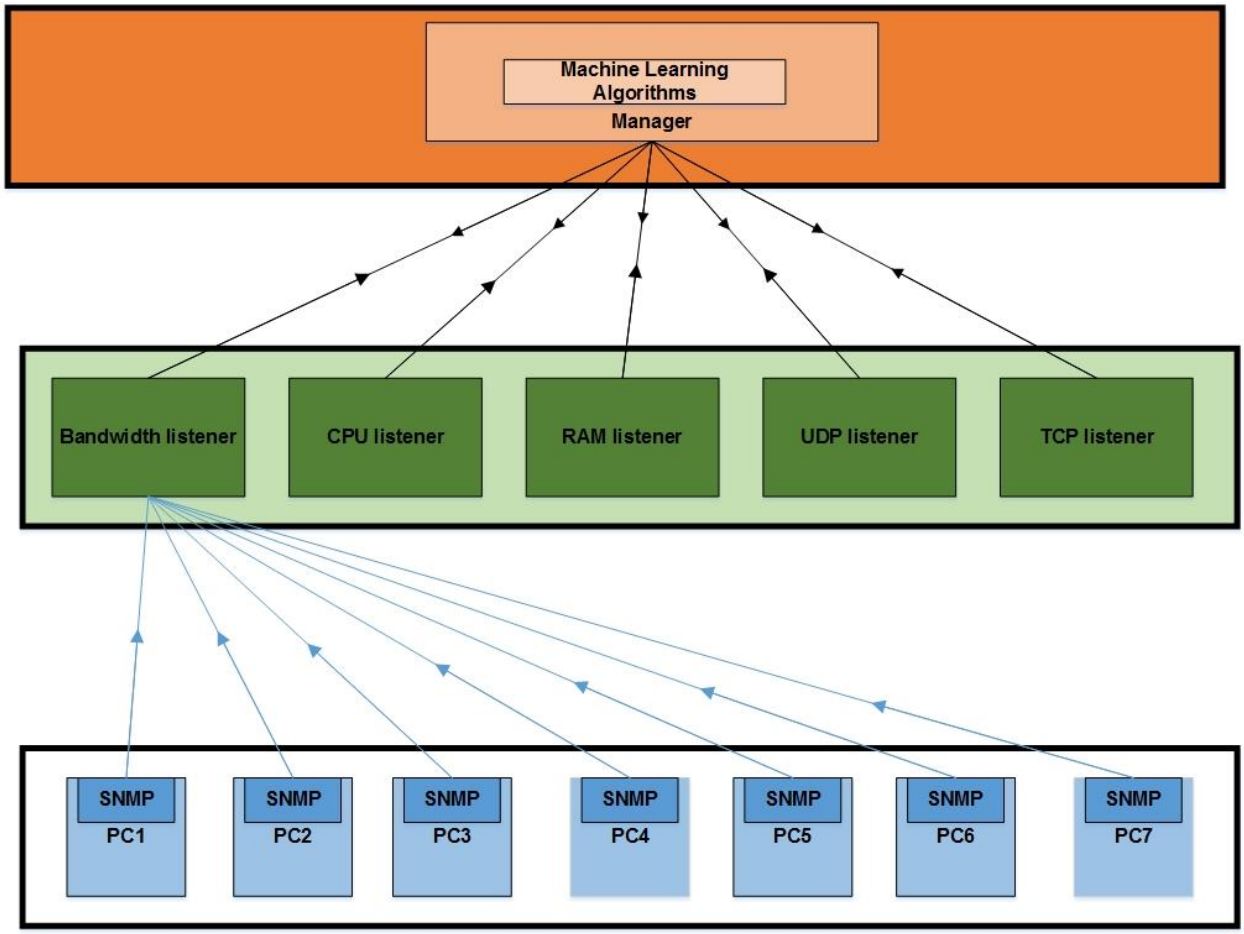

Figure 13: CEP Architecture

4.5.1 SNMP

To address the heterogeneity problem/issue, our framework uses a SNMP standard, which it can offer a mobile platform to monitor SNMP devices participating in a network. This protocol makes our life easier because, protocol provides transparency according to the different characteristics of the devices, so it is very flexible and portable.

The measurements and their OIDs, that we track from SNMP are:

- Bandwidth (OID:1.3.6.1.2.1.2.2.1.10.2 & 1.3.6.1.2.1.2.2.1.16.2): Bandwidth of the network.
- TCP (OID:1.3.6.1.2.1.6.5.0): The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.
- UDP (OID:1.3.6.1.2.1.7.2.0): The total number if received UDP datagrams for which there was no application at the destination port.
- CPU (OID:1.3.6.1.4.1.2021.11.11.0): percentages of idle CPU time.
- RAM (OID:1.3.6.1.4.1.2021.4.6.0): total RAM used.

Measurements are collected by SNMP, about every 2 seconds, as data streams and are sent to Esper Engine.

4.5.2 Esper Engine

Esper is a simple, efficient and effective framework, with which you can implement event listeners very easy and with little effort. An Esper program consists from two important components, the handler and the listener. The handler is a singleton class and is responsible for all the functionality of the program. It the first class that it is running and creates the environment of the program. It initializes the listeners and all the other things that are related to them, such as thresholds. When the program running, handler has the role of the “mailman”, it receives the records of input data and forwards them to the corresponding listener, to be analyzed. The listener is the component that receives the records and analyze them to detect any abnormal activities. This processing happens by using the EPL, all records are filtering with the EPL query and decides if we have an event or not. EPL Queries, which look like SQL Queries, are a very useful tool to find common patters in dataset and they can be as complicated as you want. The listener inherits two function from the Esper, the “getStatement” and the “update”. The “getStatement” initializes the query and the “update” catches the event and you can add code inside her to deploy anything you want.

We implemented two layers of events, one layer for the event detection and the second to inform and parameterized the measurements of the devices. The first layer, called SNMPManager, searching for events, in our stream data, corresponding to the OIDs of SNMP. In other works, we try to discover events. Some of these listeners are parameterized dynamically using thresholds, or they are static When an event occurs, these listeners send their event to the second layer. To implement the second layer, we used Event Processing Language (EPL).

4.5.2.1 First layer of the Esper Engine

The first layer of listeners is responsible for analyzing and mining the information from smart devices, to check if the smart devices working properly and detect any possible intrusion against them. Some of these listeners are parameterized with dynamic thresholds and some other are static, this separation is based on the data, that listeners analyze. The parameterization is a very vital concept in our project, because you can learn the normality of an attribute and change the threshold depending on the distribution of data. Moreover, makes CEP more flexible, because some events are elastic. For example, if the memory of a smart device is decreasing, may be not an intrusion, but related with user needs, because running a lot of applications at the same time.

