JYVÄSKYLÄ STUDIES IN COMPUTING
231

Payam Vahdani Amoli

Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Network Attacks and Botnets

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 231

Payam Vahdani Amoli

Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Network Attacks and Botnets

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Agora-rakennuksen auditoriossa 2
joulukuun 14. päivänä 2015 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in building Agora, auditorium 2, on December 14, 2015 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2015

Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Network Attacks and Botnets

JYVÄSKYLÄ STUDIES IN COMPUTING 231

Payam Vahdani Amoli

Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Network Attacks and Botnets

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2015

Editors
Timo Männikkö
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Ville Korkiakangas
Publishing Unit, University Library of Jyväskylä

URN:ISBN:978-951-39-6452-8
ISBN 978-951-39-6452-8 (PDF)

ISBN 978-951-39-6451-1 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2015

ABSTRACT

Vahdani Amoli, Payam
Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Network Attacks and Botnets
Jyväskylä: University of Jyväskylä, 2015, 54 p. (+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 231)
ISBN 978-951-39-6451-1 (nid.)
ISBN 978-951-39-6452-8 (PDF)
Finnish summary
Diss.

Today, the occurrence of zero-day and complex attacks in high-speed networks is increasingly common due to the high number vulnerabilities in the cyber world. As a result, intrusions become more sophisticated and fast to detrimental the networks and hosts. Due to these reasons real-time monitoring, processing and intrusion detection are now among the key features of NIDS. Traditional types of intrusion detection systems such as signature base IDS are not able detect intrusions with new and complex strategies. Now days, automatic traffic analysis and anomaly intrusion detection became more efficient in field of network security however they suffer from high number of false alarms. Among all type of anomaly detection methods unsupervised machine-learning techniques are commonly applied in NIDS to detect unknown and complex attacks in the network without any prior knowledge. This dissertation manly focuses on analyzing network traffic to find abnormal behavior in real time. The proposed framework consists of network traffic preprocessing, anomaly detection and clustering methods. The proposed framework is capable of generating meaningful reports related to the detection of real intrusions in well-known datasets. Unsupervised learning methods are capable of adapting their required features to the dynamically behavior of the network. Due to unfeasibility of payloads checking in high-speed network the proposed framework monitors network flows instead. Network flow contains the behavior of the network in higher extensive vision and shows the explicitness of the network data, which results in faster and higher detection rate of network attacks. This research shows that by using proper data preprocessing and unsupervised data analyzing methods it is possible to detect fast and complex zero days (new) attack in real time. The practical experiments are presented in the included articles.

Keywords: machine learning, clustering (unsupervised), network security, anomaly detection, intrusion detection

Author Payam Vahdani Amoli
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisors Prof. Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Prof. Gil David
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers Prof. Amir Averbuch
School of Computer Science
Tel Aviv University
Israel

Adjunct Prof. Kari Luostarinen
Kehittämispäällikkö / Development Manager
Keski-Suomen liitto / Regional Council of Central Finland
Finland

Opponents Prof. János Sztrik
Department of Informatics Systems and Networks
University of Debrecen
Hungary

ACKNOWLEDGEMENTS

I have to honor to express my sentiment and sincerest gratitude to my supervisors, Prof. Timo Hämäläinen and Prof. Gil David, for their continues guidance and valued support which has been rendered during the course of my study in University of Jyväskylä.

My warm gratitude goes to my colleagues and collaborators, Mr. Farhoud Hosseinpour, Mr. Farhood Farid Etemad, Mr. Younes Abdi, Mr. Mohammad Tabatabaei and all others who have collaborated and helped me along the way.

I would also like to seize this opportunity to thank Faculty of Information Technology of University of Jyväskylä for giving this opportunity to me to work on my dissertation and supporting my research financially too. I would also like to thanks the Centre for International Mobility (CIMO) in Finland for partly supporting my research financially and giving me motivation toward.

I would like thank my lovely and kind wife, Mahsa, for her endless support and patients during the term of my PHD study. I would like to thanks to my parents and brother and sister who have given me the opportunity of study and rendering the best possible support and giving a perspective to the life beyond the academic world. I would like to thank my mother/father and brother in law for providing a loving and supporting environment for me. Last, but definitely not least, I thank my God, the compassionate and merciful who was helping me to achieve one the best goal of my life.

Jyväskylä 01.09.2015
Payam Vahdani Amoli

GLOSSARY

AI	Artificial Intelligence
AIS	Artificial Immune System
APT	Advance Persistent Threat
C&C	Command-and-Control
CIA	Confidentiality, Integrity and Availability
CPU	Central Proceeding Unit
CRA	Constructive Research Approach
DBSCAN	Density-based Spatial Clustering of Applications with Noise
DDoS	Distributed Denial of Service
DoS	Denial of Service
FNR	False Negative Rate
FPR	False Positive Rate
GA	Genetics Algorithm
Gbps	Gigabyte per second
HIDS	Host-based Intrusion Detection System
HIS	Human Immune System
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IPv6	Internet Protocol version 6
IRC	Internet Relay Chat
NIDS	Network Intrusion Detection System
NMAP	Network Mapper
OS	Operating System
R2L	Remote to Local
SSH	Secure Shell
TNR	True Negative Rate
TPR	True Positive Rate
U2R	User to Root

LIST OF FIGURES

LIST OF TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

PI. Etemad, F. F. & Amoli, P. V. 2012. Real-time Botnet command and control characterization at the host level. Telecommunications (IST), 2012 Sixth International Symposium on. Tehran, Iran: IEEE, 1005-1009.

PII. Amoli, P. V. & Hämäläinen, T. 2013. A real time unsupervised NIDS for detecting unknown and encrypted network attacks in high speed network. Measurements and Networking Proceedings (M&N), 2013 IEEE International Workshop on. Naples, Italy: IEEE, 149-154.

PIII. Hosseinpour, F., Ramadass, S., Meulenberg, A., Amoli, P. V. & Moghaddasi, Z. 2013. Distributed Agent Based Model for Intrusion Detection System Based on Artificial Immune System. International Journal of Digital Content Technology and its Applications (JDCTA) 7(9), 206-214.

PIV. Hosseinpour, F., Amoli, P. V., Farahnakian, F., Plosila, J. & Hämäläinen, T. 2014. Artificial Immune System Based Intrusion Detection: Innate Immunity using an Unsupervised Learning Approach. International Journal of Digital Content Technology and its Applications (JDCTA) 8(5), 1-12.

PV. Amoli, P. V., Hämäläinen, T., David, G., Zolotukhin, M. & Mirzamohammad, M. (Accepted Nov/2015). Unsupervised Network Intrusion Detection Systems for Zero-Day Fast-Spreading Attacks and Botnets. International Journal of Digital Content Technology and its Applications (JDCTA).

1 INTRODUCTION

This chapter presents the motivation behind the research concerning machine learning algorithms, which are used for anomaly detection in network security. Next, the research questions are answered. Finally, the overall structure of the work and author's contribution in the included articles is briefly described.

1.1 Research motivation

Nowadays, computer and network revolutionized our daily life. Most of the personal, organizational and governmental information store and transfer via computers and networks. Due to the growing number of cyber-attacks, computer security become more important than ever and considered as the principal function in any system or organization.

The three advance security layers to minimize the risk of attacks are (Abad et al. 2003, Komninos, Vergados & Douligeris 2010):

1. Prevention
2. Detection
3. Reaction

Prevention is the first security level which protects system and network from intruders. Access controls, security policies, security awareness and intrusion prevention systems (IPS) are the main elements of prevention layer. The main outcome of prevention layer is to identify and patch security vulnerability of network and system. Due to the increasing rate of zero-day (new) attacks, detection layer become the most important security level since most advance security mechanism in prevention layer may not stop the motivated and high skill intruders. Intrusion detection system (IDS) has the capability of monitoring the network and system activities to detect the intrusions and notify the administrator. Reaction layer contain pre-planned procedure after the intrusion

detection such as stopping the intruder, fixing the newly founded vulnerability and restoring the system and data.

Zero-day attacks can be fast, brutal and complex. Fast attacks aim to fill the network with enormous amount of traffic to cause latency in the network or disrupt a service on specific machine (server). Brutality and complexity of attack can result in data breach and hacking while they deceive the IDS. Due to the increasing rate of zero day attacks, researchers are investing more on finding the most suitable methods to increase the detection rate of fast and sophisticated attacks while traditional techniques such as misuse (signature based) detection methods are not capable of detecting these types of attacks.

In (Denning 1987) the first model of anomaly detection proposed. In anomaly-based detection techniques, normal state will be defined and behaviors which pass the criteria will be flag as abnormal. Using this method will increase the probability of detecting novel attacks. One of the well-known anomaly detection methods is probabilistic which rely only on statistics and do not correlate alarms. On the other hand, scenario-based methods need to observe specific steps to detect attacks. Due to the dynamic and complex structure of sophisticated attacks, probabilistic or scenario based NIDS may produce high number of false alarm. Machine learning can be considered as the central sub-set of the Artificial Intelligence (AI). Machine learning algorithms construct a model from example inputs and use it for decision and prediction making in future. Due to the learning and decision making capability, many researchers applied machine learning techniques in IDS to improve the performance of attack detection (Nguyen & Armitage 2008). There are no known machine learning methods that can be applied in IDS for detecting all types of attacks. Finding the right input data and applicable algorithms for real life situation is the main challenge of using machine learning in IDS.

1.2 Research questions

The objective of the research is to study and improve different methods of data preprocessing and machine learning in network security. To achieve this, the dissertation presents several case studies. In most of these studies, one or more machine learning algorithms are customized and employed in order to solve well posed problems. The main research questions of this study are as follows:

1. How to monitor unbalance behavior of machines' and high speed networks' in real time
2. How to detect fast spreading network intrusion (such as denial of service attack or scanning) without any prior knowledge
3. How to detect complex attacks (such as Botnet) without any prior knowledge
4. Is it practical to apply machine learning algorithms in IDS in real time
5. Is it possible to detect intrusions in encrypted communications

6. How effective are decentralized monitoring methods for improving the speed and detection rate of intrusions

1.3 Research approach

To answer the previously mentioned questions, this research uses Constructive Research Approach (CRA) (Piirainen & Gonzalez 2013). This research aimed to create an innovative model to a real problem and contribute to the particular field of science where it has been applied. CRA consists of:

- Planning phase: Discovering and selecting the scientifically relevant cases in network security
- Analyzing phase: Analyzing and literature review will be done to gain a comprehensive understanding of the problems and previously proposed methods on real-time intrusion detection system and unsupervised machine learning which applied in network security
- Designing phase: Creating a blueprint from the innovative model which is capable to solve the cases which found in analyzing phase
- Implementation phase: Developing the predefined project
- Testing and documentation phase: Testing the proposed model on well-known traffic samples and gathering the produced data (results) and compare the result with the previous proposed solutions

In this research, the problem of fast spreading network intrusions and complex attacks in normal and encrypted communication has been studied. The proposed solution is an unsupervised multi-stage network intrusion detection system which is capable of monitoring high speed networks in real time.

1.4 Structure of the work

The rest of this dissertation is organized as follows. First, the theoretical background on intrusion detection and machine learning is introduced. Then, the contributions and results obtained in the research articles and new unpublished results are presented. Finally, the dissertation is concluded and outlines the future works and research directions.

Chapter 2 covers deep explanation of potential intrusions that threaten networks and systems. These intrusions include new types of attacks which are fast, brutal and complex. Furthermore the history and current state of intrusion detection system will be explained in detail. Chapter 3 presents deep technical aspects of data prepossessing and data analyses algorithms (supervised and unsupervised) which have been used in this research are presented.

Chapter 4 outlines the research contribution of the dissertation. It contains the results presented in the included research articles to discuss the benefits and performance of using different machine learning algorithms for intrusion detection. In addition, some new framework and results that have not yet been published are added to support the research work.

Finally, Chapter 5 concludes the work and provides future research directions.

1.5 Research contribution

The author's contribution to the included articles is in the design and development of the entire framework with co-authors. Figure 1 shows the relation of included articles, their common topics and their place in overall scheme. The deviation is based on data source (host or network) for the IDS and their detection methods (supervised or unsupervised).

Article PI presents a real time model for host based intrusion detection system (HIDS) to deal with well-known centralized Botnet attacks. Article PII propose a multi stage unsupervised network intrusion detection system (NIDS) for detecting fast-spreading and complex network attacks. Article PIII uses artificial immune system (AIS) to train the IDS. In addition it monitors and learns the current behavior of network from distributed network sensors for detecting new types of intrusions. Article PIV uses the first unsupervised engine which proposed in PII to add the capability of online training for the central AIS engine which proposed in PIII. Finally, article PV implemented the both engines which proposed in PII and enhanced the detection rate of unsupervised engines in PIV. Each article is discussed in detail below.

Article PI presents a real time and active model for detecting centralized Botnets in host level. The proposed HIDS is capable of inspecting the hypertext transfer protocol (HTTP) and internet relay chat (IRC) packets and dropping suspicious packets while it observe patterns of command-and-control (C&C) communications. The experimental part consists of inspecting HTTP and IRC packets from a real Botnet traffic. The capability of bot detection by real-time processing of host-related data solely, distinguishes this model from other existing approaches. The author is responsible for literature review on Botnets behavior, proposing the packet inspection and design the overall framework

Article PII presents a multistage unsupervised NIDS to detect intrusions in high speed network. Due to the infeasibility of checking packets payload in encrypted or high speed networks the proposed model uses network flow as the input data for NIDS. The intrusion detection procedure has been divided in two stages, the first engine is responsible to detect fast spreading network attacks in real time and the second engine conducts deeper analysis and correlates the traffic of the previous attackers to find the potential Botnet. The main novelty of the propose model is to train DBSCAN clustering algorithm by clean network traffic before the attack to represent the normal behavior of

network to the clustering engine. The author carried out deep research on finding the most effective unsupervised machine algorithms which was used and implemented in previously proposed NIDS and proposed the overall framework.

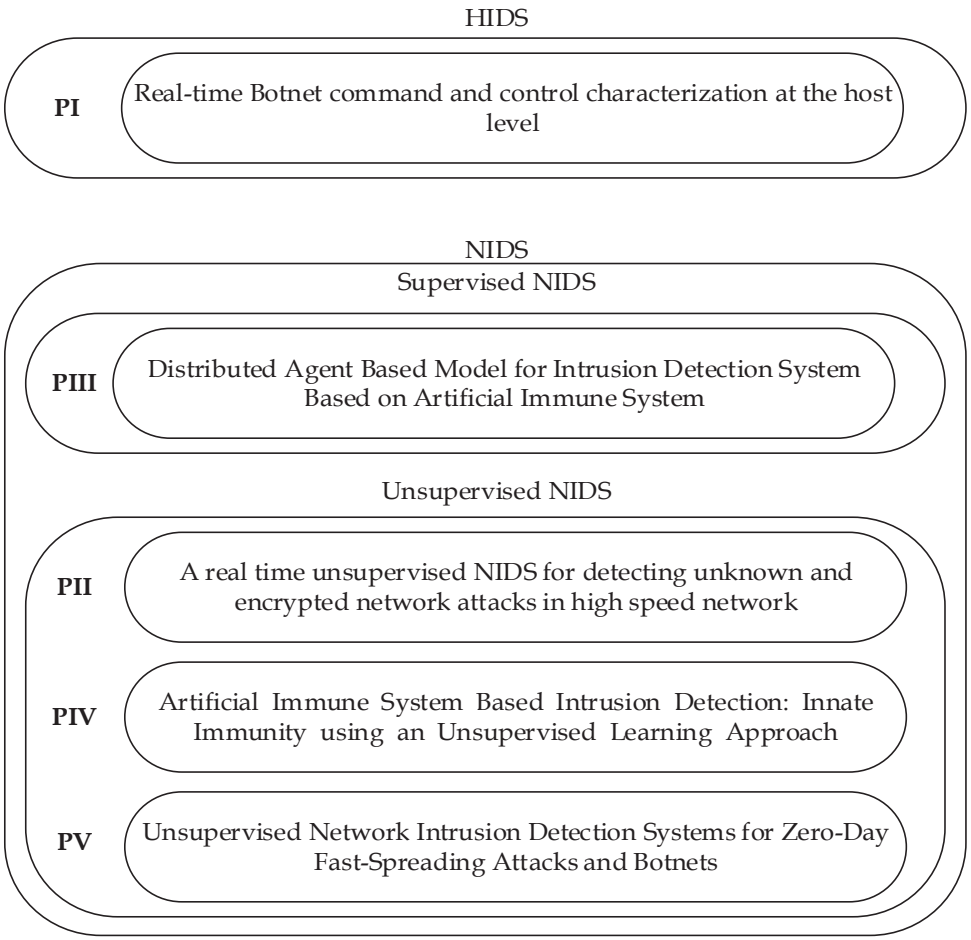


FIGURE 1 The Common Topics of Different Papers.

Article PIII presents a distributed agent based design of AIS for IDS. The detectors are distributed in each host in network. The central engine is located in server and manages the detectors and makes final decision about current behavior of the network. The detector agents actively updated and synchronized with detector agents of other hosts through the IDS's central engine. The main novelty of the proposed model was to apply distributed model in AIS based IDS to increase the speed and detection rate of intrusions. Based on the results during the test phase new types of anomaly were detected

due to the dynamically increased number of memory cell in each host. The author's contribution includes the implementation of the framework.