There are many distinct patterns, that one can observe, in order to detect smart devices intrusions. In this thesis, we decide to discover events, which are related with the Bandwidth, TCP, UDP, CPU and RAM.

1. Event Listener for Bandwidths

This listener is responsible to manage the bandwidth of the devices. Bandwidth is a very important factor, because in an occasion of intrusion, the value of bandwidth increases or decreases abnormally. The query for the bandwidth is:

"select * from SnmpInfo where bandwidth > BANDWIDTH_THRESHOLD"

Using this query, we take all the records, where bandwidth is bigger than the value of the threshold. When the program begins, initializes the numeric value of BANDWIDTH_THRESHOLD and stays until the end of execution. When a bandwidth event occurs, listener sends the event to the second layer of the Esper.

2. Event Listener for TCP connections

This listener is used to discovery changes in the TCP connections of the device. In more detail, we track the number of times, that TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state, and we compare them with a threshold, in order to detect an intrusion in the device. If the efforts are greater than the threshold, we have an intrusion. The query for the TCP is:

"select *,avg(tcpInErrs) from SnmpInfo(tcpInErrs >0) having tcpInErrs > TCP_THRESHOLD"

With this query, we take the average of the TCP efforts and we compare them with the threshold. The average is calculated from the beginning of the program and no from the last event. When the program begins, initializes the numeric value of TCP_THRESHOLD. When a TCP event occurs, listener sends the event to the second layer of the Esper. This listener is not static, the TCP threshold is changing dynamically from the Esper Manager.

3. Event Listener for UDP

This listener is responsible for detecting weird behavior of the UDP connection of the devices. It trucks the total number of received UDP datagrams for which there was no application at the destination port. If the total number of the UDP datagrams is greater than the threshold, then there is a possibility of intrusion. The query for this listener is:

"select *,avg(udpFailures) from SnmpInfo(udpFailures >0) having udpFailures > UDP_THRESHOLD"

With this query, we take the average of the UDP efforts and we compare them with the threshold. The average is calculated from the beginning of the program and no from the last event. When the program begins, initializes the numeric value of UDP_THRESHOLD. When a UDP event occurs, listener sends the event to the second layer of the Esper. This listener is not static, the UDP threshold is changing dynamically from the Esper Manager.

4. Event Listener for CPU

With this listener, we try to discovery abnormally behavior to the CPU of the device. In more detail, we watch the percentage time of CPU being idle, time that CPU is not in used, and we compare with a threshold, aiming to detect an intrusion. If this percentage is lower than our threshold, then there is a possibility of an intrusion to the smart device. The query for the CPU event is:

"select * from SnmpInfo where cpuIdleTime < CPU_THRESHOLD"

Using this query, we truck all the percentage time of CPU being idle that are lower from our threshold. This listener is static, so the threshold never changes. When the program

begins, initializes the numeric value of the CPU_THRESHOLD. When a CPU event occurs, listener sends the event to the second layer of the Esper.

5. Event Listener for RAM

This listener is responsible for detect RAM changes in the devices. We truck the total RAM memory that the device uses and we compare it with a threshold in order to discover an attack. If the total RAM memory id greater than the threshold, there is a possibility to of an intrusion to the smart device. The query for the RAM event is:

"select *,avg(totalUsedMemory) from SnmpInfo(totalUsedMemory >0) having totalUsedMemory > MEMORY_THRESHOLD"

With this query, we take the average of the used RAM memory and we compare them with the threshold. The average is calculated from the beginning of the program and no from the last event. When the program begins, initializes the numeric value of MEMORY_THRESHOLD. When a RAM event occurs, listener sends the event to the second layer of the Esper. This listener is not static, the MEMORY_THRESHOLD is changing dynamically from the Esper Manager.

4.5.2.2 Second layer of the Esper Engine - Network Manager

The second Layer called NetworkManager, and it is responsible for the proper function of the smart devices. Network Manager is a vital component, because it has under his supervision all the event listeners/components and communicate with them. All the events of the first layer are sent to the Network Manager, in order to get processed. At first Manager, must identify the event, and keeps a log with counters for all different events that we described above. Then, if the event is not static, manager should inform the first layer listeners and update their threshold.

The dynamics thresholds are very significant and indispensably to detect an intrusion and we evolve them with the progress of our framework. In the first version, we called it algorithm 1, we use dynamic thresholds, and the thresholds changing with static way. More specific, we increase the thresholds with a static number. In our second version, we implemented Shewart Algorithm, which calculates the thresholds using the current and the previous values of the variable. And in our last version, we deploy ART, which is a vector algorithm and uses the current values of all the dynamic variables and calculates the new thresholds. More details, about the two last algorithms you will read it below.

4.5.3 Rules: feature extraction

There are many anomaly detection algorithms proposed in the literature that differ according to the information used for analysis and according to techniques that are employed to detect deviations from normal behavior. In this section, we provide classification of anomaly detection techniques based on employed techniques into the following five groups:

- ***Statistical Methods***. Statistical methods monitor the user or system behavior by measuring certain variables over time (e.g. login and logout time of each session in intrusion detection domain). The basic models keep averages of these variables and detect whether thresholds are exceeded based on the standard deviation of the variable. More advanced statistical models also compare profiles of long-term and short-term user activities.

- ***Distance based Methods***. Distance based approaches attempt to overcome limitations of statistical outlier detection approaches and they detect outliers by computing distances among points. Several distances based outlier detection algorithms have been recently proposed for detecting anomalies in network traffic. These techniques are based on computing the full dimensional distances of points from one another using all the available features, and on computing the densities of local neighborhoods.
- ***Rule based systems***. Rule based systems used in anomaly detection characterize normal behavior of users, networks and/or computer systems by a set of rules.
- ***Profiling Methods***. In profiling methods, profiles of normal behavior are built for different types of network traffic, users, programs etc., and deviations from them are considered as intrusions. Profiling methods vary greatly ranging from different data mining techniques to various heuristic-based approaches. In this section, we provide an overview of several distinguished profiling methods for anomaly detection.
- ***Model based approaches***. Many researchers have used different types of models to characterize the normal behavior of the monitored system. In the model-based approaches, anomalies are detected as deviations for the model that represents the normal behavior. Very often, researchers have used data mining based predictive models such as replicator neural networks or unsupervised support vector machines.

Although anomaly detection algorithms are quite diverse in nature, and thus may fit into more than one proposed category, this classification attempts to find the most suitable category for all described anomaly detection algorithms.