Article PIV combines the capability of PII and PIII. The novelty of this work is utilization of AIS and DBSCAN (density-based spatial clustering of applications with noise) in order to provide online and real-time training for the adaptive immune system within the central artificial immune system. Different methods for unsupervised machine learning are investigated and DBSCAN is selected to be utilized in this architecture. The adaptive immune system also takes advantage of the distributed structure, which has shown better self-improvement rate compare to centralized mode and provides primary and secondary immune response for unknown anomalies and zero-day attacks. The author contributed to the idea of using unsupervised machine learning algorithms with AIS to add the online capability to the central engine and implement the overall framework.

Article PV implement the proposed model in PII and enhanced the detection rate in PIV. The model detects network intrusion without any prior knowledge via two separate engines. The first engine detects fast-spreading DoS, probes and DDoS attacks (e.g. POD, SMURF, Mail-bomb, SSH-process-table, UDP Storm, port scanning, network scanning) in real time to stop the paralysis of both network and victims. The second engine finds the eventual internal Botnet (Bots or Botmaster), while the monitored network filled by DDoS attacks traffic implement the overall framework. One of the main novelty of the propose model is using a dynamic and self-adaptable threshold to detect unexpected behavior in the network to decrease the computation time of the clustering process during the normal state of the network. Standardizing data input via a logarithm (log) and monitoring the different size of subnets through the threshold increase the performance of the NIDS. To evaluate the proposed model, the NIDS tests with two publicly available and well-known datasets to ensure the detection process. The author implements the overall framework.

2 INTRUSION DETECTION SYSTEM

This chapter presents the fundamental concept of intrusion and intrusion detection system. First, different types of intrusions and their signatures (symptoms) will be explained. Next, different architecture of intrusion detection systems (IDS) and their specificities properties will be discussed.

2.1 Intrusions

Intrusion is a formal term for describing the malicious act of compromising a network or system. Accessing or manipulating data should be authorized by sets of rules which defined in confidentiality, integrity and availability (CIA triad) policies of the data. Attackers aim to bypass layers of computer security (which presented in section 1.1) to breach the confidentiality, integrity and availability of data or services (Hernández-Pereira et al. 2009, Kumar, Kumar & Sachdeva 2010).

Based on the literature, successful intrusions aim to pass through the main four stages as listed below (Asaka, Taguchi & Goto 1999, Kruegel, Vigna & Robertson 2005):

- Probing stage:
  Considered as the first stage of intrusion, which intruders scan the victims' systems or network to collect information related to their potential vulnerabilities. (Also known as scanning, surveillance or search stage)
- Exploitation stage:
  In case of finding vulnerability in probing stage, the intruder tries to gain the control of victim's machine for further activities. (Also known as activity stage)
- Action stage:
  By gaining the control of victim's machine in activity stage, intruder can access and manipulate victim's data and install malware to attack other

systems in the network. Malware or malicious software is a program to penetrate computers without the user's permission or notification. (Also known as mark stage)

- Masquerading stage:
  Finally, the intruder tries to removes or hides the traces of the attack.

Based on the stages which described above, Intrusions can be divided into four categories (Lippmann et al. 2000):

- Probe:
  As mentioned above this type of attack looks for live IP addresses (valid IP), open ports, victims' operating systems (OS) and other useful information to find the potential vulnerabilities in each host. IPsweep (network scanning), Portsweep (machine scanning), Nmap, Mscan, SAINT (Security Administrator's Integrated Network) and satan are the common probe attacks (Ghorbani, Lu & Tavallaee 2009). Network Mapper known as Nmap (Lyon 2009) is a well-known network scanning program for security purposes which have been used to extract data about hosts and their services for creating a map of the targeted network. Other recent offensive methods of scanning for large networks have been proposed (Durumeric, Wustrow & Halderman 2013) to maximize the scanning performance which may paralyze the network and targeted hosts due to the high number of network flows.
- Remote to Local (R2L):
  In R2L attack, intruder attempts to obtain a local account in the network through the founded vulnerability in probing stage. Social engineering, man in the middle, password guessing are the well-know and common R2L attacks. Password guessing which mainly performed by SSH brute force is an old type of attack which still strongly occurs (Cid 2015) to the web server by trying all possible combination of characters to find the correct keys (passwords).
- Denial of Service(DoS):
  DoS is a network level attack which aims to disrupt the usability of a service or network. Attacker uses a compromised machine to sends high amount of malicious traffic to specific machine/s for paralyzing their services and network. Engaging DoS attack via high number of compromised machines is referred as Distributed Denial of Service (DDoS). Today, many internet users do not install or update proper security software (such as: firewall and antivirus). Professional attackers use automated tools to find vulnerable machine and install malware to compromise them. The compromised machines are referred as Robots or Bots. Botnet is a collection of compromised computers (Bots) which are remotely controlled by the intruder (BotMaster) under a common Command-and-Control (C&C) infrastructure. Botnets are used to perform malicious activity in wider scale such as DDoS attacks and spamming. SYN flooding, Ping of Death (PoD), HTTP flood, XOR DDoS Botnet, Smurf

attack, Mail-Bomb are the well-known DoS and DDoS attacks. SYN flood is an old and well know DoS attack which intruder sends high number SYN requests to the victim in an attempt of consuming high resource from the victim to make it unresponsive to legitimate requests. High skilled intruders are able to deceive and bypass firewalls and antiviruses and victimize the most advance and up-to-date operating system such as Red Hat Enterprise Linux 7 (Brouer 2014). For instance in September 2015, distributed SYN flood had been detected with bandwidth range from few to 170 gigabyte per second (Gbps) victimizing gaming sectors and educational institutes in Asia. The attacker used SSH Brute force attack to gain privilege to several Linux servers in Asia to construct his own botnets. Afterward with the compromised Linux servers, he launched crippling DDoS attacks of over 150 Gbps (Khandelwal 2015). PoD is another well-known DoS attack which the intruder attacks machines through sending malicious ping requests. This issue was fixed in many operating systems by 1998 however with the recent usage of Internet Protocol version 6 (IPv6) different version of Microsoft operating system were vulnerable to it (Jackson 2013). HTTP flood is a new DOS/DDoS attack which occurred for the first time in 2009. It victimize web servers by sending high number of legitimate sessions of HTTP GET or POST requests to make the web server unresponsive (Cid 2014).

- User to Root (U2R):
  U2R and R2L are in same class of attacks however in U2R the intruder has local access and tries to access and manipulate the policy file in the OS to gain administrator privilege. Buffer overflow, Sql-attack and perl are the common U2R attacks.

Besides the traditional categorizing of intrusions which discussed above many new types of sophisticated attacks have been discovering in the recent decade. One of the big and well-known classes of intrusion is advanced persistent threat (APT). It consists of stealthy though continuous hacking processes which operated by high skilled hackers. The main difference between APT and traditional threats is the stealthy and data-focused nature of it. Based on statistics the main victims of APT are business and political organizations. The term "advanced" refers to applying sophisticated malware to take advantages from the existing vulnerabilities of victim. The term "persistent" refers to the external continuously monitoring method to collect confidential data from the victim. At the end "threat" refers to the process involved by high skilled hacker to organize the attack (Tankard 2011, Cole 2012).

Stuxnet is one of the well-known examples of an APT which was highly sophisticated and targeted specific infrastructure (such as Iran's nuclear facilities) via numerous zero day vulnerabilities and spread via several propagation methods (Virvilis & Gritzalis 2013).

Recent evidence which mentioned above shows that conventional security measures like firewalls, anti-virus and signature based IDS are not

enough, since sophisticated intrusions can deceive or bypass them. Due to this reason deeper and more automated analysis of data is the baseline for network and machine monitoring (Tankard 2011).

2.2 Intrusion detection system

Intrusion detection system (IDS) is a tool or device which monitors the behavior of network or systems to detect abnormal activity. IDS notifies administrator regarding the observed suspicious activities and in some case IDS is capable of blocking the abnormal traffics or activities (Patcha & Park 2007). Figure 2 shows the overall classification of IDS.

IDS can be categorized based on the input data (Xin, Dickerson & Dickerson 2003, Engen 2010). The IDS which monitors and inspects the behavior of the whole network is NIDS (network-based intrusion detection system). Current NIDS solutions monitor bytes, packets' payload or network flows to detect intrusions. According to (Claise 2008) "A flow is defined as a set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties". Each network flow contains information about IP addresses and port numbers of source and destination, number of packets, protocol, duration, average size of packets and other useful information which can be retrieved from packets header.

Whereas, the IDS which derives information from single host is HIDS (host-based intrusion detection system). HIDS monitors and inspect the system activities such as: incoming connection attempts, network traffic, login information and resource usage (CPU, Memory, Storage and etc.).

In general there are two main detection methods for IDS: Signature-based and anomaly-based. Signature-based IDS monitor the behavior of machine or network and compare it with the characteristics of known attacks. Signature-based IDS have high detection rates for well-known attacks; however, as mentioned before they even fail to detect known intrusions with small variations to their signatures. Providing attack signatures consumes money and time, and with the increasing rate of zero-day attacks, using signature-based IDS is not a safe solution.

In anomaly-based detection techniques, normal state of system or network will be defined. If behavior of system or network passes the criteria, the IDS will be suspicious and flag it as abnormal. Using this method will increase the probability of detecting novel attacks; however, it makes lots of detection errors because of the difficulty of defining the normal state. Having fewer false alarms and an increased detection rate of zero-day and complex attacks, especially in imbalanced network traffic, has become an important challenge in the design of detection techniques for IDS (Sperotto et al. 2010, Engen 2010).

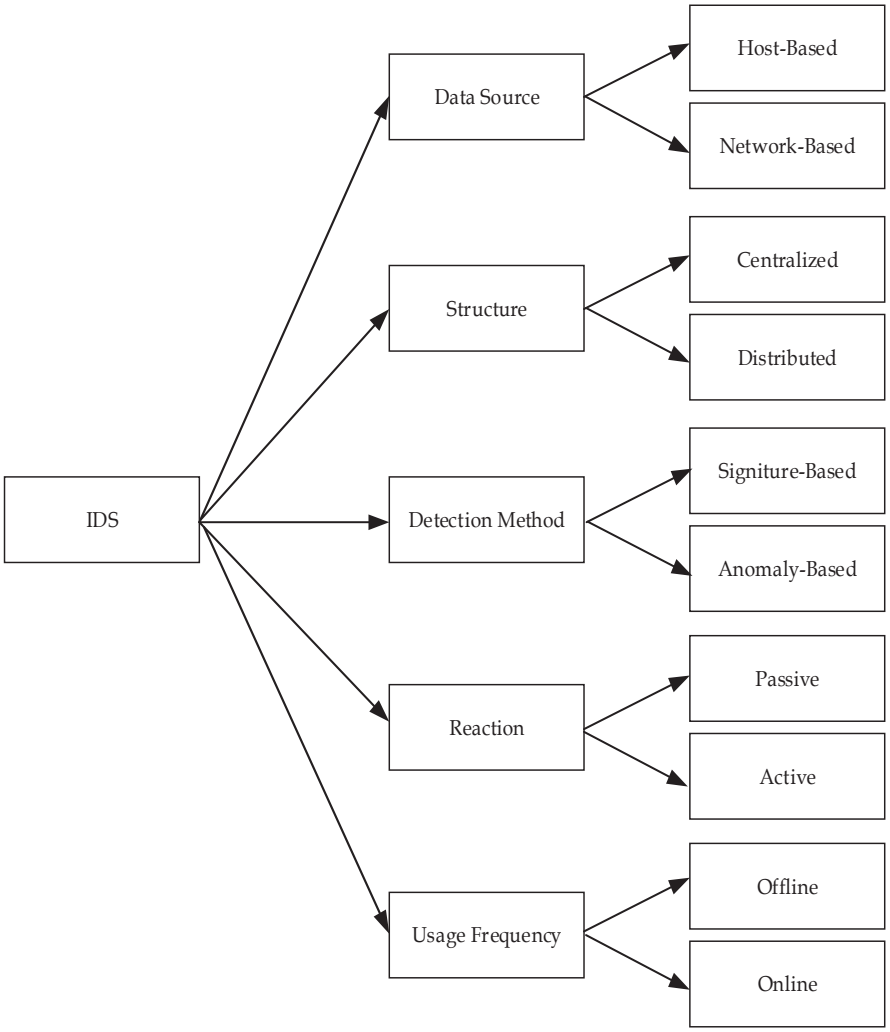


FIGURE 2 Classification of Intrusions Detection System (IDS).

From the reaction perspective, IDS can be categorized in two groups: passive and active. Passive IDS stores and log the detail detected intrusion and send it to the administrator however active IDS take immediate action on the intrusion to stop it. In real life, active IDS are the efficient solution since they stop the intrusion immediately and mitigate the damage of intrusion to the system and network, however, false alarm may create problem for legitimate actions (Engen 2010).

Finally, the last way to classify IDS is based of usage frequency: offline IDS or online IDS. Offline IDS analyses pre-logged data to find intrusion however

they are not efficient solution for fast and brutal attacks. In the other hand online (real-time) IDS are capable of detecting intrusions immediately and make it possible for administrator to mitigate the damage in intrusions.

3 MACHINE LEARNING

This chapter discusses the overall structure of data preprocessing, data analyzing and performance evaluation. It is important to note that many algorithms can be used in data preprocessing or analyzing, however, the chapter mostly focuses and explains the algorithms which applied in the articles included in this dissertation. First, the preprocessing procedures such as data selection, feature extraction and selection will be outlined. Next, machine learning methods such as classification and clustering are presented. At the end, performance evaluation techniques are described.

3.1 Data gathering and preprocessing

In general data mining algorithms which use machine learning methods have inductive bias. As a result, the characteristics of the data being mined will directly affect the performance of machine learning methods (Freitas & Timmis 2007). Preprocessing the logged or live data (raw data) aims to remove irrelevant (redundant) and duplicated information to prepare the preprocessed input data for the analyzing phase since the raw data may contain chaotic, missing and irrelevant data. The other two objectives in preprocessing data are data extraction which masked by another of data (such as noise) and dimention reduction. Preprocessing data (Fayyad, Piatetsky-Shapiro & Smyth 1996, Hand, Mannila & Smyth 2001) contains several steps. Each step performs specific actions to extract the useful information form the available data for the analyzing phase. The main five main stages of data preprocessing are listed and explained below.

3.1.1 Data selection and feature extraction

Depend on the application area, the most available and relevant source of data will be selected. In feature extraction phase, sets of derived values (features)

will be extracted from the raw data. Feature extraction reduce the representation of data to increase the performance of data processing since analyzing the full size raw data is time consuming and decrease the accuracy of output. For instance, to analyze network traffic for network-based intrusion detection, the application should gather the data from routers and extract information such as number of transferred bytes, packets, networks flows, IP addresses, protocol, port numbers and other useful features of network traffic for intrusion detection (Guyon & Elisseeff 2003, Liu & Yu 2005, Novakov et al. 2013). Many researchers are suggesting using network flows for intrusion detection purposes (Lakhina, Crovella & Diot 2004, Peng, Leckie & Ramamohanarao 2004, Mark, Crovella & Diot 2004, Tedesco & Aickelin 2006). Monitoring network flows enhances the detection rate of complex attacks and decreases false alarms. As network attacks may occur in several stages or via a lengthy communication, inspecting the packet's payload or counting the number of transferred bytes may not provide sufficient information for their detection. Sampling network traffic is one of the main solutions to reduce the resource requirement and computation time of analyzing the packet's payload; however, it increases the probability of losing anomalous data (data related to intrusions) and pushes the NIDS to produce a high level of false-negative alarms. The extracted features such as network flows improve the detection rate since they contain the behavior of the network and the nodes in higher extensive vision. As the data volume of network flows is only 0.1 per cent compared to the packet payload, real-time detection is practical and implementing the NIDS in a high-speed network will be feasible (Sperotto et al. 2010). In addition intrusions in encrypted communication raise a false-negative alarm in payload-based NIDS as a result of the inaccessibility of the packets' payload; however, monitoring and inspecting encrypted communication in the form of network flows provides useful information to the NIDS (Koch & Rodosek 2010, Augustin & Balaz 2011).

3.1.2 Standardization

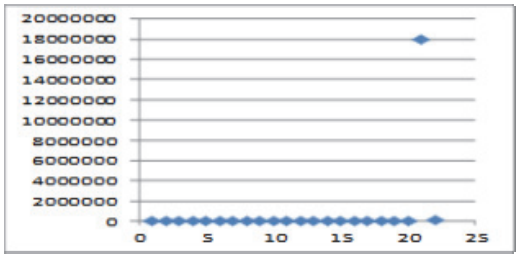
Extracted features from the raw data can be divided into two main categories: discrete and continuous. Discrete data can be finite number of values which are discrete and there is no grey area in between, such as protocol types of communication. However continuous data can take any value and there is no restricted predefined restricted separate values such as the duration of network flows between two machines.

Continuous data can be different in scales. Unfair comparison of features during data analyses phase can result in false learning (and decisions making). Data standardization techniques try to define a standard scale for all features (with different scales) to effect equality during analyses phase.