Network manager encapsulates the algorithms that have been developed in order to dynamically change the thresholds of an event listener. For the master thesis purposes two algorithms are being presented: Shewhart [5] and Adaptive Vector Quantization (AVQ) [4].

4.5.3.1 Shewhart Controller

In the Shewhart control chart, a variable x_t is detected to deviate at time k from its normality denoted by two control limits: the Upper Control Limit (UCL) and Lower Control Limit (LCL). The control limits are defined as the distance from the current mean value of the process $x_1, \dots, x_k$ which is:

- $\bar{x}_k + \alpha\sigma_k$ for UCL, and
- $\bar{x}_k - \alpha\sigma_k$ for LCL

That is, x_k is detected to fire an alarm if $x_k > UCL_k$ or $x_k < LCL_k$. The UCL_k and LCL_k values are defined as follows (through incremental methods):

$$\bar{x}_k = \bar{x}_{k-1}\frac{x_k - \bar{x}_{k-1}}{k}$$

and

$$\sigma_k^2 = \frac{1}{k}\left((k-1)\sigma_{k-1}^2 + (x_k - \bar{x}_k)(x_k - \bar{x}_{k-1})\right)$$

The controller returns $+1$ if $x_k > UCL_k$, -1 if $x_k < LCL_k$ and normality, i.e., 0 if $x_k \epsilon (LCL, UCL)$. The parameter a is usually defined to be $a = 3$.

4.5.3.2 Adaptive Reasonance Theory

In ART given an input all of the output units calculate their values and the one similar to the input is chosen. Let us assume that we use the Euclidean distance. If the minimum value is smaller that a certain threshold value, named the vigilance, the update is done as online k-means. If this distance is larger than vigilance, a new output unit is added and its center is initialized with the instance. This defines a hypersphere whose radius is given by the vigilance defining the volume of scope of each unit; we add a new unit whenever we have an input that is not covered by any unit. Denoting vigilance by ρ we use the following equations at each update

$$b_i = \left\| m_i - x^t \right\| = \min_{l<k} \| m_l - x^t \|$$

$$\begin{cases} m_{k+1} \leftarrow x^t, & if\ b_i > \rho \\ \Delta m_i = \eta(x - m_i), & otherwise \end{cases}$$

Putting a threshold on distance is equivalent to putting a threshold on the reconstruction error per instance, and if the distance is Euclidean and the error is defined as in the following equation this indicates that the maximum reconstruction error allowed per instance is the square of vigilance.

$$E^t(\{m_i\}_{i=1}^k | x^t) = \frac{1}{2}\sum_i b_i^t \left\| x^t - m_i \right\|^2 = \frac{1}{2}\sum_i \sum_{j=1}^{d} b_i^t (x_j^t - m_{ij})^2$$

Where

$$b_i^t \begin{cases} 1, & if\ \left\| x^t - m_i \right\| = \min_l \| x^t - m_i \| \\ 0, & otherwise \end{cases}$$

X={x^t}$_t$ is the sample and m_i, i=1,…,k are the cluster centers. b_i^t is 1 if m_i is the closest center to x^t in Euclidean distance. It is as if all m_i compete and the closest wins.

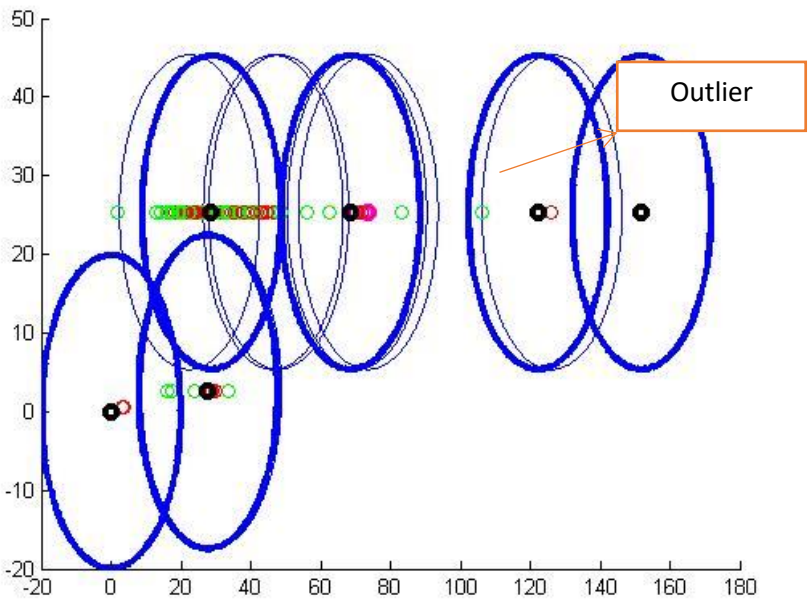

Figure 14 – ART Clustering in two dimensions

4.6 Experimental Results

4.6.1 Simulation Environment

The dataset, that we used to take the experiments is “UNB ISCX Intrusion Detection Evaluation DataSet” [26],**Error! Reference source not found.**,[27]. This dataset contains real traces, which are analyzed to create profiles for agents that generate real traffic for HTTP, SMTP, SSH, IMAP, POP3 and FTP. We select this dataset because according to the authors has the following characteristics:

- **Realistic network and traffic:** Ideally, a dataset should not exhibit any unintended properties, both network and traffic wise. This is to provide a clearer picture of the real effects of attacks over the network and the corresponding responses of workstations. For this reason, it is necessary for the traffic to look and behave as realistically as possible. This includes both normal and anomalous traffic. Any artificial post-capture trace insertion will negatively affect the raw data and introduce possible inconsistencies in the final dataset. Consequently, all such adjustments are highly discouraged.
- **Labeled dataset:** A labeled dataset is of immense importance in the evaluation of various detection mechanisms. Hence, creating a dataset in a controlled and deterministic environment allows for the distinction of anomalous activity from normal traffic; therefore, eliminating the impractical process of manual labeling.
- **Total interaction capture:** The amount of information available to detection mechanisms are of vital importance as this provides the means to detect anomalous behavior. In other words, this information is essential for post-evaluation and the correct interpretation of the results. Thus, it is deemed a major requirement for a dataset to include all network interactions, either within or between internal LANs.
- **Complete capture:** Privacy concerns related to sharing real network traces has been one of the major obstacles for network security researchers as data providers are often reluctant to share such information. Consequently, most such traces are either used internally, which limits other researchers from accurately evaluating and comparing their systems, or are heavily anonymized with the payload entirely removed resulting in decreased utility to researchers. In this work, the foremost objective is to generate network traces in a controlled testbed environment, thus completely removing the need for any sanitization and thereby preserving the naturalness of the resulting dataset.
- **Diverse intrusion scenarios:** Attacks have increased in frequency, size, variety, and complexity in recent years. The scope of threats has also changed into more complex schemes, including service and application-targeted attacks. Such attacks can cause far more serious disruptions than traditional brute force attempts and also require a more in-depth insight into IP services and applications for their detection. Through executing attack scenarios and applying abnormal behavior, the aim of this objective is to perform a diverse set of multistage attacks; each carefully crafted and aimed towards recent trends in security threats. This objective often labels many of the available datasets as ineffective and unfit for evaluating research results.