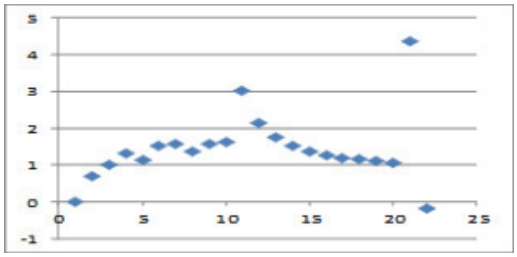
Data standardization is the process of removing variant scale of different data features. Data with off the scale magnitude may dominate other feature during data analyses. Natural logarithm (Log) and z-score are the well-known data standardization methods (Chiu et al. 2015).Logarithmic transformation of

data is suitable where the data covers a wide range of values. Among all of the standardization methods, Log transformation is one of the best methods to reduce the impact of outliers since it squeezes the bigger values and stretches smaller values. Figure 3 show the standardization result of network traffic sample. As shown the abnormal behavior started from 11th second of network traffic however the high traffic during 21st second masks all of the abnormal behaviors. Figure 3 (A) shows the normal view of traffic sample. Figure 3 (B) shows the data which standardized through z-score method, however due to the high amount of traffic in 21st second most of the abnormal behaviors have been masked. The log transformation which has been shown in Figure 3 (C) was particularly effective in standardizing positively skewed distributions (Leydesdorff & Bensman 2006).

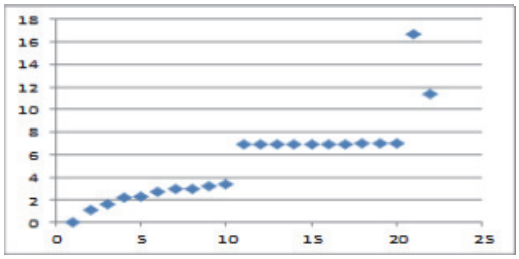
Time	Network Flows
1	1
2	3
3	5
4	9
5	10
6	15
7	19
8	20
9	25
10	29
11	1000
12	1001
13	1005
14	1009
15	1010
16	1020
17	1025
18	1059
19	1065
20	1070
21	18000000
22	90000



(A)



(B)



(C)

FIGURE 3 Data Standardization.

3.1.3 Feature selection

The number of extracted features from the raw data can be huge. Redundant and irrelevant features should be removed without losing important information, since they may result in the high computational burden and memory usage during data analyses. Feature selection can result in dimensional reduction since it has been used for simplifying model interpretability, reduce training times and enhance generalization by reducing over fitting.(James et al. 2014)

3.1.3.1 Subspace clustering

Utilizing high dimensional data in machine learning algorithms increase the performance of prediction and anomaly detection since the algorithm can model and learn behavioral changes of each specific feature. This has motivated researchers to development techniques for finding a low-dimensional representation of a high-dimensional data. As mentioned before feature selection removes irrelevant and redundant features by analyzing the entire dataset.

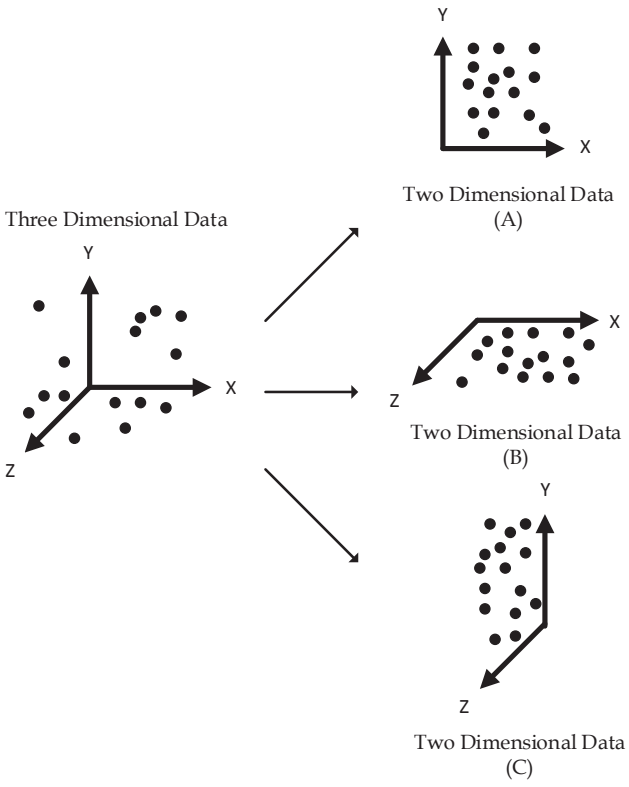


FIGURE 4 Subspace Clustring.

Subspace clustering is a feature selection method which localize the search for relevant dimensions allowing them to find Clusters that exist in multiple and overlapping subspaces (Vidal 2010). Subspace clustering has been applied in machine learning clustering-based methods (Boult & Brown 1991, Goh & Vidal 2007, Elhamifar & Vidal 2009, Elhamifar & Vidal 2010). Figure 4 shows how a three dimensional data will be divided into 3 two dimensional data through subspace clustering algorithm.

3.1.3.2 Genetic Algorithm

Genetic algorithm (GA) inspired by Darwin's theory of evolution. The process starts with a set of population. Samples of population will be taken and used to form a new population. The aim is to produce a better population. The new produced population will be selected, prioritized and reproduce based on their fitness. As show in Figure 5 this procedure will continue and repeated till some conditions (for example number of populations or improvement of the best solution) meet the criteria. GA tends to an optimal solution by using crossover and mutation processes similar to evolution.

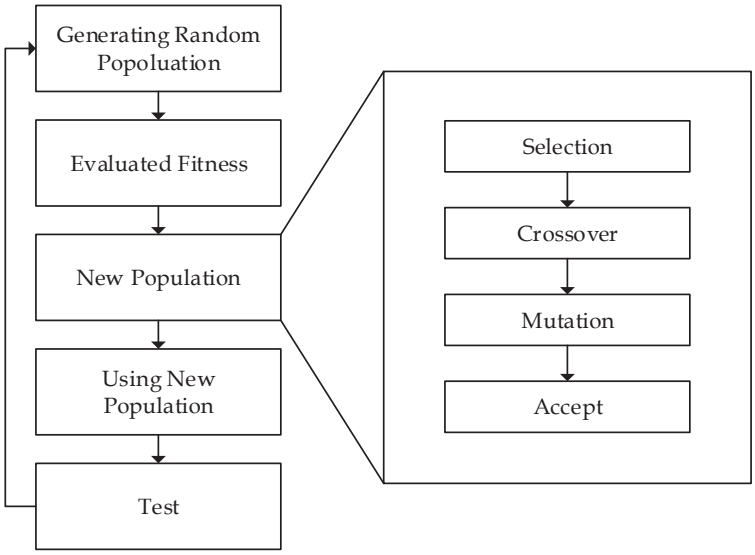


FIGURE 5 Outline of Genetic Algorithm (GA).

Crossover combines more than one population (parent) to produce a new population (children). As a result the child may be better than any of the parents since there is a probability of taking the best characteristics from them. There are many well know methods to apply crossover such as: One-point crossover, two-point crossover, Uniform crossover (Molina, Belanche & Nebot 2002, Zhang & Chang 2009) which shown in Equation 2-4.

One-point Crossover:

Parent-A **01011**100 + Parent-B 11001**001** = Children **01011001** (2)

Two-point Crossover:

Parent-A **01**0111**00** + Parent-B 11**0010**01 = Children **01001000** (3)

Uniform Crossover:

Parent-A 0**1**0**1**11**0** + Parent-B **1**10**0**1**0**01 = Children **11001000** (4)

Mutation is used to maintain genetic diversity from one generation of a population (parents) to the next (children). In mutation, the new population may change entirely from their parents. Mutation alters one or more values from parents to obtain better solutions (children).

Mutation:

Parent 011**1**10**0**10 → Children 011**0**01**1**10 (5)

3.2 Data analyses

Analysis of data is a process of inspecting, modeling and extracting information from preprocessed data. Data analysis has multiple approaches and techniques to extract and discovering useful information for the prediction and anomaly detection. Machine learning algorithms have been used broadly in field of data analyses (Liao & Vemuri 2002, Ramadas, Ostermann & Tjaden 2003, Kruegel et al. 2003, Estevez-Tapiador, Garcia-Teodoro & Diaz-Verdejo 2003, Li 2004). Machine learning techniques establish an explicit or implicit model for categorize the input data and they are capable of changing their execution strategy while it acquires new data (the self-learning capability) (Garcia-Teodoro et al. 2009).

In general there are three types of machine-learning technique: supervised, semi-supervised and unsupervised. In supervised machine learning techniques, the engine needs to be trained properly by a labeled dataset in order to create models for future prediction or decision-making; however, the attainment of labeled data needs to be carried out by experts, which is both costly and time-consuming(Kotsiantis, Zaharakis & Pintelas 2007).

Semi-supervised machine-learning techniques need to be trained by small amounts of labeled data and large amounts of unlabeled data to build the model of normal and abnormal data(Chapelle, Scholkopf & Zien 2006). Unsupervised machine learning techniques discover and formulate the invisible model of unlabeled data without any prior knowledge(Kotsiantis & Pintelas 2004).

3.2.1 Supervised Machine Learning Algorithems

Supervised machine learning algorithms referred to the methods which predict based on pre-observed evidence in the training dataset. The training dataset includes data and the response examples. Supervised learning algorithms analyze the training dataset and generate an inferred function, which can be used for mapping new samples. Observing more data result in improvement of the prediction performance. The main challenge in each application area is to choose the best algorithm which can correctly determine the class labels for unseen samples. Figure 6 shows the overall procedures of testing and training phase in supervised machine learning.

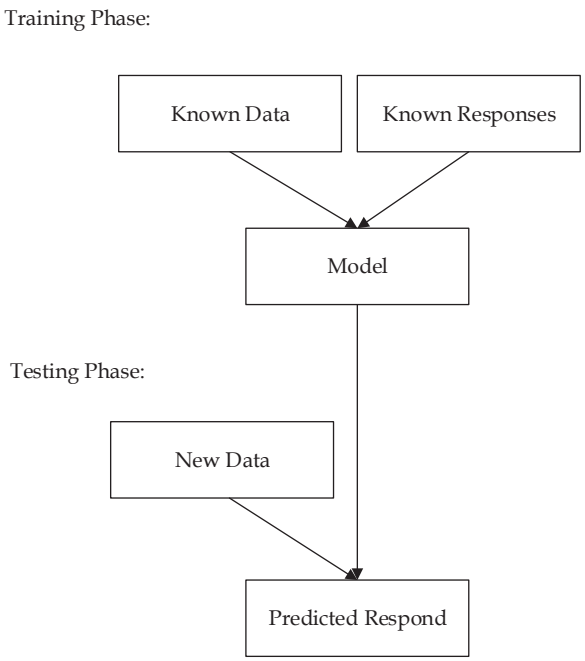


FIGURE 6 Testing and Training Phase in Supervised Machine Learning.

Supervised machine learning algorithms can be categorized as classification and anomaly detection. In classification, each training instance is already classified into one of the predefined categories. During the training phase the aim is to discover the relationship between the instance and it's category without any observation of the test data. The discovered rules should be useful to predict the class of each unknown instances in the test data. Classification algorithms usually apply to nominal response values (Kesavaraj & Sukumaran 2013). In anomaly detection the goal is to identify data points that are abnormal. Since the potential variations are high and the training instances are few it's not

feasible to train the system. Anomaly detection methods will train by normal activities and then identify significantly differences during testing phase (Chandola, Banerjee & Kumar 2009).

Classification and anomaly detection methods have been applied in broad domains of application such as network traffic monitoring (Li & Kianmehr 2012, Huang & Huang 2013), credit analysis (Hsu & Hung 2009), and biomedical modeling (Retnakaran & Pizzi 2005).

3.2.1.1 Artificial immune system

AIS has been defined (De Castro & Timmis 2002) as "Adaptive systems inspired by theoretical immunology and observed immune functions, principles and models, which are applied to problem solving".

Among different types of algorithm which have been inspired by the biological systems such as evolutionary algorithms, swarm intelligence and neural networks, AIS algorithms are bio-inspired from the human immune system. Each common technique in AIS is inspired by specific immunological theories such as: clonal selection, immune networks and negative selection.

The idea of clonal selection focus on the Darwinian attributes of the theory where the selection and reproduction of antibodies is prioritized based on the affinity rate of produced antigen and antibody. Clonal selection algorithms have been applied in optimization and pattern recognition domains.(De Castro & Von Zuben 2002)

Beside the respond to the antigen which produced by external invaders, lymphocytes may attack to the materials which produced by internal host own cells. A full immune response may damage the host's organism. Negative selection algorithm inspired the positive and negative selection processes that occur during the maturation of T cells in the thymus. Negative selections identify and delete actions which may attack self-tissues. These algorithms have been applied in classification and pattern recognition domains where the algorithm can learn from the labeled dataset (Forrest et al. 1994, Esponda 2005). The following algorithm shows the basic pseud code of negative selection in AIS (Dal et al. 2008):

```
Input: a set of normal (self) and abnormal (non-self) data instances
REPEAT
   Randomly generate immature detector
   IF match with self
      THEN Discard
   ELSE IF match with non-self
      Measure the affinity between detector and non-self
      IF affinity between the detector and non-self passes threshold
         Add to finial detector set
UNTIL stopping criterion
Output: a set of mature detector
```

Negative selection algorithms can be used to detect abnormal behavior inside of network or host (Hofmeyr & Forrest 2000). Human immune system (HIS) is able responds quickly to perversely seen antigen since it can remember their specific antibody. Creating memory cells via genetic algorithm have been

applied in negative selection methods in AIS to form a secondary immune response without human involvement.(Dal et al. 2008)

3.2.2 Unsupervised Machine Learning Algorithems

Unsupervised machine learning algorithms try to find hidden structure of unlabeled data. Clustering algorithms is one of the main approaches in unsupervised machine-learning techniques; it detects noises or abnormal behavior via categorizing patterns (data) into group/s (cluster) according to their resemblance (Nguyen & Armitage 2008, Bhuyan, Bhattacharyya & Kalita 2012).

The two main category of clustering algorithms are cluster association and centroid distance. In cluster association techniques such as DBSCAN, the clustered data will be considered as normal and the data point outside of clusters will be mark as abnormal (noise). Centroid distance based clustering techniques such as K-means, evaluate points based on their distance to their cluster centroid cluster coordinate hence small distance will be considered as normal and high distance as abnormal. In this section the two well-known relevant clustering algorithms (DBSCAN and K-means) for anomaly detection in network security are introduced.

3.2.2.1 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a powerful and well known unsupervised cluster association technique (Ester et al. 1996) which has been used in anomaly detection (Casas, Mazel & Owezarski 2012). It requires two parameters: maximum radius of the neighborhood (d) and minimum number of samples required to form a cluster (minSpl).

It starts with a random starting point, if the starting point contains sufficiently neighborhood a cluster will be created, otherwise, the point will labeled as noise, however, the noise point might later be found in density reachable with a different point and hence be made part of a cluster. All points that are found within the acceptable reachable neighborhood will be added to cluster. This process will continue until all the density connected points are found. Then, the algorithm will check a new unvisited point and apply all of the steps which may lead to discover a new cluster or label the point as noise. Figure 7 shows the steps which will be taken by DBSCAN algorithm for anomaly detection purposes.

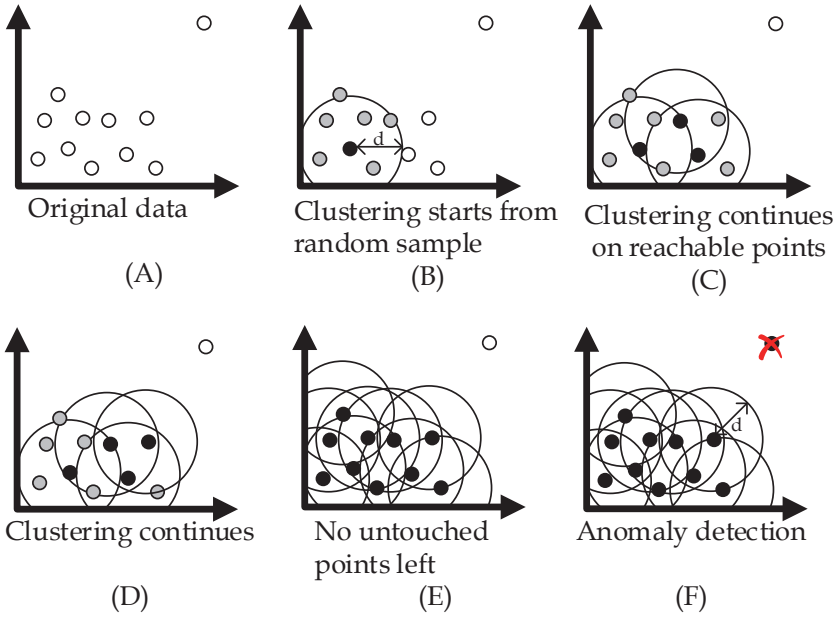


FIGURE 7 DBSCAN Clustering for Anomaly Detection.

DBSCAN algorithm appears to have a high detection rate of network intrusions without any prior knowledge as a result of the ability to cluster data in any size and arbitrary shape. In addition it will cluster the data without requiring knowing the number of clusters. (Erman, Arlitt & Mahanti 2006, Ghourabi, Abbes & Bouhoula 2010)

3.2.2.2 K-means

K-means is a well-known and widely used unsupervised centroid distance based clustering algorithm for anomaly detection. The given dataset will group into k clusters, in which each cluster has a cluster center (centroid). To increase the accuracy of data analyses, the algorithm tries to find optimal coordinates for centroid points which minimize their sum of distance to the clustered data points. In general k-means algorithm can be divided into these steps (Jain & Dubes 1988):

1. Randomly select the number of clusters (centroid) as k
2. Assign the original data to the nearest centroid for creating the clusters
3. Assign new coordinates for centroids based on the previously clustered data
4. Repeated step 2 and 3 until there is no changes in the coordinate of centroid points

Figure 8 shows the how k-mean clustering can be applied in anomaly detection (Münz, Li & Carle 2007). First, the original data groups in to k cluster. Then after finding the optimal centroid the data point with high distance will be flagged as abnormal.

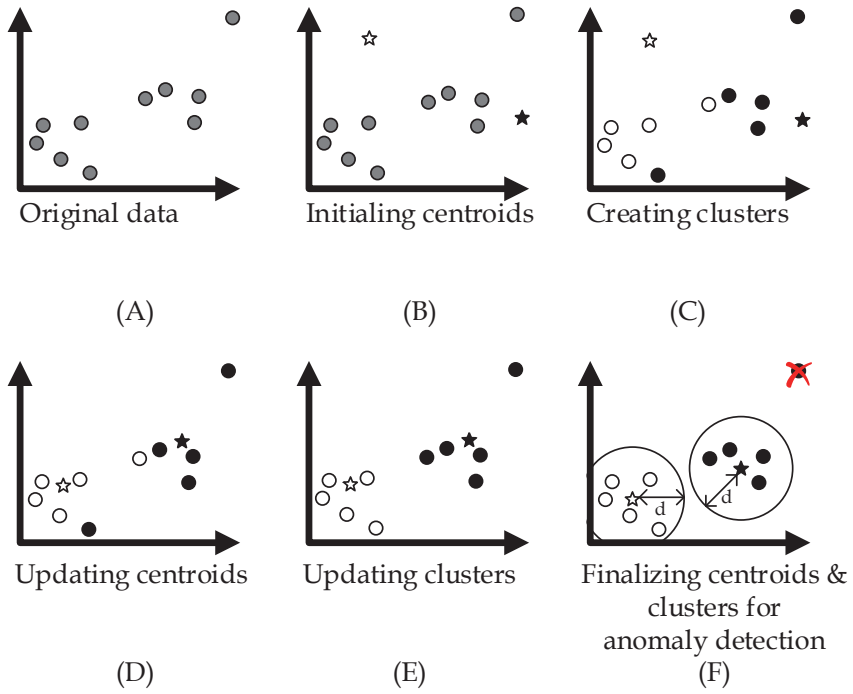


FIGURE 8 K-means Clustring for Anomaly Decetion.

3.3 Performance evaluation

Anomaly detection methods aim to find and mark abnormal behavior which derived from normal profiles (Patcha & Park 2007). In general there are three main metrics to evaluate an algorithm in anomaly detection area:

1. Prediction accuracy which refers to the ability of correctly modeling the data and predicting the class of new (unseen) data
2. Ability to make correct predictions in noisy, unbalance and missing data
3. The computation burden of data gathering and analyzing

In anomaly detection context "positive" class usually refers to anomalies data, whereas "negative" will be considered as normal instances. Table 1 shows the basic prediction conditions in anomaly detection area.

TABLE 1 Prediction Conditiones.

		Predicted Class	
		Normal Class	**Anomaly Class**
True Class	**Normal Class**	True Negative	False Positive
	Anomaly Class	False Negative	True Positive

Based on these terms the following performance metrics can be extracted:

- True Positive Rate (TPR) is the correctly classification ratio of the detected anomalous data to the total number of anomalous data
- True Negative Rate (TNR) is the correctly classification ratio of the detected normal data to the total number of normal data
- False Positive Rate (FPR) is the misclassified ratio of the detected normal data as anomalous to the total number normal data
- False Negative Rate (FNR) is the misclassified ratio of the detected anomalous data as normal to the total number anomalous data
- Accuracy is the ratio of total correctly classified data (true negative and true positive) to the total number of samples
- Recall is the ratio of correctly detected anomalies (true positive) to the total number of anomalies
- Precision is the ratio of correctly detected anomalies to the total number of predicted anomalies (true positive and false positive)

In general high false alarms (false positives or false negatives) can conclude the usability of anomaly detection algorithm.

3.3.1 Estimantion methodology

Estimation methodology is used to predict how the proposed application works. The proposed model should be tested on different data samples to extract the performance evaluation metrics and compared it with previously proposed model. Obviously the data which have been used for training the algorithm should not be used to estimate the performance of the model since it will result in not-realistic and over-optimistic performance prediction. Due to this reason different rules have been proposed to evaluate the machine learning algorithms.

K-fold is one of the well-known estimation methods which divide the total data sample to K equal size data subsets (Ramaswamy et al. 2001, Li, Zhang & Ogihara 2004). Each time one of the subsets will be used as the testing dataset and the rest (k-1) will be used as training samples. This process will be repeated for K times to use the entire K folds as the testing dataset for once. Finally, the k

cross validation estimates is averaged. Reducing the bias associated with the random sampling of the training samples is the main advantage of this method since for each time; the algorithm will be tested via separate portion of the data.

4 RESULTS

This chapter presents the research contribution and results which obtained in this dissertation. First the real time HIDS for botnet detection is presented. Next, the results and performance of supervised and unsupervised machine learning algorithms which applied in centralized and distributed NIDS are discussed. At the end, new unpublished method for finding anomalous outliers will be presented.

4.1 Real time HIDS for botnet detection

Article PI proposed a real-time approach which not only detects Botnet traffic on the host, but also can filter it from outgoing traffic in order to suppress the Botnet. The proposed approach works by detecting Botnet communication patterns which belongs to a centralized C&C structure in HTTP and IRC protocols. The capability of bot detection by real-time processing of host-related data solely, distinguishes this model from other existing approaches. We have implemented our detection approach within a packet filtering firewall for Windows XP machines (firewall drivers) to control the inbound and outbound connections. During testing phase against IRC bots such as Rx bot, results showed that suspicious IRC packets have been filtered out. Figure 9 shows the architecture overview of the proposed approach.

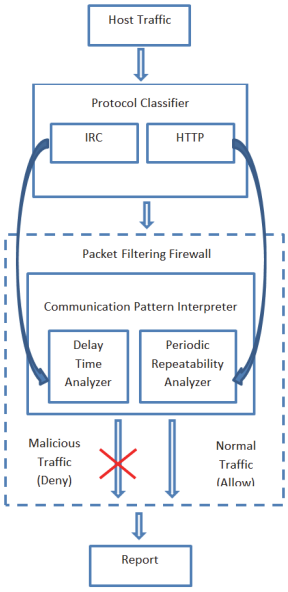


FIGURE 9 Architecture Overview of HIDS [PI].

4.2 Machine learning algorithms applied on NIDS

Human immune system (HIS) has decentralized architecture, which orchestrate its messages to the number of different types of cell to respond to the detected threats and repair the damaged tissue. Previously many researcher applied AIS on centralized machine which results in massive process in central engine (Dal et al. 2008, Dressler & Akan 2010). Article PIII introduced a new distributed, agent based AIS for intrusion detection. The main novelty of the proposed model is to distribute detectors in each host while the central engine manages the detectors to finalize decision about current abnormal behavior based on previous history. The main advantage of using AIS in distributed IDS is to benefited unique features of AIS such as self-learning, self-adaptation and self-improvement since detector agents in each host is actively updated and synchronized with detector agents of other hosts in the network through the central engine of IDS. Using memory cells in each host decrease the intrusion detection time for previously seen attacks since it contains the characteristics of the know attacks. However to add the capability of self-improvement in AIS, memory cells of newly detected anomalies by each host will be generated and sent to all hosts to synchronize them. The results after simulation show that numbers of memory cell detectors are dynamically increased and the framework is able to learn and detects new types of anomalies.

Previously, real time NIDS such as (Amini, Jalili & Shahriari 2006) used supervised machine learning algorithms to train their engine however the

acquisition of labeled data from security experts, or finding an attack-free data set are costly. In addition unsupervised NIDS such as (Casas, Mazel & Owezarski 2012) have high computation and it is not feasible to monitor the network online whereas real-time monitoring, processing and intrusion detection are now among the key features of NIDS. The proposed model in article PII which further developed in the article PV presents a new, real-time unsupervised NIDS, which detects zero-day attacks without any prior knowledge. Figure 10 shows the overall architecture of the implemented model.

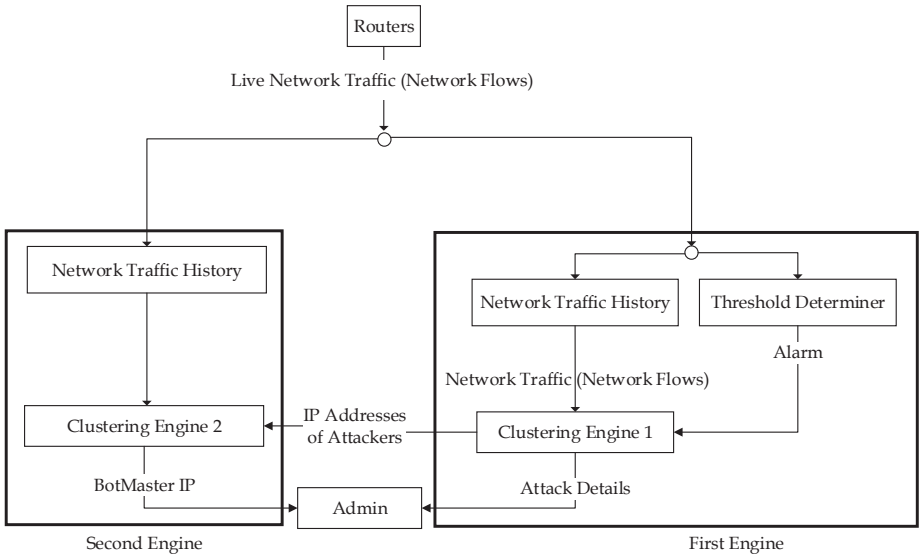


FIGURE 10 Architecture of the NIDS [PV].

The implemented model used a dynamic and self-adaptable threshold to detect unexpected behavior in the network to decrease the computation time of the clustering process during the normal state of the network. Standardizing data input via a logarithm (log) and monitoring the different size of subnets through the threshold increase the performance of the NIDS. In addition, dividing the process of intrusion detection by multistage engines decreases the computation time, which leads to having real-time intrusion detection for fast-spreading network attacks. Since, in the first engine, the DBSCAN trains itself with the previous clean network traffic; we reached a 100 per cent detection rate with 3.61 per cent of false alarms during our experiment. As a result of an increasing rate of DDOS attacks via the botnet, we implemented the second engine to trace the traffic of bots in order to detect the botmaster in centralized models under different protocols (HTTP and IRC). To evaluate the proposed model, we used two publicly available and well-known data sets to ensure the detection

process. Due to the unsupervised nature of the proposed model it will adapt to the structure of the data without training or previous knowledge. Since the data analyses of data will be done without any prior examples or attack signatures it may also detect zero-day (new) attacks.

Article PIV presented a novel architecture for an intrusion detection system based on the artificial immune system. As shown in Figure 11 the innate immunity will be done online via unsupervised machine learning methods.

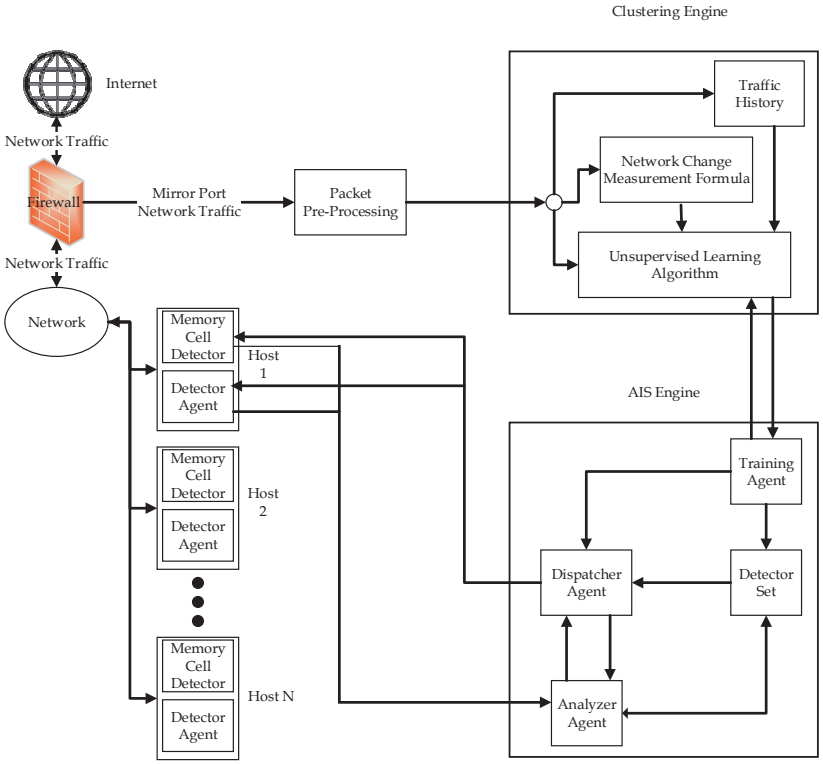


FIGURE 11 Proposed System Architecture [PIV].

In this multi-layered framework, the clustering engine labels the network traffic as self and non-self without any prior knowledge and previous training or knowledge about network flow profiles, thus acting as the first line of defense in AIS-based IDS to provide online innate immunity. The dynamic threshold has been used to facilitate the detection of abnormal network behaviors in the crusting engine. The output of clustering is used to feed the training data for the adaptive immune system as online and real-time training data. The primary detectors will be distributed to hosts in the network and provide primary immune response for the AIS based IDS. Based on the results the distributed structure for IDS is more efficient than the centralized mode. Suspected intrusions reported from hosts are analyzed and an optimized memory cell

detector is generated through a genetic algorithm process. Memory cells are attack specific detectors, which provide a secondary immune response. Detector life cycle rules update and eliminate weak or inefficient detectors to enhance the performance of detection. The main novelty of this framework is utilization of unsupervised machine learning methods in order to provide online and real-time training for the adaptive immune system within the artificial immune system without prior knowledge.

4.2.1 New unpublished results

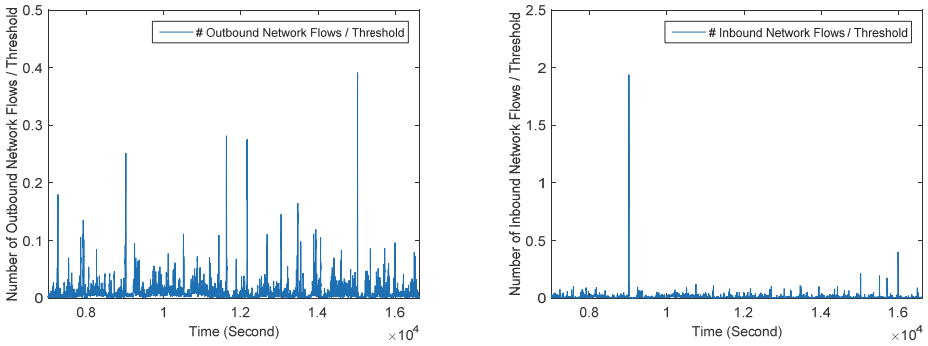
One of the future works proposed in article PV was to group users into different behavioral class. Real network contains traffic from different classes of users such as: normal users, busy users and servers. In general the number of busy users and servers is smaller than α thus they may not form a cluster in DBSCAN. Since the proposed model in article PV considers all of the network behavior (in the clean traffic windows) as normal it will increase the acceptable distance β for DBSCAN by high value of Δ to include all of the points inside the nearest cluster. Clustering data with high value of acceptable distance increase false negative rate (FNR) in certain cases. To overcome this issue we will propose a new method which compares the previous behavior of outliers to distinguish normal high traffic users from intrusions.

Similar to the proposed model in article PV whenever the volume of network flows passes the threshold, the NIDS uses the DBSCAN to cluster the number of in-bounded and out-bounded network flows for each machine to find the attacker/s. During the training phase, the NIDS clusters the clean network traffic transmitted before the threshold raised the alarm in order to obtain the most accurate distance during the detection phase. Technically the normal users will form into clusters while the density of busy users or servers may not reach the required level. Nevertheless, since training phase uses the clean network traffic the proposed model will consider outliers as busy machines with normal profiles.

Afterwards, to find the anomalous outliers which caused the high volume of network traffic, the detection engine clusters the suspicious network traffic window. The outliers IP addresses from detection phase will be compared to their previous profile. If the distance of current behaviors and previously seen behavior does not exceed the acceptable distance β, the detection engine will mark it as normal high traffic machine. However if the new behavior of outlier IP exceed the distance it will consider the behavior of that machine as abnormal. It important to note that if the outlier IP addresses do not have any profile from the training phase, the detection engine will mark it as abnormal.