We used six days of measurements from the above dataset. In more detail:

- 12/6/2010 Normal Activity. No malicious Activity
- 14/6/2010 HTTP Denial of Service + Normal Activity

- 15/6/2010 Distributed Denial of Service using an IRC Botnet
- 16/6/2010 Normal Activity. No malicious Activity
- 17/6/2010 Brute Force SSH + Normal Activity

All the files are pcaps, so we have to use a replay framework in order to simulate them, such as tcp replay. The experiments were simulated in VMs and one physical computer with the following specifications:

- VMs: cpu: dual core 2.6GHz, memory:4gb and hard disk: 60gb.
- Physical Computer: cpu: Inter(R) Core(TM) i5-4300U CPU @1.90GHz 2.50Ghz, memory 8gb and hard disk: 80gb.

In the VMs we install the SNMP protocol and tcp reply, in order to simulate the dataset. We use this VMs as smart devices. In the physical machine we install Esper and constitutes the main part of our framework, which take the records of the dataset and analyze them, in order to detect an event.

4.6.2 Evaluation Experiments

In our experiments, we try to detect events concerning to tcp,udp, bandwidth, memory and cpu. In order to find the best way to discover an intrusion, we implement three algorithms with different philosophy and compare them. The first algorithm uses only dynamic thresholds, the changes in thresholds are with a static number. The second algorithm Shewart has dynamic thresholds with not static number, takes the current value of the event and calculates a new threshold for the variable, using the previous values of the variable. The third algorithm ART has dynamic thresholds such as Shewart, but calculates the new threshold with a vector way. Using all the current values(in our situation the three values: tcp, udp, memory) of the event to calculate the new threshold. We run our three algorithms for the five dataset, that we analyze above. The first and the second algorithm run for one time for each dataset, but the ART, because it is a vector algorithm and takes two arguments, we executed more times. More specific, the two arguments are: r (radius of the circle) and η(defines the velocity of the centroid updates), and the executions for each dataset are :

- $r = 50, \eta = 0.5$
- $r = 50, \eta = 1$
- $r = 100, \eta = 0.5$
- $r = 100, \eta = 1$

In the ART algorithm, we make an admission, that when an event occurred, we counted as an event for all the dynamic variables.

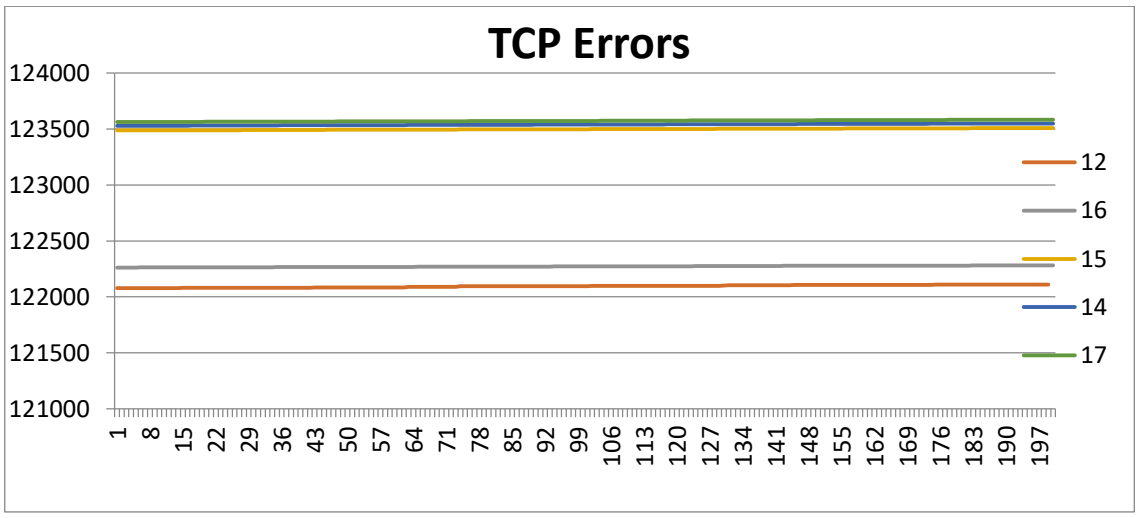

Figure 15 : All values of TCP in each dataset

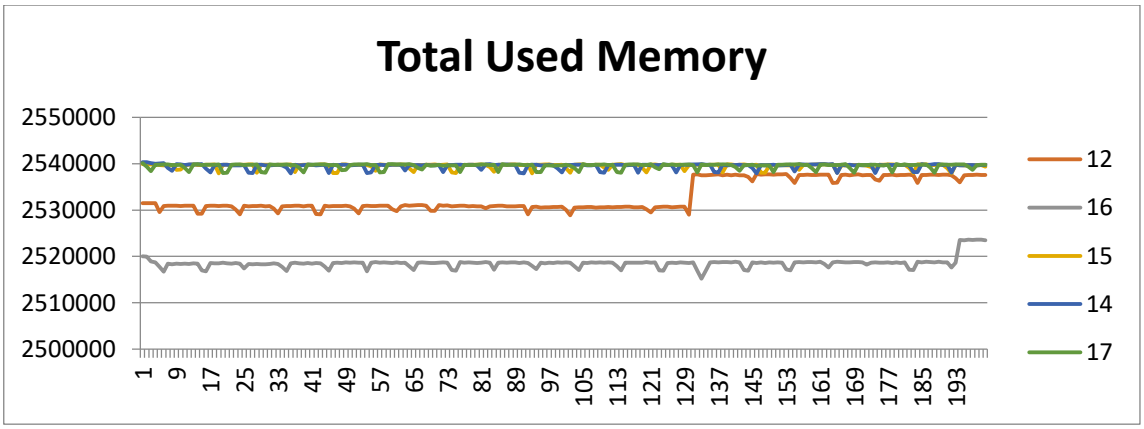

Figure 16 : All values of Memory in each dataset

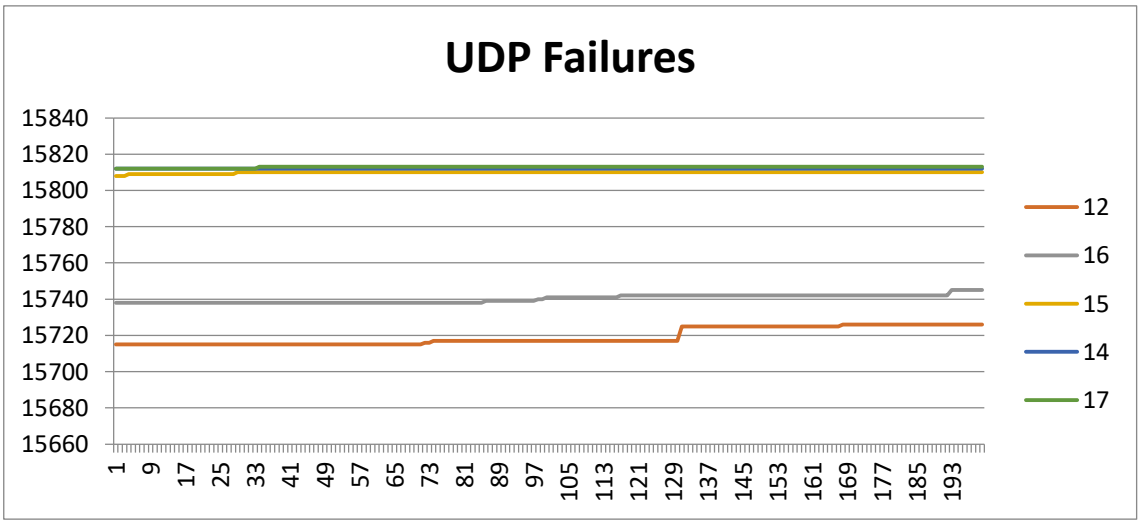

Figure 17 : All values of UDP in each dataset

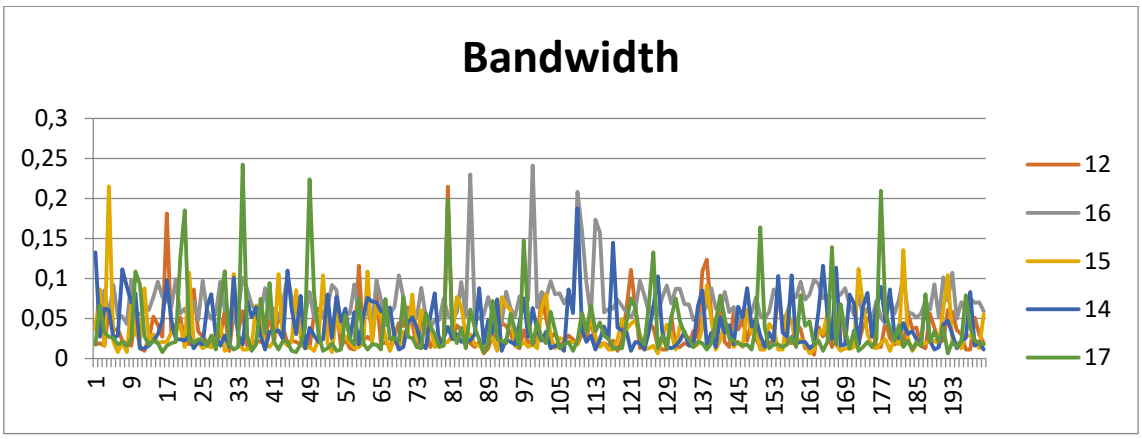

Figure 18 : All values of Bandwidth in each dataset

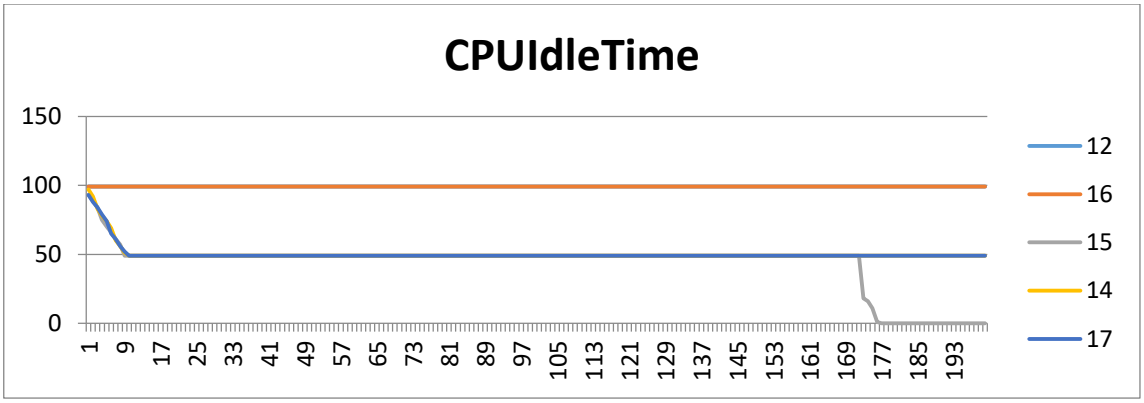

Figure 19 : All values of CPU in each dataset

In the diagrams above, you can see all the values of the variables, which exist in each the dataset. In the table below, you can see all the event counters for dynamic variables from our algorithms for each dataset. Something that we can observe is that when you increase the radius and η in the ART, then you have fewer events. So the values of these arguments, depends on how strict you want your framework. Furthermore, you can see that all the algorithms, the attacking days, detect more events than the normal days, so have the proper functionality. Also you can observe that the normal days, all the algorithms have similar results. Moreover, the Shewart algorithm detects an event, and if the next values, are not very different, then loses all the next events.