To evaluate the new proposed model on detecting fast spreading attacks, we have used the same traffic sample as in article PV. Beside the DARPA traffic sample, we have tested our model on SSH Brute Force from ISCX. Since today most of the servers with SSH protocol limits number of user attempt, we have change the SSH Brute Force attack in ISCX to a distributed model which various number of bots participate in it. Figure 12 show the network's behavior during

the attack. As shown in Figure 12 (A) the ratio of outbound flows to the threshold is below one since the number of attackers is high however in Figure 12 (B) the threshold for inbounded traffic raise alarm since all of the traffic goes to the limited number of machines.



(A) (B)

FIGURE 12 Netwokrs Behavoiur During Distributed SSH Brute Force Attack.

Figure 13 shows the self-training phase during distributed SSH Brute Force attack. As mentioned before the NIDS marks IP address of machines which were located inside the clusters as normal. However the IP address of outliers will be profiled as busy users or servers. As shown in figure 14 during the comparison phase all of the outliers will be compared to their previous history. If the distance does not exceed the threshold, NIDS will mark them as normal users (with high traffic). Otherwise if the machine exceeds its traffic abnormally the NIDS will mark it as the abnormal machine. Figure 15 shows the final decision of NIDS.

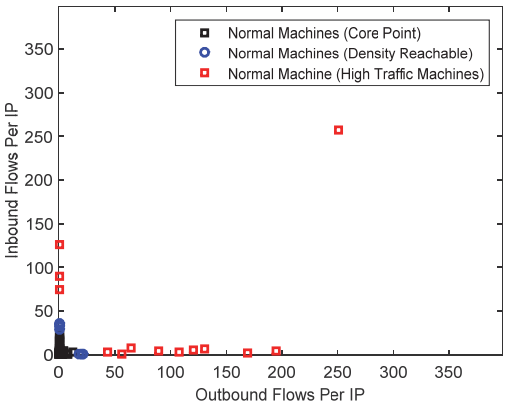


FIGURE 13 Self-Training Phase During Distributed SSH Brute Force Attack.

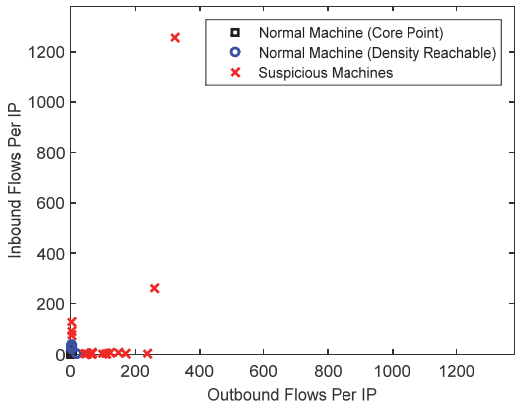


FIGURE 14 Comparison Phase During Distributed SSH Brute Force Attack.

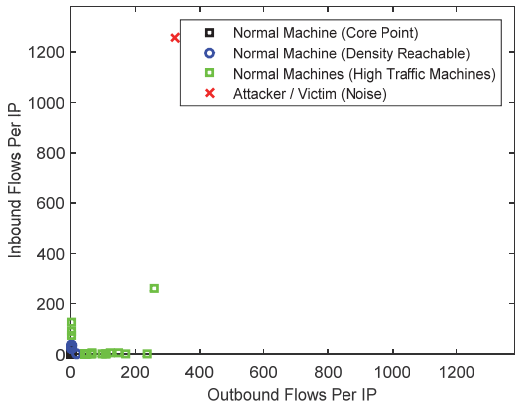


FIGURE 15 Detection Phase During Distributed SSH Brute Force Attack.

Table 2 shows the comparison of average performances of the new proposed model and article PV. To evaluate the performance of "different behavioral classes" feature in the new proposed model we have added traffic from busy users and servers during the occurrence of intrusion. Since the proposed model compares the behavior outliers to their previous history the overall performance was higher than the previous proposed model in article PV.

TABLE 2 Performance Evaluation.

	New proposed model	Proposed model in article PV
False positive rate	3.51%	4.53%
True negative rate	96.49	95.47%
Accuracy	98.35%	96.23%
Recall	100%	95.37%
Precision	97.83%	91.21%

5 CONCLUSION

Today, the occurrence of zero-day and complex attacks in high-speed networks is increasingly common due to the high number vulnerabilities in the cyber world. As a result, intrusions become more sophisticated and fast to detrimental the networks and hosts. Due to these reasons real-time monitoring, processing and intrusion detection are now among the key features of NIDS. Traditional types of intrusion detection systems such as signature base IDS are not able detect intrusions with new and complex strategies. Now days, automatic traffic analysis and anomaly intrusion detection became more efficient in field of network security however they suffer from high number of false alarms. In this dissertation, to tackle the above described problems several approaches have been applied. Due to unfeasibility of payloads checking in high-speed network, the proposed framework monitors network flows instead. Network flows contains the behavior of the network in higher extensive vision and shows the explicitness of the network data which results in faster and higher detection rate of network attacks.

Among all type of anomaly detection methods unsupervised machine-learning techniques are commonly applied in NIDS to detect unknown and complex attacks in the network without any prior knowledge. Unsupervised learning method suffers from two main drawbacks: high number of false alarm since it make the decision without any prior knowledge and high computational burden since it need to find the similarities and relation among all of the input data. To overcome computational burden we have applied automatic and adaptive threshold to minimize the required input data. In addition, to improve the accuracy of the clustering algorithm we have used the clean network traffic to train the engine. In this dissertation we also analyzed and compared the performance of centralized and decentralized AIS based NIDS. Due to the distribution of process and parallel learning capability in decentralized NIDS we achieved to have better detection rate in our proposed model. At the end we have added the capability of unsupervised learning for distributed AIS based NIDS to achieve online learning without any prior knowledge.

YHTEENVETO (FINNISH SUMMARY)

Tämän päivän nopeissa tiedonsiirtoverkoissa monimutkaisten ja nollapäivähyökkäysten yleisyys kasvaa kybermaailman haavoittuvuuksien suuren määrän vuoksi. Tämän seurauksena järjestelmiin tunkeutumiset muuttuvat hienostuneemmiksi ja nopeasti vahingollisiksi verkoille ja niiden laitteille. Täten reaaliaikainen monitorointi, prosessointi sekä tunkeilijan havainnointi ovat nykyään pääominaisuuksia verkkopohjaisissa tunkeilijan havaitsemisjärjestelmissä. Perinteiset tunkeilijan havaitsemisjärjestelmä, kuten allekirjoituspohjaiset tunkeilijan havaitsemisjärjestelmät, eivät kykene havaitsemaan uusia ja monimutkaisia strategioita omaavia tunkeutumisia. Tällä hetkellä verkkoturvallisuudessa automaattinen tietoliikenteen analysointi ja poikkeavien tunkeutumisten havainnointi ovat kehittyneet tehokkaammiksi, mutta ne kärsivät väärien hälytysten suuresta määrästä. Verkkopohjaisissa tunkeilijan havaitsemisjärjestelmissä käytetyt yleisimmät poikkeavuuksien havaintomenetelmät, joilla pyritään havaitsemaan tuntemattomia ja monimutkaisia hyökkäyksiä, ovat ei-ohjattuja koneoppimistekniikoita.

Tämä väitöskirja keskittyy pääosion verkkoliikenteen analysointiin poikkeavan liikenteen löytämiseksi reaaliajassa. Ehdotettu viitekehys koostuu verkkoliikenteen esikäsittelystä, poikkeamien havainnoimisesta sekä klusterointimenetelmistä. Esitetty viitekehys pystyy muodostamaan merkityksekkäitä raportteja etsittäessä todellisia haavoittuvuuksia tunnetusta datajoukosta. Ei-ohjatut oppimismenetelmät pystyvät mukauttamaan vaaditut ominaisuutensa verkon dynaamiseen käyttäytymiseen. Nopeissa tiedonsiirtoverkoissa pakettikohtainen tarkastaminen ei ole soveltuvaa, joten ehdotettu menetelmä havainnoi sen sijaan verkon pakettivirtoja. Pakettivirta sisältää verkon käyttäytymisen laajemmassa näkymässä ja näyttää selkeästi verkon datan, mikä johtaa verkkohyökkäysten nopeampaan ja luotettavampaan havaitsemiseen. Tämä tutkimus osoittaa että käyttämällä sopivaa datan esikäsittelyä sekä ei-ohjattuja datan analysointimenetelmiä on mahdollista havaita reaaliajassa nopeita ja monimutkaisia nollapäivänhyökkäyksiä.

REFERENCES

Abad, C., Taylor, J., Sengul, C., Yurcik, W., Zhou, Y. & Rowe, K. 2003. Log correlation for intrusion detection: A proof of concept. Computer Security Applications Conference. Computer Security Applications Conference, 2003. Proceedings. 19th Annual. IEEE, 255-264.

Amini, M., Jalili, R. & Shahriari, H. R. 2006. RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. Computers & Security 25(6), 459-468.

Asaka, M., Taguchi, A. & Goto, S. 1999. The implementation of ida: An intrusion detection agent system. Proceedings of the 11th FIRST Conference.

Augustin, M. & Balaz, A. 2011. Intrusion detection with early recognition of encrypted application. Intelligent Engineering Systems (INES), 2011 15th IEEE International Conference on. Poprad, Slovakia: IEEE, 245-247.

Balasubramaniyan, J. S., Garcia-Fernandez, J. O., Isacoff, D., Spafford, E. & Zamboni, D. 1998. An architecture for intrusion detection using autonomous agents. Computer Security Applications Conference, 1998. Proceedings. 14th Annual. IEEE, 13-24.

Bhuyan, M. H., Bhattacharyya, D. K. & Kalita, J. K. 2012. An effective unsupervised network anomaly detection method. Proceedings of the International Conference on Advances in Computing, Communications and Informatics. ACM, 533-539.

Boult, T. E. & Brown, L. G. 1991. Factorization-based segmentation of motions. Visual Motion, 1991., Proceedings of the IEEE Workshop on. Princeton, NJ, USA: IEEE, 179-186.

Brouer, J. 2014. Mitigate TCP SYN Flood Attacks with Red Hat Enterprise Linux 7 Beta. Available in: http://rhelblog.redhat.com/2014/04/11/mitigate-tcp-syn-flood-attacks-with-red-hat-enterprise-linux-7-beta/. Accessed: 20.Sep.2015.

Casas, P., Mazel, J. & Owezarski, P. 2012. Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. Computer Communications 35(7), 772-783.

Chandola, V., Banerjee, A. & Kumar, V. 2009. Anomaly detection: A survey. ACM computing surveys (CSUR) 41(3),15, 1-58.

Chapelle, O., Scholkopf, B. & Zien, A. 2006. Semi-Supervised Learning. The MIT Press.

Chiu, Y. T., Liu, S. T., Huang, H. C. & Hong, K. F. 2015. Discovering Potential Victims Within Enterprise Network via Link Analysis Method. Current Approaches in Applied Artificial Intelligence , 326-335.

Cid, D. 2015. WordPress Brute Force Attacks – 2015 Threat Landscape. Available in: https://blog.sucuri.net/2015/09/wordpress-brute-force-attacks-2015-threat-landscape.html. Accessed: 01.Oct.2015.

Cid, D. 2014. Layer 7 DDOS – Blocking HTTP Flood Attacks. Available in: https://blog.sucuri.net/2014/02/layer-7-ddos-blocking-http-flood-attacks.html. Accessed: 10.Aug.2015.

Claise, B. 2008. Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information. Available in: http://tools.ietf.org/html/rfc5101. Accessed: 01.Sep.2015.

Cole, E. 2012. Advanced persistent threat: understanding the danger and how to protect your organization. Newnes.

Dal, D., Abraham, S., Abraham, A., Sanyal, S. & Sanglikar, M. 2008. Evolution Induced Secondary Immunity: An Artificial Immune System Based Intrusion Detection System. Computer Information Systems and Industrial Management Applications, 2008. CISIM '08. 7th. Ostrava, Czech Republic: IEEE, 65-70.

De Castro, L. N. & Timmis, J. 2002. Artificial immune systems: a new computational intelligence approach. Springer Science & Business Media.

De Castro, L. N. & Von Zuben, F. J. 2002. Learning and optimization using the clonal selection principle. Evolutionary Computation, IEEE Transactions on 6(3), 239-251.

Denning, D. E. 1987. An Intrusion-Detection Model. Software Engineering, IEEE Transactions on 13(2), 222-232.

Dressler, F. & Akan, O. B. 2010. A survey on bio-inspired networking. Computer Networks 54(6), 881-900.

Durumeric, Z., Wustrow, E. & Halderman, J. A. 2013. ZMap: Fast Internet-wide Scanning and Its Security Applications. Proceedings of the 22nd USENIX Security Symposium. , 605-620.

Elhamifar, E. & Vidal, R. 2010. Clustering disjoint subspaces via sparse representation. Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on. Dallas, TX, USA: IEEE, 1926-1929.

Elhamifar, E. & Vidal, R. 2009. Sparse subspace clustering. Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. Miami, FL, USA: IEEE, 2790-2797.

Engen, V. 2010. Machine learning for network based intrusion detection: an investigation into discrepancies in findings with the KDD cup '99 data set and multi-objective evolution of neural network classifier ensembles from imbalanced data. Bournemouth University.

Erman, J., Arlitt, M. & Mahanti, A. 2006. Traffic classification using clustering algorithms. Proceedings of the 2006 SIGCOMM workshop on Mining network data. ACM, 281-286.

Esponda, F. 2005. Negative representations of information. University of New Mexico.

Ester, M., Kriegel, H. P., Sander, J. & Xu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. Knowledge Discovery and Data Mining (KDD-96). Proceedings of the Second International Conference on. Portland, USA: AAAI Press, 226-231.

Estevez-Tapiador, J. M., Garcia-Teodoro, P. & Diaz-Verdejo, J. E. 2003. Stochastic protocol modeling for anomaly based network intrusion detection. Information Assurance, 2003. IWIAS 2003. Proceedings. First IEEE International Workshop on. IEEE, 3-12.

Fayyad, U. M., Piatetsky-Shapiro, G. & Smyth, P. 1996. Knowledge Discovery and Data Mining: Towards a Unifying Framework. KDD 96, 82-88.

Forrest, S., Perelson, A. S., Allen, L. & Cherukuri, R. 1994. Self-nonself discrimination in a computer. Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on. Oakland, CA: IEEE, 202-212.

Freitas, A. A. & Timmis, J. 2007. Revisiting the Foundations of Artificial Immune Systems for Data Mining. Evolutionary Computation, IEEE Transactions on 11(4), 521-540.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G. & Vázquez, E. 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security 28(1), 18-28.

Ghorbani, A. A., Lu, W. & Tavallaee, M. 2009. Network Intrusion Detection and Prevention: Concepts and Techniques. (1st edition) Springer Publishing Company, Incorporated.

Ghourabi, A., Abbes, T. & Bouhoula, A. 2010. Data analyzer based on data mining for Honeypot Router. Computer Systems and Applications (AICCSA), 2010 IEEE/ACS International Conference on. Hammamet, Tunisia: IEEE, 1-6.

Goh, A. & Vidal, R. 2007. Segmenting Motions of Different Types by Unsupervised Manifold Clustering. Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on. Minneapolis, MN, USA: IEEE, 1-6.

Guyon, I. & Elisseeff, A. 2003. An introduction to variable and feature selection. The Journal of Machine Learning Research 3, 1157-1182.

Hand, D. J., Mannila, H. & Smyth, P. 2001. Principles of data mining. MIT Press, Adaptive computation and machine learning .

Hernández-Pereira, E., Suárez-Romero, J. A., Fontenla-Romero, O. & Alonso-Betanzos, A. 2009. Conversion methods for symbolic features: A comparison applied to an intrusion detection problem. Expert Systems with Applications 36(7), 10612-10617.

Hofmeyr, S. A. & Forrest, S. 2000. Architecture for an artificial immune system. Evolutionary computation 8(4), 443-473.

Hsu, C. F. & Hung, H. F. 2009. Classification Methods of Credit Rating - A Comparative Analysis on SVM, MDA and RST. Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on. Wuhan, China: IEEE, 1-4.

Huang, S. Y. & Huang, Y. N. 2013. Network traffic anomaly detection based on growing hierarchical SOM. Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on. Budapest, Hungary: IEEE, 1-2.

Jackson, W. 2013. Microsoft issues fix for resurrected Ping of Death. Available in: https://gcn.com/Blogs/CyberEye/2013/08/Microsoft-patch-ping-of-death-IPv6.aspx. Accessed: 05.Sep.2012.

Jain, A. K. & Dubes, R. C. 1988. Algorithms for clustering data. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

James, G., Witten, D., Hastie, T. & Tibshirani, R. 2014. An Introduction to Statistical Learning: With Applications in R.

Kesavaraj, G. & Sukumaran, S. 2013. A study on classification techniques in data mining. Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on. Tiruchengode: IEEE, 1-7.

Koch, R. & Rodosek, G. D. 2010. Command Evaluation in Encrypted Remote Sessions. Network and System Security (NSS), 2010 4th International Conference on. IEEE, 299-305.

Komninos, N., Vergados, D. & Douligeris, C. 2010. Security for ad hoc networks. Handbook of Information and Communication Security. , 421-432.

Kotsiantis, S. & Pintelas, P. 2004. Recent advances in clustering: A brief survey. WSEAS Transactions on Information Science and Applications 1(1), 73-81.

Kotsiantis, S. B., Zaharakis, I. & Pintelas, P. 2007. Supervised machine learning: A review of classification techniques. Emerging Artificial Intelligence Applications in Computer Engineering , 3-24.

Kruegel, C., Mutz, D., Robertson, W. & Valeur, F. 2003. Bayesian event classification for intrusion detection. Computer Security Applications Conference, 2003. Proceedings. 19th Annual. IEEE, 14-23.

Kruegel, C., Vigna, G. & Robertson, W. 2005. A multi-model approach to the detection of web-based attacks. Computer Networks 48 (5), 717-738.

Kumar, G., Kumar, K. & Sachdeva, M. 2010. The use of artificial intelligence based techniques for intrusion detection: a review. Artificial Intelligence Review 34(4), 369-387.

Lakhina, A., Crovella, M. & Diot, C. 2004. Characterization of network-wide anomalies in traffic flows. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. ACM, 201–206.

Leydesdorff, L. & Bensman, S. 2006. Classification and powerlaws: The logarithmic transformation. Journal of the American Society for Information Science and Technology 57(11), 1470-1486.

Li, L. & Kianmehr, K. 2012. Internet traffic classification based on associative classifiers. Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2012 IEEE International Conference on. Bangkok, Thailand: IEEE, 263-268.

Li, T., Zhang, C. & Ogihara, M. 2004. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. Bioinformatics 20(15), 2429-2437.

Li, W. 2004. Using genetic algorithm for network intrusion detection. Proceedings of the United States Department of Energy Cyber Security Group. , 1-8.

Liao, Y. & Vemuri, V. R. 2002. Use of k-nearest neighbor classifier for intrusion detection. Computers & Security 21(5), 439-448.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K. & Zissman, M. A. 2000. Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings. , 12-26.

Liu, H. & Yu, L. 2005. Toward integrating feature selection algorithms for classification and clustering. Knowledge and Data Engineering, IEEE Transactions on 17(4), 491-502.

Lyon, G. F. 2009. Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. USA: Insecure.

Mark, A. L., Crovella, M. & Diot, C. 2004. Characterization of Network-Wide Anomalies in Traffic. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. ACM, 201-206.

Molina, L. C., Belanche, L. & Nebot, A. 2002. Feature selection algorithms: a survey and experimental evaluation. Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on. , 306-313.

Münz, G., Li, S. & Carle, G. 2007. Traffic anomaly detection using k-means clustering. GI/ITG Workshop MMBnet.

Nguyen, T. T. T. & Armitage, G. 2008. A survey of techniques for internet traffic classification using machine learning. Communications Surveys & Tutorials 10(4), 56-76.

Novakov, S., Lung, C. H., Lambadaris, I. & Seddigh, N. 2013. Studies in applying PCA and wavelet algorithms for network traffic anomaly detection. High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference on. , 185-190.

Patcha, A. & Park, J. 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer networks 51(12), 3448-3470.

Peng, T., Leckie, C. & Ramamohanarao, K. 2004. Proactively Detecting Distributed Denial of Service Attacks Using Source IP Address Monitoring. Networking , 771-782.

Piirainen, K. A. & Gonzalez, R. A. 2013. Seeking constructive synergy: design science and the constructive research approach. Design Science at the Intersection of Physical and Virtual Design , 59-72.

Ramadas, M., Ostermann, S. & Tjaden, B. 2003. Detecting anomalous network traffic with self-organizing maps. Recent Advances in Intrusion Detection , 36-54.

Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C. H., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J. P., Poggio, T., Gerald, W., Loda, M., Lander, E. S. & Golub, T. R. 2001. Multiclass cancer diagnosis using tumor gene expression signatures. Proceedings of the National Academy of Sciences. , 15149-15154.

Retnakaran, N. & Pizzi, N. J. 2005. Biomedical pattern classification using an optimized fuzzy adaptive logic network. Electrical and Computer Engineering, 2005. Canadian Conference on. Saskatoon, Canada: IEEE, 382-385.

Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A. & Stiller, B. 2010. An Overview of IP Flow-Based Intrusion Detection. Communications Surveys & Tutorials, IEEE 12(3), 343-356.

Tankard, C. 2011. Advanced Persistent threats and how to monitor and deter them. Network security 8, 16-19.

Tedesco, G. & Aickelin, U. 2006. An Immune Inspired Network Intrusion Detection System Utilising Correlation Context. AISB '06: adaptation in artificial and biological systems. Society for the Study of Artificial Intelligence and the Simulation of Behaviour. Bristol, UK: , 16-17.

Vidal, R. 2010. A tutorial on subspace clustering. (IEEE Signal Processing Magazine 28(2) edition) IEEE.

Virvilis, N. & Gritzalis, D. 2013. The Big Four - What We Did Wrong in Advanced Persistent Threat Detection? Availability, Reliability and Security (ARES), 2013 Eighth International Conference on. IEEE, 248-254.

Xin, J., Dickerson, J. E. & Dickerson, J. 2003. Fuzzy feature extraction and visualization for intrusion detection. Fuzzy Systems, 2003. FUZZ'03. The 12th IEEE International Conference on. IEEE, 1249-1254.

Zhang, Q. & Chang, S. 2009. An Improved Crossover Operator of Genetic Algorithm. Computational Intelligence and Design, 2009. ISCID '09. Second International Symposium on. , 82-86.

PI

REAL-TIME BOTNET COMMAND AND CONTROL CHARACTERIZATION AT THE HOST LEVEL

by

Farhood Farid Etemad and Payam Vahdani Amoli 2012

Telecommunications (IST), 2012 Sixth International Symposium on, pp. 1005-1009, Tehran, Iran

6'th International Symposium on Telecommunications (IST'2012)

Real-Time Botnet Command and Control Characterization at the Host Level

Farhood Farid Etemad
Department of Computer engineering
Engineering Faculty
Ferdowsi University of Mashhad
Mashhad, Iran
fa.faridetemad@gmail.com

Payam Vahdani
Dept. of Mathematical Information Technology
Faculty of Information Technology
University of Jyväskylä
Jyvaskyla, Finland
pavahdan@student.jyu.fi

Abstract— **A Botnet is a network of compromised machines which are controlled by a person called botmaster via a typical Command and Control (C&C) structure. Besides malicious activity on infected host, bots are employed to deliver attacks against outside targets including phishing, Distributed Denial of Service (DDoS) attacks and spamming. Counter measures against Botnet phenomenon are usually formed based on passive traffic analysis at network level. This limits encountering Botnets in a proactive manner. In this paper, we proposed a real-time approach which not only detects Botnet traffic on the host, but also can filter it from outgoing traffic in order to suppress the Botnet. Our approach works by detecting Botnet communication patterns which belongs to a centralized C&C structure. The capability of bot detection by real-time processing of host-related data solely, distinguishes our work from other existing approaches.**

Keywords- Centralized C&C; Botnet; real-time; detection; Host-Based;

I. INTRODUCTION

Botnet is a large group of compromised machines which are remotely controlled by a person or group of persons called botmaster. They are currently the biggest security threat to the Cyber world [1]. The main difference of Botnet and other kind of malwares is the presence of a Command and Control (C&C) mechanism through which botmaster issues commands to the compromised machines (also called zombie) to employ them for various kind of attacks [26]. An attacker who also called BotHerder controls Bots through different protocols and structures. Based on this, Botnets C&C mechanisms are mainly categorized into centralized and decentralized. The early samples of Botnets were using centralized C&C structures with communication protocols like IRC and HTTP. Recently new type of bots has been emerged that are using decentralized structure as the C&C structure e.g. peer to peer communication protocols, though bots with centralized C&C models are still active and prevalent[1].

Botnet can cause many problems mainly include launching Distributed Denial of Service (DDOS) attacks against various web servers or service providers, sending spam e-mails to other hosts, hosting malicious phishing sites and using by botmaster for the purposes of click fraud. Besides centralized attacks against external targets, Botnets compromise the infected machine and usually install backdoors and various types of malware on their host. Current counter measures against Botnet are mainly focused on monitoring and analysis of passive network traffic to detect Botnet[30]. Botnet detection mechanisms monitor passive network traffic captured from network switches or routers, looking for suspicious behavior or signature patterns based on the similarities between network flows. Once a match found, detection mechanism issues a warning. To suppress the Botnet, command and control server which counts as a central point of communication between zombies and botmaster must be tracked and shut down. Current detection approaches mainly operate at the network level and formed based on passive network traffic analysis.

With powerful and advanced capabilities, it is very difficult for average users to avoid or prevent infection by Botnet malware. Based on this fact, infection is irresistible and that makes the role of Botnet detection approaches, which are operating at the host level more significant. Network based detection approaches do not resolve infected hosts or even notify them of infection. Moreover, with network based techniques it is impossible to trace Botnet executables on the infected machine and investigate zombies in order to capture evidence, which is helpful in studying Botnet behavior. Stopping Botnet by shutting down the command and control channel can solve the problem temporary but hosts will stay infected and compromised in the way that they can be exploited easily in future attacks. Moreover, botmasters usually use alternative C&C server addresses for their bots to set up a Botnet again. To improve previous works on the field of Botnet detection, a real-time detection algorithm is required, which be able to detect bots on the host to manage the disinfection process and filter out malicious traffic to suppress the Botnet.

In this paper we proposed a novel host-based approach that recognizes a host infected by the bot based on analyzing the host inbound/outbound traffic. We process host network traffic to infer the existence of bot C&C communication and upon detection, malicious outgoing traffic can be filtered out actively. This forms a proactive approach which can stop bots before delivering their payload or taking part in attacks.

The rest of the text is organized as follows. In section 2, we review the previous related work. In section 3, the proposed detection approach and its components is described and finally conclude in section 4.

978-1-4673-2073-3/12/$31.00 ©2012 IEEE

II. Related Work

Various works have been done regarding detection of Botnet. There are mainly two major approaches to detect Botnet. One approach forms based on locating honeynets in the network and the other is based on monitoring and analysis of passive network traffic [30].

Anyhow, honey net detection approaches [24] are more useful to study Botnet characteristics rather than detection. Passive network traffic monitoring can be categorized into signature-based, anomaly-based, DNS-based and mining-based [27].

Signature based detection approaches look for matching predefined signatures in the network related data or host related data. Anomaly based approaches which are the majority of Botnet detection techniques, look for similar activities among captured netflows to detect an anomaly which can be the sign of Botnet existence. DNS based approaches try to detect the Botnet by analyzing DNS data which exchanged between bot and Command and Control (C&C) servers [30]. Like anomaly based approaches, these mechanisms also analyze captured network traffic passively and cannot detect bot in real-time manner. Mining-based approaches focus on processing log files to infer an abnormal behavior which is the sign of Botnet existence [9].

Majority of previous works regarding Botnet detection and prevention, are operating at the network level. Only few works has been done to detect bots at the host level. Zeng *et al.* [29] in 2010, proposed a model which combined host-based and network-based methods together to detect Botnet, independent of the command and control (C&C) mechanisms used by Botnet. This approach applied host memory data and network flows to detect Botnet though, still performed detection passively.

Strayer *et al.* [23] propose a detection approach which examines how characteristics such as bandwidth, packet timing, and burst duration determine the existence of Botnet command and control activity. Schiller *et al.* [22] in 2007 suggested investigating the infected host by scrutinizing event and firewall logs to determine the payload and functions of the Bot. They also suggested looking for suspicious start-up processes so as to identify the location of the malware. Goebel and Holz [9] proposed *Rishi* in 2007. Rishi is primarily based on passive traffic monitoring for odd or suspicious IRC nicknames, IRC servers, and unusual server ports. They use n-gram analysis and a scoring system to detect bots that use uncommon communication channels.

Most researchers have proposed investigating Botnet at the network scope; anyhow, this neglects the importance and potential advantages of examining an infected host at the local scope. Obviously, network-based investigation forms based on communication protocol information obtained from Bot-infected machines. This highlights the significance of host-based investigation and the fact that these two approaches are in direct relevance.

III. Proposed Botnet Detection Approach and Components

Our proposed detection approach is based on real-time analysis of host's inbound/outbound traffic to infer existence of centralized Botnet C&C communication model. Figure 1 illustrates architecture of the proposed detection system which comprises of two main components: Protocol Classifier and Communication Pattern Interpreter. The latter component consists of two components itself: IRC Part and HTTP Part. At the first step, the whole host outgoing/incoming traffic is redirected to the Protocol Classifier. This component separates IRC and HTTP packets from the rest of traffic and forwards it on to the next component. In Communication Pattern Interpreter, IRC Part is responsible for detecting IRC malicious traffic based on IRC bots communication model with C&C server. HTTP Part is also responsible to recognize HTTP-based Botnet C&C communication pattern based on Periodic Repeatability of messages. Output of Communication Pattern Interpreter is malicious traffic distinguished from normal traffic which can be filtered by means of a packet filtering firewall on the host.

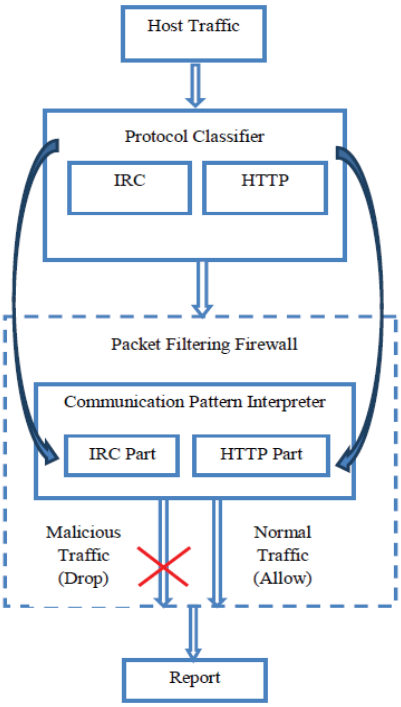


Figure 1. Architecture overview of our proposed approach

A. Protocol Classifier

Since our detection approach is formed based on characterization of bot's C&C traffic, we need to separate traffic which is more likely used in C&C communication protocols at the first place. To manage that, the main function of Protocol Classifier is to separate IRC and HTTP (which are currently the most common protocols used in centralized C&C [28]) packets from the rest of traffic and forward them to Communication Pattern Interpreter component.

To detect IRC traffic, we can inspect the contents of the packets to look for some predefined strings which are actually keywords in IRC protocol (defined in RFC1459). For this purpose, light payload inspection would be enough to look for specific IRC strings including NICK for client's nickname, PASS for user's password, USER for the username, JOIN for joining a channel, PRVIMSG for private messages, OPER for when a normal user wants to become the channel operator and MOTD which returns message of the day[21]. This method of detecting IRC traffic can be accomplished by using intrusion detection software like SNORT[35]. In some cases botmasters encrypt the IRC communication traffic to evade detection mechanisms which is not our aim here.

To recognize HTTP traffic, we also inspect the early bytes of a packet looking for some patterns and keywords in an http request message. To detect HTTP traffic, we need to focus on the concepts of HTTP protocol. HTTP is a protocol which works on the basis of client-server model. According to this, a client initiates a connection and sends a *HTTP Request* message to an HTTP server (e.g. "get me the file 'website.html'"). The server then processes the client's request and responds to it via an *HTTP Response* message (e.g. "here is the file" followed by the file's contents). After that, the server closes the connection; make the HTTP a *stateless* protocol which does not maintains the connection information between transactions [33]. In order to characterize HTTP traffic on the host's outgoing traffic, we focus on the HTTP method. Three main HTTP methods are "GET", "POST" and "HEAD" [33]. Hence, all we need is to look for "GET", "POST" or "HEAD" keywords in the contents of the packets. Like IRC traffic detection, this also can be done by inspecting the first few bytes of network packets. Upon recognition of HTTP packets, these flows are also forwarded to the Communication Pattern Interpreter for further processing.

B. Communication Pattern Interpreter

When communication between zombies and botmaster through C&C happens, certain patterns can be seen. Previous works regarding characterizing such patterns, mainly formed based on analysis of network flow characteristics related to group of hosts. Unlikely, our approach solely examines a host inbound/outbound traffic to determine such patterns. This gives the opportunity of detecting bots in a real-time manner and filtering the malicious traffic.

Communication Pattern Interpreter constitutes the major component of our approach and is responsible for detecting bot's malicious traffic by identifying Botnet C&C structure. It comprises of two modules for characterizing IRC and HTTP malicious traffic which belongs to bot. we first explain characteristics of a typical IRC and HTTP Command and Control structure and the way they differ from normal IRC and HTTP traffic. Then based on these model characteristics, we explain how IRC Part and HTTP Part operate.