Day	Algo	TCP Failures	UDP Failures	MemoryAllocated
12	**Algorithm 1**	9	2	2
	Shewhart	10	2	2
	ART 50 0.5	14	14	14
	ART 50 1	9	9	9
	ART 100 0.5	12	12	12
	ART 100 1	6	6	6
16	**Algorithm 1**	8	1	2
	Shewhart	5	7	2
	ART 50 0.5	16	16	16

	ART 50 1	12	12	12
	ART 100 0.5	13	13	13
	ART 100 1	11	11	11
14	**Algorithm 1**	67	26	1
	Shewhart	57	1	1
	ART 50 0.5	148	148	148
	ART 50 1	151	151	151
	ART 100 0.5	90	90	90
	ART 100 1	86	86	86
15	**Algorithm 1**	58	2	2
	Shewhart	57	2	2
	ART 50 0.5	128	128	128
	ART 50 1	132	132	132
	ART 100 0.5	86	86	86
	ART 100 1	56	56	56
17	**Algorithm 1**	138	71	71
	Shewhart	49	2	2
	ART 50 0.5	140	140	140
	ART 50 1	148	148	148
	ART 100 0.5	98	98	98
	ART 100 1	85	85	85

Table 7 : All the metrics of the dynamic variables for each dataset

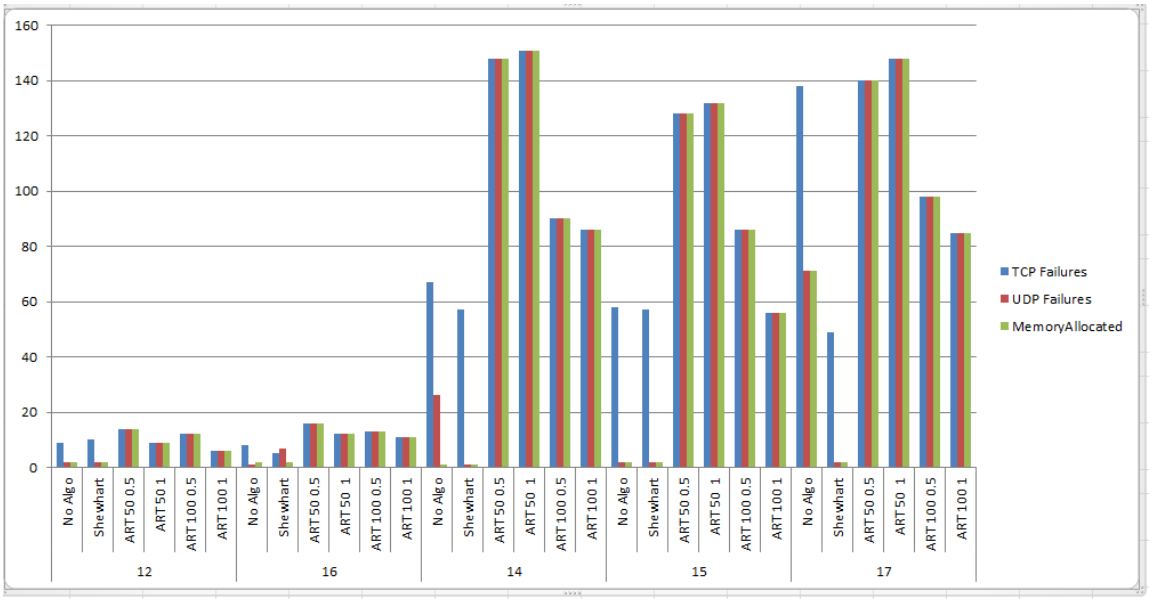

Figure 20 : All the metrics of the dynamic variables for each dataset

Figure 20 shows an overview of the events measures during the days of the experiments. Ii is higlhited that the days with normal user network activity the sum of the events triggered are much fewer than the attacking days for all the experiment. In general we can notice that ART algorithm generates 10 times more events in attack mode than in normal activity.

Analyzing an attacking day, in the figures 21 and 22, we can observe that if the radius is small and increase the η, we have very deviations in the two algorithms. The same does not occur when we have a bigger radius and increase the η.

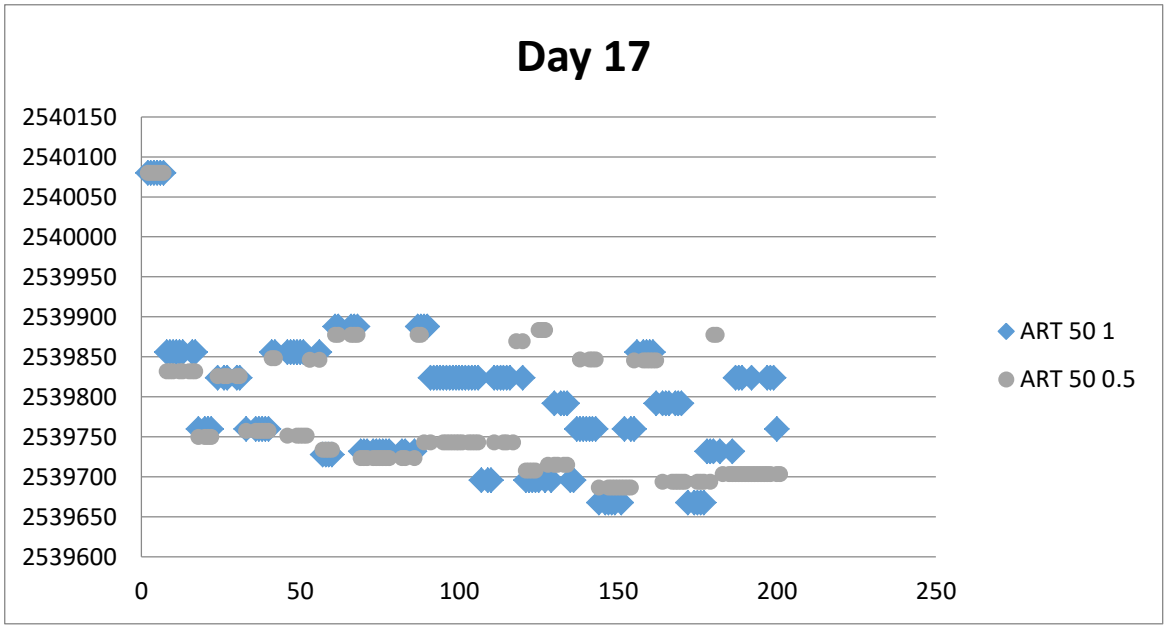

Figure 21 : Comparison between ART radius=50, η=1 and ART radius:50, η=0.5 for day 17

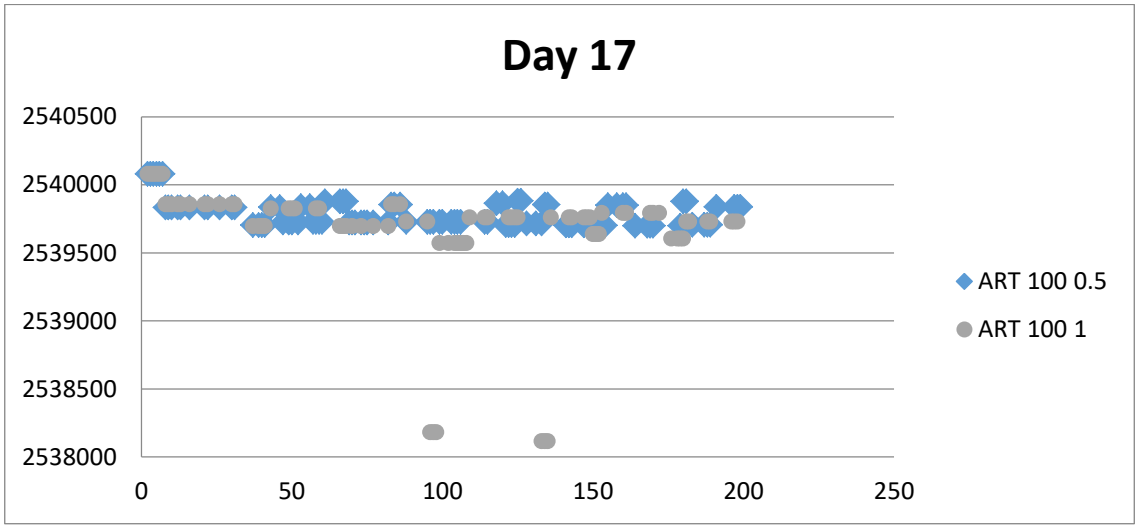

Figure 22 : Comparison between ART radius=100, η=1 and ART radius:100 , η=0.5 for day 17

In the figure below, you can notice that, all the ART algorithms have almost the same behavior in the normal days. On the other hand, the attacking days, you observe that the ART algorith with the less radius, have more events than the other ART algorithms. Finally, we can see that if you increase the radius and the η , you have less events.

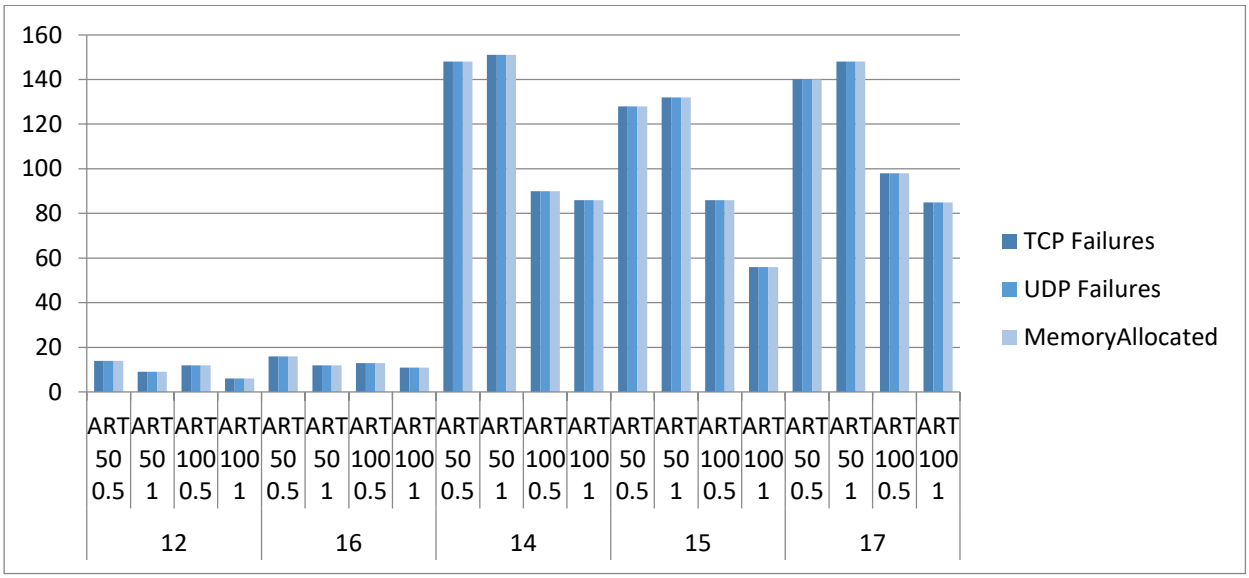

Figure 23 : event counters for dynamic variables of all ART algorithms for each day

Comparing the performance of the three different algorithms in one attacking day we can observe that Algo1 has more events than the other two algorithms. However as the attacks are progressing during time algorithms “get used to” abnormality and after a specific time of attacks the abnormality becomes the normality of the measured feature. For this reason we shall highlight that the learning algorithms should be continually fed with normal behaviors in order to avoid the degeneration of values.

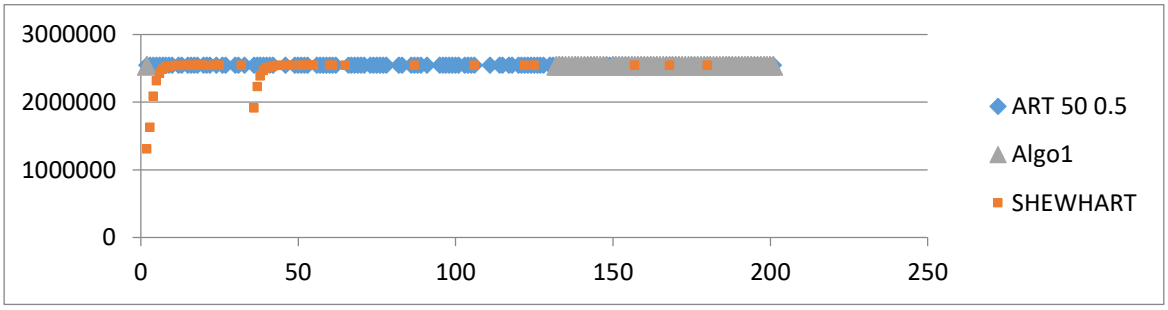

Figure 24 : Comparison of Algo1, Shewhart and ART algorithms on day 17

5. CONCLUSIONS

In this thesis we implement a framework for event detection in real time input streams. In more details, we track information from smart devices and feed them to our framework, in order to detect critical events and inform the smart devices for the possibility of an intrusion. The proposed approach offers an application solution to the problem of security intrusions (anomaly-based detection) by using streams generated by IoT devices relevant to their network properties in order to detect abnormal behavior and notify the user via an alert. In our case, each device participating in a IoT network is handled as a sensor device that generates streams of network measurements by using SNMP. These measurements are provided as input to CEP framework, i.e. Esper. CEP listeners detect and analyze the sensor streams in real time based on thresholds related to the normal behavior. Such abnormal statistical behavior can be a clear indication of an event occurrence (e.g., intrusion). The estimations of CEP engine are based on statistical predictors including machine learning methods like ART.