Gu *et al.* [32] categorize centralized C&C models into two categories: "Pull" style and "Push" style, based on the way Bot receives command from botmaster. In a push style C&C model, bots are waiting to receive commands from botmaster during a persistent connection. IRC-based C&C is an example of this type in which bots are waiting in the channel for botmaster to issue a command. In the pull style C&C, botmaster set the commands within a file at a C&C server (e.g. a HTTP server) and bots connect back to read the commands from this file. So in this style, there is no need for a persistent active connection between bots and botmaster. HTTP-based C&C is a distinct example of this style [32].

C. IRC Part

Based on our experiments during traffic analysis of various IRC bots (e.g. RBot, Agobot, etc), the communication life cycle of an IRC bot can be divided into two phases, as shown in Figure 2. Phase 1 indicates the time period before the bot joining an IRC channel and phase 2 is the time period after the bot joining a channel. As it can be seen in Figure 2, during phase 1 it is the bot who issues IRC messages. Though, after connecting to the IRC server and joining a channel, bot only receives commands from botmaster, i.e. botmaster is the initiator rather than the bot. In the case of no botmaster presents at the channel or no command issues, bot would stay dormant at the channel. This is a characteristic which is not seen in normal IRC chat message sequence (except for times packet errors happen which entails retransmission). Consequently, if a client sends no IRC message after joining a channel, it can be a bot.

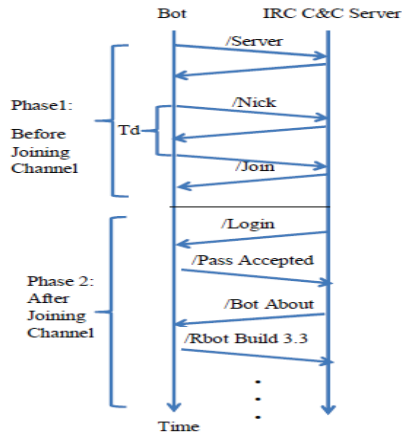


Figure 2. Two phases of IRC bot communication life cycle

We focus on detection of malicious IRC traffic based on this communication model. The key is to detect the transition

point between two phases. As it can be seen in Figure 2, two consecutive commands, issued from botmaster is a key to infer the transition point. On the infected host, this is interpreted as the last two consecutive incoming IRC packets, after which normal command-response model proceeds. These two consecutive commands may differ in various versions of IRC bots, although this deviation is slight. Normally, based on our experiments, in most IRC bots they are "JOIN" and "LOGIN" commands.

Other experimental factors can be added to IRC Part module to increase the detection accuracy. Here we introduce a characteristic based on packets timing and the delay between issuing commands. Delay time (Td) is defined as a time frame between the time user or bot connects to server and time it tries to join a channel. That is to say a time frame between sending "Nick" command and "Join" command:

Td=T(Nick) – T(Join)

Since bots are automated programs, they join channel immediately after connecting to the server (they are running programs) while a real IRC user needs to use command line or graphical user interface in order to accomplish IRC commands (i.e. joining a channel). So the delay time for bot would be too short (less than a second) while for IRC normal user it takes at least few seconds. Based on several experiments, it has perceived that a threshold value can be assigned to Td and it is called "Tdh". If "Td" exceeds "Tdh" it is the proof of normal IRC communication:

If Td > Tdh : Normal IRC communication

If Td < Tdh : Malicious IRC communication

Analyzing Delay Time to detect IRC bots works properly in most cases at normal conditions, though it can be designed by bots to mimic human behavior. So we consider it a supplementary method rather than a standard approach.

D. HTTP Part

Figure 3 depicts a typical HTTP-based C&C communication pattern. As it can be seen, there is a regular periodic trend in times which bot connects back to the C&C server checking for commands. This repeatability for one specific flow (same source and destination port) does not take place in normal Http communication [33]. Although in some cases this periodical pattern is not seen in HTTP bot communications, it counts as a good characteristic to detect HTTP malicious traffic [32]. We use this characteristic in Periodic Repeatability Analyzer to distinguish between HTTP malicious and normal traffic.

Lee *et al.* [33] in 2008 defined degree of periodic repeatability and repeatability standard deviation to describe relationship between HTTP hosts and HTTP servers. These two criterions can also be used for detecting HTTP malicious traffic of a Botnet. The repeatability standard deviation demonstrates degree of periodic repeatability between HTTP clients and HTTP servers [33]. Based on periodic characteristic of HTTP bots in communication with C&C server, it is concluded that the degree of periodic repeatability of bot machines is quite lower than normal users. Degree of periodic repeatability of normal users is calculated highly because the intervals between pollings in normal users are not regular.

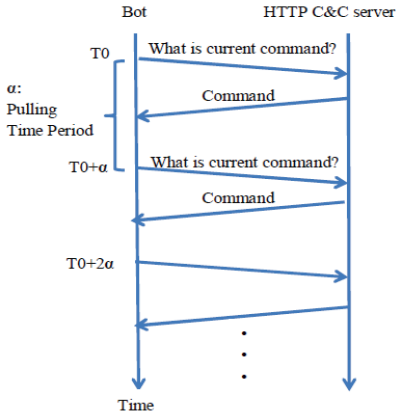


Figure 3. HTTP bot C&C traffic pattern

Finally, it should be mentioned that in Botnets with several botmasters which bots can connect to them randomly, Periodic Repeatability criterion may not suffice to detect HTTP-based C&C structure.

E. Real-Time Filtering

By means of a packet filtering firewall on the host machines, we can manage filtering malicious traffic after detection, as long as our detection mechanism works in real-time manner. Presence of filtering mechanism is optional and we can simply report the bot existence to the host without trying to filter out the bot traffic.

We have implemented our detection approach within a packet filtering firewall for Windows XP machines. After testing it against various IRC bots including Rx bot, results showed that suspicious IRC packets have been filtered out.

IV. CONCLUSION

Botnets are new generation of sophisticated malwares which are more difficult to trace, detect and shut down, in comparison with other types of malwares. Few works has been done to actively detect and block Botnets traffic on the infected hosts. This feature, distinguishes our approach from the previous works done in the field of Botnet detection. In this paper, we proposed an approach for online detection of Botnet traffic on the infected host. To do this, we inspect the host traffic for signs of Botnet C&C communication patterns. Although our work here is limited to centralized Botnet C&C models, a component for detection of peer to peer bots on the host also can be added to our detection mechanism. Since we solely use host related traffic, there would be no main challenge to recognize peer to peer C&C models as well.

REFERENCES

[1] Bailey, M., Cooke, E., Jahanian, F., Xu, Y., Karir, M.,"A Survey of Botnet Technology and Defenses," In Proceedings of the Cybersecurity Applications & Technology Conference For Homeland Security (CATCH '09), Washington, District of Columbia, USA, March 2009,pp. 299-304.

[2] Barford, P., Yagneswaran, V.,"An Inside Look at Botnets.," In: Special Workshop on Malware Detection, Advances in Information Security, Springer, Heidelberg, 2006.

[3] Binkley, J. R. and Singh, S.,"An Algorithm for Anomaly-Based Botnet Detection.," Proceedings of 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06), July 2006,pp. 43-48.

[4] Cooke E., Jahanian F. and McPherson D., "The zombie roundup: Understanding, detecting and disrupting Botnets.," In Proceedings of UsenixWorkshop on Stepts to Reducing Unwanted Traffic on the Internet (SRUTI '05), Cambridge, MA, July 2005.

[5] Crimeware: Bots. Web publication (2010, August), Available: http://www.symantec.com/norton/cybercrime/Bots.jsp

[6] Law, F.Y.W.,Chaw, K.P., Lai, P.K.Y.and TseH.K.S., "A Host Based Approach to Botnet Investigation?," Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, ICDF2C 2009, LNICST 31, 2010, pp. 161-170.

[7] Freiling, F., Holz, T., and Wicherski, G.,"Botnet Tracking: Exploring a Root- Cause Methodology to Prevent Distributed Denial-of-Service Attacks.,"In 10thEuropean Symposium on Research in Computer Security, 2005.

[8] Gianvecchio, S., Xie, M., Wu, Z. and Wang, H ,"Measurement and classification of huamans and bots in internet chat.," In Proceedings of the 17th USENIX Security Symposium (Security' 08), San Jose, CA, 2008.

[9] Goebel , J. and Holz, T. "Rishi: identify bot contaminated hosts by IRCnickname evaluation,"HotBots'07 Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, USENIX Association Berkeley, CA, USA, 2007,pp. 8-8.

[10] Gu, G., Porras P., Yegneswaran V., Fong M., and Lee W.,"Bothunter:Detecting Malware Infection Through Ids-Driven Dialog Correlation,"Proceedings of the 16th USENIX Security Symposium, Berkeley, CA, USA, 2007, pp. 1-16.

[11] Gu, G., Perdisci, R., Zhang, J., and Lee, W.,"BotMiner: Clustering analysis ofnetwork traffic for protocol- and structure-independent Botnet detection,"Proceedings of the 17th USENIX Security Symposium, 2008.

[12] Honeynet Project. (2005). Know your Enemy: Tracking Botnet,http://www.honeynet.org/papers/Bots

[13] Ianelli N. and A. Hackworth.(2005). Botnet as a Vehicle for Online Crime, CERT.

[14] Oliva, J., "An Adventure: How to implement a Firewall-Hook Driver?," Thecode projects development Reources, 2004.

[15] Karagiannis, T., Papagiannaki, K. and Faloutsos, M.,"BLINC:multileveltraffic classification in the dark," In Proceedings of the Conference onApplications, Technologies, Architectures, and Protocols for ComputerCommunications, 2005, pp. 229-240.

[16] Karasaridis A. Rexroad B. and Hoein D., "Widescale Botnet Detection andCharacterization," Proceedings of Hot Topics in Understanding Botnet, 2007.

[17] Leder, F., Werner, T. and Martini P.,"Proactive Botnet Countermeasures – AnOffensive Approach," Cooperative Cyber Defence Bonn, Germany, 2010.

[18] Livadas, C., Walsh, R., Lapsley, D. and Strayer. W. T.,"Using machinelearning techniques to identify Botnet traffic," In Proceedings of the 2nd IEEELCN Workshop on Network Security, 2006.

[19] Puri R., Bots & Botnet: An Overview. Research on Topics in InformationSecurity, 2003.

[20] Ramachandran, A. and Feamster, N.,"Understanding the network-levelbehavior of spammers," In Proc. ACM SIGCOMM, 2006.

[21] Rayome, J.,"IRC on Your Dime? What You Really Need to Kno aboutInternet Relay Chat," CIAC/LLNL, 1998.

[22] Schiller, C., Binkley, J., Evron, G. and Willems, C.,"Botnet – The killer webapp," Syngress.179–208, 2007.

[23] Strayer, W. T., Lapsley, D., Walsh, R., and Livadas, C.,"Botnet detectionbased on network behavior," In Botnet Detection: Countering the LargestSecurity Threat, W. Lee, C. Wang, and D. Dagon, Eds., vol. 36 of Advances inInformation Security. Springer, 2008, pp. 1-24.

[24] Vrable, M., Ma, J. , Chen, J. , Moore, D. , Vandekieft, E., Snoeren, A. C. , Voelker,G.M. and Savage, S.,"Scalability, Fidelity and Containment in thePotemkin Virtual Honeyfarm," In Proc. ACM SIGOPS OperatingSystemReview, vol. 39(5), 2005, pp. 148–162.

[25] Zhang, L., Yu, S., Wu, D., Watters, P., "A Survey on Latest Botnet Attack and Defense," IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Changsha, China, 2012, pp. 53 - 60.

[26] Xiabo, M. A., Guan, X., Tao, J., Zheng, Q., Guo, Y., Liu, L., Zhao, S.,"ANovel IRC Botnet Detection Method Based on Packet Size Sequence," IEEEInternational Conference on Communication (ICC), 2010,pp.1-5.

[27] Zeidanloo, H. R. and Manaf, A. B. A.,"Botnet Detection by MonitoringSimilar Communication Patterns," (IJCSIS) International Journal of ComputerScience and Information Security, Vol. 7, No. 3, 2010.

[28] Raghava, N.S., Sahgal. D. And Chandna, S., "Classification of Botnet Detection Based on Botnet Architechture," International Conference on Communication Systems and Network Technologies (CSNT, 2012), IEEE, May, Rajkot, India, 2012, pp.569 - 572.

[29] Zeng, Y. , Hu, X. , Shin, K.G., "Detection of Botnet Using Combined HostandNetwork-Level Information," IEEEIIFIP International Conference onDependable Systems & Networks (DSN), 2010.

[30] Zhu, Z., Lu, G., Chen, Y. , Fu, Z. J., Roberts, P. and Han, K., "BotnetResearch Survey," in Proc. 32nd Annual IEEE International Conference onComputer Software and Applications (COMPSAC '08), 2008, 2008, pp.967-972.

[31] Zhuge, J., Holz, T., Han, X., Guo, J. and Zou, W., "Characterizing the ircbasedBotnet phenomenon," Peking University & University of MannheimTechnical Report, 2007.

[32] Gu, G., Zhang, J.,And Lee, W., "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic", In Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08), San Diego, CA, February 2008.

[33] Lee, J.S., Jeong, H.C.,Park, J.H., Kim, M., Noh, B.N.,"The Activity Analysis of Malicious HTTP-based Botnets using Degree of Periodic Repeatability," International Conference on Security Technology, SECTECH '08, Hainan Island, China, 2008, pp. 83 – 86.

[34] Leder, F., Werner, T. And Martini, P., "Proactive Botnet Countermeasures – An Offensive Approach," Proc. of 1st CCDCOE Conference on Cyber Warfare, Tallinn, Estonia, 2009.

[35] Snort IDS web page. http://www.snort.org, March 2006.

PII

A REAL TIME UNSUPERVISED NIDS FOR DETECTING UNKNOWN AND ENCRYPTED NETWORK ATTACKS IN HIGH SPEED NETWORK

by

Payam Vahdani Amoli and Timo Hämäläinen 2013

Measurements and Networking Proceedings (M&N), 2013 IEEE International Workshop on, pp. 149-154, Naples, Italy

A Real Time Unsupervised NIDS for Detecting Unknown and Encrypted Network Attacks in High Speed Network

Payam Vahdani Amoli
Student Member, IEEE
Department of Mathematical Information Technology
Faculty of Information Technology, Jyväskylä University
Jyväskylä, Finland
pavahdan@student.jyu.fi

Timo Hämäläinen
Department of Mathematical Information Technology
Faculty of Information Technology, Jyväskylä University
Jyväskylä, Finland
timo.t.hamalainen@jyu.fi

***Abstract*— Previously, Network Intrusion Detection Systems (NIDS) detected intrusions by comparing the behaviour of the network to the pre-defined rules or pre-observed network traffic, which was expensive in terms of both cost and time. Unsupervised machine learning techniques have overcome these issues and can detect unknown and complex attacks within normal or encrypted communication without any prior knowledge. NIDS monitors bytes, packets and network flow to detect intrusions. It is nearly impossible to monitor the payload of all packets in a high-speed network. On the other hand, the content of packets does not have sufficient information to detect a complex attack. Since the rate of attacks within encrypted communication is increasing and the content of encrypted packets is not accessible to NIDS, it has been suggested to monitor network flows. As most network intrusions spread within the network very quickly, in this paper we will propose a new real-time unsupervised NIDS for detecting new and complex attacks within normal and encrypted communications. To achieve having a real-time NIDS, the proposed model should capture live network traffic from different sensors and analyse specific metrics such as number of bytes, packets, network flows, and the time explicitly and implicitly, of packets and network flows, in the different resolutions. The NIDS will flag the time slot as an anomaly if any of those metrics passes the threshold, and it will send the time slot to the first engine. The first engine clusters different layers and dimensions of the network's behaviour and correlates the outliers to purge the intrusions from normal traffic. Detecting network attacks, which produce a huge amount of network traffic (e.g. DOS, DDOS, scanning) was the aim of proposing the first engine. Analysing statistics of network flows increases the feasibility of detecting intrusions within encrypted communications. The aim of proposing the second engine is to conduct a deeper analysis and correlate the traffic and behaviour of Bots (current attackers) during DDOS attacks to find the Bot-Master.**

Index Terms*— NIDS, Unsupervised Intelligent Engine, Encrypted Network Traffic, Network Flows, Clustering

I. Introduction

Nowadays, because of increasing occurrences of network intrusions, NIDS has become an important element within networks. Generally NIDS monitors the behaviour of networks and detects attacks when an abnormality occurs. Network attacks can cause a big latency inside the network by producing a huge amount of network traffic; thus, having a real-time NIDS is also an important factor.

Signature-based NIDSs monitor the behaviour of the network and compare it with the characteristics of known network attacks. The detection rate of known attacks in signature-based NIDS is high; however, it cannot detect zero-day attacks. Providing attack signatures consumes money and time, and with the increasing rate of zero-day attacks, using signature-based NIDS is not a safe solution. In anomaly-based detection techniques, the system will be trained by a sample of network traffic and adapted to the state of the network. After the training phase, the system will be suspicious of any abnormal behaviour that passes the criteria of the training sample. Using this method will increase the probability of detecting novel attacks; however, it makes lots of detection errors because of the difficulty of defining the normal state during training. Having fewer false alarms and an increased detection rate of complex attacks, especially in imbalanced network traffic, has become an important challenge in the design of detection techniques for NIDS. [1, 2]

NIDS monitors the behaviour of networks by analysing bytes, packets or network flows. Based on our previous experiments and other researchers, monitoring network flows enhances the detection rate of complex attacks [1, 3, 4, 5, 6, 7]. Analysing bytes or packets does not concede sufficient information about the current behaviour of machines within the network. Network flows store the important facts about the behaviour of the network, which is clearer for NIDS to analyse. On the other hand, according to [1], network flows require 0.1 per cent of storage to be saved compared to data in the form of packets. Summarising the network data in the form of network flow enhances the speed of processing, which results in fast attack detection, and makes NIDS feasible to work in real time. For instance, in high-speed networks with an average rate of ten gigabytes per second, it is impossible for NIDS to check the content of each packet in real time. Furthermore, detecting network intrusions in real time is an important factor for NIDS, which is why the use of

network flows as the input for NIDS is suggested. According to [8]: "A flow is defined as a set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties."