The dataset, that we used to take the experiments is “UNB ISCX Intrusion Detection Evaluation DataSet”. This dataset contains real traces, which are analyzed to create profiles for agents that generate real traffic for HTTP, SMTP, SSH, IMAP, POP3 and FTP. In our experiments, we tried to detect events concerning to TCP, UDP, Bandwidth, memory and CPU. In order to find the best way to discover an intrusion, we implemented three algorithms with different philosophy and compared them. A general outcome of the performance assessment was that the learning algorithms need constantly training to normal in order to avoid degeneration of values.

However, this framework is still in progress. As future work, we have schedule to implement a multivariate autoregressive(MAR) model, to compare it with the algorithms, that existed in our framework. Moreover, we want to connect a big number of smart devices, in order to calculate the latency and the communication overhead of the transactions between smart devices and our framework.

ABBREVIATIONS – ACRONYMS

IoT	Internet of things
SNMP	Simple Network Management Protocol
CEP	Complex Event Processing
ML	Machine Learning
DM	Data Mining
WWW	World Wide Web
DoS	Denial of Service
DDoS	Distributed Denial of Service
ARP	Address Resolution Protocol
DNS	Domain Name Service
MAC	Media Access Control
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
IP	Internet Protocol
HTTP	Hypertext Transfer Protocol
SSH	Secure Shell
IDPS	Intrusion Detection and Prevention System
IDS	Intrusion Detection System
FTP	File Transfer Protocol
HIDS	Host-Based Intrusion Detection System
NIDS	Network-Based Intrusion Detection System
IPS	Intrusion Prevention System
CRC	Cyclic Redundancy Check
MIB	Management Information Base
NMS	Network Management Station
OID	Object Identifier
PDU	Protocol Data Units
URL	Uniform Resource Locator
HTML	HyperText Markup Language
WSN	Wireless Sensor Networks
HMM	hidden Markov model
MOA	Massive Online Analysis
EPL	Event Processing Language
SQL	Structured Query Language
CPU	Central Processing Unit
RAM	Random-access Memory

AVQ	Adaptive Vector Quantization
UCL	Upper Control Limit
LCL	Lower Control Limit
ART	Adaptive Reasonance Theory

REFERENCES

[1] Douglas R. Mauro, Kevin J. Schmindt, Essential SNMP, 2nd Edition, OReilly, 2005

[2] Esper Team and EsperTech Inc., Esper 5.0.0 API Documentation. Available: http://www.espertech.com/esper/distributions/esper-5.4.0.zip (downloaded 2016, Apr.25)

[3] Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey, Computer networks, 54(15), 2787-2805.

[4] Luckham, David. 2002. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Pearson Education, Inc.

[5] Alpaydin , E. (2004). Introduction to Machine Learning . Cambridge , MA : MIT Press .

[6] Champ, C.W. and Woodall, W.H. (1987). Exact rules for Shewhart control charts with supplementary run rules. Technometrics, 29(4), 393–399.

[7] Umesh Hodeghatta Rao and Umesha Nayak, The InfoSec Handbook An Introduction to Information Security, ApressOpen, 2014, p.376

[8] https://www.manageengine.com/network-monitoring/what-is-snmp.html (last access at 30 September 2016)

[9] http://www.ibm.com/support/knowledgecenter/SSGU8G_12.1.0/com.ibm.snmp.doc/ids_snmp_009.htm (last access at 30 September 2016)

[10] https://www.manageengine.com/network-monitoring/what-is-snmp.html (last access at 30 September 2016)

[11] http://www.1to1media.com/data-privacy/ftc-leaves-internet-things-enforcement-door-open (last access at 30 September 2016)

[12] Zhenghong Xiao, Chuling Liu, Chaotian Chen, ”An Anomaly Detection Scheme Based on Machine Learning for WSN” IEEE International Conference on Information Science and Engineering, 2009

[13] Jonatan Gomez and Dipankar Dasgupta. Evolving fuzzy classifiers for intrusion detection. In Proceedings of the 2002 IEEE Workshop on Information Assurance, West Point, NY, USA, 2002.

[14] Y. Freund, Schapire.R. , ”Experiments with a new boosting algorithm” Thirteenth International Conference on Machine Learning, Italy, 1996.

[15] B.Pfahringer, Winning the KDD99 Classification Cup: Bagged Boosting, in SIGKDD Explorations, 2000.

[16] I. Levin, KDD-99 Classifier Learning Contest: LLSoft’s Results Overview SIGKDD Explorations, 2000.

[17] V. Miheev, Vopilov.A and Shabalin.I., The MP13 Approach to the KDD’99 Classifier Learning Contest’ SIGKDD Explorations, 2000.

[18] Jiankun Hu and Xinghuo Yu, ”A Simple and Efficient Hidden Markov Model Scheme for Host-Based Anomaly Intrusion Detection”, IEEE Network Journal, Volume 23 Issue 1, February 2009

[19] Jiong Zhang and Mohammad Zulkernine, Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection IEEE International Conference on Communications, 2006.

[20] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, Thomas Seidl. MOA: Massive Online Analysis, a Frameworkfor Stream Classification and Clustering. In proeceedings of the Workshop on Applications of Pattern Analysis, 2010.

[21] Bandyopadhyay, S., Sengupta, M., Maiti, S., and Dutta, S. (2011). Role of middleware for internet of things: A study. International Journal of Computer Science and Engineering Survey, 2(3), 94-105.

[22] Blackstock, M., Kaviani, N., Lea, R., and Friday, A. (2010). MAGICBroker 2: An open and extensible platform for the Internet of Things. In Internet of Things (IOT), pp. 18.

[23] Gert R. G. Lanckriet and Tijl De Bie and Nello Cristianini and Michael I. Jordan and William Stafford Noble, A statistical framework for genomic data fusion, Bioinformatics, 20(16), 2004, 2626-2635.

[24] Jun Wang and Huyen Do and Adam Woznica and Alexandros Kalousis, Metric Learning with Multiple Kernels, in proceedings of the NIPS-2011.

[25] Rong Jin and Steven C. H. Hoi and Tianbao Yang, Online Multiple Kernel Learning: Algorithms and Mistake Bounds, in proceeding of the ALT-2010, 390-404.

[26] IDS Dataset - http://www.unb.ca/research/iscx/dataset/iscx-IDS-dataset.html (downloaded on (last access at 30 September 2015)

[27] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A. Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Computers & Security, 31(3):357–374, May 2012.

[28] TCP Replay application, http://tcpreplay.synfin.net/ (downloaded at 30 June 2016)