According to [9, 10, 11], with the growing number of attacks in encrypted communication, it has also become an important issue to detect these types of attack while a limited amount of information can be extracted from the encrypted traffic. Monitoring encrypted communication in the form of network flows enables the system to monitor the state and transitions of communication to detect attacks such as DOS or brute force, in addition to the types of attack that produce large network traffic in encrypted communication.

Currently, detecting complex attacks is one of the issues for NIDS. Since probabilistic approaches to NIDS rely only on statistics and do not correlate alarms, the rate of false alarms increases during complex attacks. On the other hand, scenario-based NIDS need to observe specific steps to detect attacks, and since complex attacks do not follow constant steps it is not suggested to apply it to NIDS [12]. According to [13], machine learning techniques have been used in anomaly-based NIDS and enhanced the performance of attack detection. Self-learning abilities in machine learning techniques improve the detection rate of new, complex and encrypted intrusions [14, 15].

Supervised machine learning algorithms need to be trained by a labelled data set in order to produce functions for distinguishing the normal and abnormal behaviour of the network. Semi-supervised machine learning algorithms can be trained by an attack-free unlabelled data set to formulate the normal behaviour of the network, or by a small labelled data set, which requires less effort from security experts. Nevertheless, the acquisition of labelled data from security experts, or finding an attack-free data set for both supervised and semi-supervised techniques are costly. Unsupervised machine learning techniques formulate the invisible structure of an unlabelled data set without any supervision. Clustering algorithms put objects based on their similarities into a group or groups, called clusters. Clustering algorithms have been used for unsupervised NIDS to classify the behaviour of the network and distinguish the abnormal behaviour of the network from normal traffic. [16, 17]

In this paper we propose a new real-time unsupervised NIDS, which can work in normal or encrypted communications by monitoring the behaviour of network flows in two different window sizes and detect attacks by correlating outliers from the multiple clusters. The first engine has the ability to detect different types of intrusion in real-time, such as DOS, DDOS, scanning or any other type of network attack that produces a huge amount of network traffic. At the same time, the characteristics of encrypted network flows will be analysed in order to detect intrusions within encrypted traffic. Based on our previous works [18, 19, 20, 21] and other researchers [22, 23], detecting Botnet attacks through checking the network flow only (without checking the payload of the packet) takes longer and, because of the complex structure of Botnet attacks, the NIDS needs more time to observe sufficient information. The second engine correlates the traffic of attackers (while the victim is under distributed attack) to find the similarities inside previous communications to find the eventual Bot-Master.

II. RELATED WORKS

In [24] they observed the behaviours of the network by monitoring and analysing the network flows. Using network flows as input for the proposed solution reduces the computation complexity and requires fewer resources. On the other hand, they have improved the detection rate and decreased false alarms compared to the previous solutions, which analyse packets as the input for NIDS. Several solutions applied sampling to decrease the computation time for NIDS [1]; however, sampling network traffic based on random selection increases the probability of losing important data, which leads the NIDS to produce a high false-negative error rate.

Multi-stage engines have been applied to NIDS to improve the detection rate of attack within the network [12, 25]. Analysing the behaviors of the network in several phases filters the unrequired data, improves the quality of input for NIDS, highlights the suspicious behaviour of the network and decreases the computation time for intelligent engines. In [25] they proposed multi-stage engines to filter suspicious network flows in the first stage and to send them for further analyses in the second stage. The proposed solution was not applicable for high-speed networks as the window size of the first engine was only 60 seconds and it could store and analyse 10 network flows at the same time.

Because of cost and time-consuming solutions for creating the attack signature of misuse-based NIDSs or traffic sample for anomaly-based NIDSs, several researchers applied unsupervised machine learning algorithms to NIDS. In [26] they proposed a real-time unsupervised NIDS to detect known and unknown network attacks using neural networks. They applied several neural networks to improve the detection rate of intrusions. In [27] they also proposed an unsupervised NIDS, which uses different clustering algorithms with a high detection rate.

Several solutions were proposed in [9] to detect intrusions inside encrypted communications. As suggested in [9] and proposed in [10, 11], clustering algorithms allow the NIDS to distinguish the behaviour of networks based on statistics. Analysing network flows provides sufficient information to detect intrusion within encrypted communications. Clustering network flows is the feasible solution, for encrypted communication while the payloads of packets is not accessible to the NIDS.

There is a specific behavioural structure in communications between the Bot-Master and Bots in Botnet attacks. Based on our previous research [21] and others [22, 23], Botnet attacks can be detected by finding the similarities between Bots and the Bot-Master. For instance, Bots regularly ping their Bot-Master to report their current status. Analysing and clustering the behaviour of Bots highlights those similarities (between other Bots) and also uniqueness (compared to other machines in the network).

III. PROPOSED SOLUTION

One of the main goals in this proposed solution is to design a real-time NIDS. To achieve this goal, several packet sniffers should be installed inside the network to aggregate the traffic and send it to the NIDS.

Figure 1 shows the overall architecture of the proposed NIDS. After traffic aggregation, all of the duplicated packets will be filtered and synchronised based on their time stamp. Special features from the network's behaviour (based on Table 1) will be sent to the first engine for further analyses; in the meantime, the past hour of network traffic is stored in the database to increase the accessibility of the second engine to the previous actions inside the network, while the system needs to trace the Bot-Master during Botnet attacks. As explained below, the main reason for proposing the first engine is to detect intrusions with a small windows size and the second engine for detecting Botnet attacks.

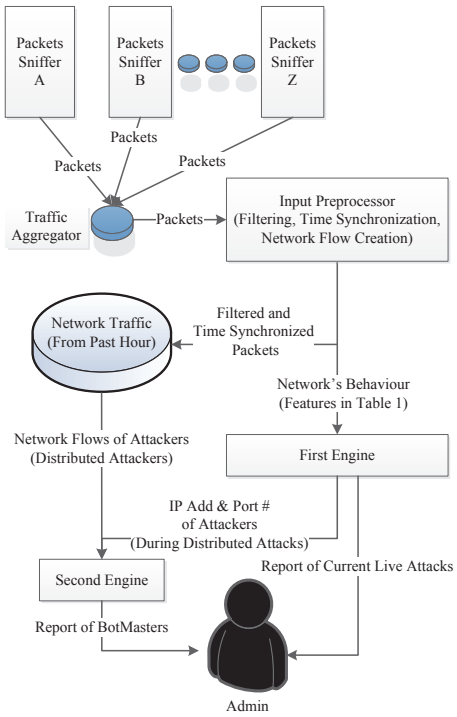


Figure 1. High-level structure of the NIDS

a) First engine:

Network attacks produce a huge amount of traffic. Analysing the volume of three different metrics such as bytes, packets and network flows can highlight suspicious activity. We have applied *network change measurement formula,* which works based on time-series analyses to monitor those features [28].

As mentioned in [28] this mechanism can monitor one million records per second. It is a fast and reliable solution to check the behaviour of the network and it can detect any small changes in real time. The proposed mechanism is unsupervised (does not need any assumption) and uses less memory than previous solutions. If any of the features in Table 1 passes the threshold of network change measurement formula, the system will flag that specific time slot as an anomaly.

As shown in Table 1, the network features will be analysed in four different resolutions: the whole network traffic and three small subnets (/0, /8, /16 and /24). High-speed networks have vast amounts of traffic and there is a significant possibility of losing the signs of network attacks. According to [27], network changes will be more visible while the NIDS monitors the network's behaviour in small resolution and decreases the probability of fading of attacks in normal traffic. On the other hand, according to [26], due to having an increased rate of DOS attacks (or any other type of network attack) with spoofed IP addresses, the direct use of IP addresses is not suitable and will increase the rate of false-negative alarms.

Apart from monitoring and analysing the volume of bytes, packets and network flows in different resolutions, it is also suggested to monitor the *time implicitly* of packets and network flows in small resolution. In [26] the detection rate of intrusions was enhanced by 2 per cent, while the system examines the traffic by *time implicitly* of the network element. Since network attacks produce a vast amount of packets or network flows, the rate of time difference between each packet (RTDP) and network flow (RTDF) will increase significantly and monitoring these parameters will improve the detection rate.

TABLE 1 – MONITORED FEATURES

Resolution	***Feature***
/0, /8, /16, /24	Number of in-bounded byte
/0, /8, /16, /24	Number of out-bounded byte
/0, /8, /16, /24	Number of in-bounded packet
/0, /8, /16, /24	Number of out-bounded packet
/0, /8, /16, /24	Number of in-bounded network flow
/0, /8, /16, /24	Number of out-bounded network flow
/24	Rate of Time Difference between each Packet (RTDP) $RTDP = \frac{\text{Windows Size (Seconds)}}{\text{Time Difference between each Packet (Seconds)}}$
/24	Rate of Time Difference between each Flow (RTDF) $RTDF = \frac{\text{Windows Size (Seconds)}}{\text{Time Diff. between each Net. Flow (Seconds)}}$

While any feature from Table 1 passes the threshold of network change measurement mechanism, it will flag that time slot as an anomaly. Then the system will extract more information from the packets to finalise the structure of the network flow. As Table 2 shows, several features will be selected to create the network flow based on the protocol of the communication (IP, TCP, UDP and ICMP). According to [29], it is suggested to define the default value of inactive network flow as 15 seconds and the default value of active timeout as 30 minutes. Extracting all the information from

packets and converting it into network flows takes resources (time and process). Extracting information to complete the structure of the network flow after network change measurement mechanism decreases the computation process while the network is not under attack.

While the NIDS detects an anomaly-flagged time slot, it will cluster the traffic to distinguish normal traffic from suspicious actions. Clustering algorithms do not require any prior knowledge and they can work unsupervised. However, each clustering algorithm has its own strengths and limitations and it is impossible to find one suitable algorithm for detecting all types of intrusions within the network. To overcome this issue we have applied multi-clustering algorithms to enhance the rate of detection and decrease the error rate.

During attack, several features of network flows (Table 2) should be clustered. Clustering high-dimensional data is not suggested since the computation process will be so complex and takes a long time to process. To resolve this problem, high-dimensional data can be divided into smaller dimensions by Sub-Space clustering algorithms. Figure 2 shows an example of Sub-Space clustering, which divides the three-dimensional data set into 3 two-dimensional data sets.

TABLE 2 – NETWORK FLOW SPECIFICATION FOR EACH TYPE OF PACKET

Packets Protocol	***Features***
IP	Source IP Address, Destination IP Address, Time of the First Packet, Time of the Last Packet, Duration
TCP	Source Port Number, Destination Port Number, Number of Packets, Number of SYN Packet, Number of SYN-ACK packet, Number of RST Packet, Number of RST-ACK Packet, Number of FIN-ACK Packet, Average Size of Packet from Source, Average Size of Packet from Destination, Biggest Packet Size, Smallest Packet Size, Time of Last packet from Source, Time of last Packet From Destination, Average latency of packets from Source, Average latency of packets from Destination
UDP	Source Port Number, Destination Port Number, Number of Packets, Average Packet Size, Biggest Packet Size, Smallest Packet Size
ICMP	Average Packet Size from Source, Average Packet Size from Destination, Biggest Packet Size, Smallest Packet Size, #Eco Request, #Eco Reply

The next stage is to cluster all the two-dimensional data (from an anomaly-flagged time slot) with DBSCAN (density-based spatial clustering of applications with noise). DBSCAN [30] is a powerful density-based clustering algorithm, which can create clusters in any arbitrary shapes and sizes. DBSCAN will cluster the events to distinguish the normal traffic from suspicious actions and it will designate the outliers.

DBSCAN works with two important parameters: minimum size of cluster (α); and acceptable distance between each point (β). The stated solution in [27] sets 'α' as 5 per cent of the network flows (number of network flows during the attack) and 'β' as the average distance (Euclidean distance) between 10 per cent of network flows (randomly selected) during the attack (from the anomaly-flagged time slot). As the detection rate in their proposed model is so high we decided to use the same amount of information; however, setting the required parameters for DBSCAN ('α', 'β') from the anomaly-flagged time slot is not suggested. According to [31], the Mahalanobis distance considers the density of points for measuring the distance between points, whoever, It is suggested to use the Mahalanobis distance while the system needs to set 'β' for the DBSCAN. Using the Mahalanobis distance for DBSCAN enhances the accuracy rate of clusters.

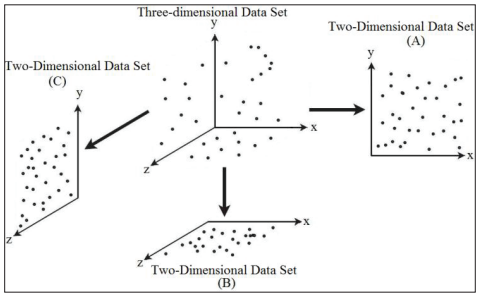


Figure 2. Three-dimensional Data Set divided into 3 Two-Dimensional Data Sets by Sub-Space Clustering Algorithem (A,B,C)

Clearly, during an attack, the network is loaded with a huge amount of objects, which belong to the intrusion (packets and network flows). Sampling data to set the parameters of the intelligent engine from the anomaly-flagged time slot decreases the probability of assigning an accurate value for those parameters, which leads the system to generate a high number of false alarms and a decreased detection rate.

As the proposed model saves the past hour of network traffic, it is possible to effectuate the sampling process from the previous time slot (the time slot before the anomaly-flagged time slot). The NIDS will set 'α' by counting 5 per cent of network flows and 'β' by calculating the Mahalanobis distance between 10 per cent of network flows from attack-free time slot. As the density rate of the network's behaviour does not follow any specific rule, it is important to consider this factor while the system needs to measure distances between elements within the network traffic.

While the features inside Table 2 are clustered by DBSCAN, the outliers will be marked as suspicious elements. The final step is to correlate all of the suspicious network flows to highlight the similarities and repost it to administrator to identify the type of attack.

Most of the proposed models (for instance [27]) check the number of SYN packets and label the intrusion as 'SYN flood attack' whenever the rate of SYN packets becomes high. However, according to [24] it is more accurate to find the current abnormal situation of TCP and ICMP connections using the features in Table 3. Comparison of the features in Table 3 from previous attack-free time slot and suspicious time slot will generate more accurate and clearer report to administrator for identifying the attacks. For instance, while the number of SYN packets is high and labelled as outlier during clustering, the system will calculate the rate of AHS in normal and anomaly-flagged network traffic. Whenever the differences between AHS rate in normal and anomaly-

flagged time slot becomes high, the system will suggest high probability of SYN flood attack to administrator.

TABLE 3 – MONITORED FEATURES

Resolution	*Feature*
/0	Rate of Accepted Hand Shake (AHS) $$AHS = \frac{\#\text{Syn} - \#\text{SynAck}}{\#\text{Syn}}$$
	Rate of Syn-Ack Arrival (SAA) $$SAA = \frac{\#\text{SynAck} - \#\text{Rst}}{\#\text{Syn}}$$
	Rate of Non-Ack Arrival (NAA) $$NAA = \frac{\#\text{SynAck} - \#\text{RstAck}}{\#\text{SynAck}}$$
	Rate of Successful Closed Connection (SCC) $$\text{SCC} = \frac{\#\text{FinAck} - \#\text{SynAck}}{\#\text{SynAck}}$$
	Rate of ICMP-Echo-Request to all of the Packets (ICP)
	Rate of ICMP-Echo-Reply to all of the Packets (IRP)

Clustering the different features of network flow enhances the detection rate of intrusions within encrypted communications. For instance, as mentioned in [11], encryption does not make any significant changes in the size, number and arrival time of packets. Clustering the behaviour of the network (encrypted and normal traffic) allows the NIDS to detect any significant changes in the network and it can correlate evidence to identify the attack type.

b) Second engine:

Whenever the first engine identifies DDOS attack, it will send the details of the attackers to the second engine. The second engine is responsible for clustering the behaviour of attackers and finding similarities in the previous communication in order to find the potential Bot-Master.

As proposed in [22, 23], one of the suitable solutions for finding the Bot-Master is to aggregate the traffic of Bots and correlate their communication before the distributed attack (DDOS, spam senders or other distributed types of attack) to determine the similarities. The Bot-Master communicates with the Bots in a particular way. For instance, the Bot-Master will define a rule for Bots to send Ping requests to notify the Bot-Master about their current status. Clustering the previous traffic of current attackers allows the NIDS to find the similarities in their communication and increases the probability of detecting the Bot-Master.

Based on our previous research [21] and others [23], one of the common methods of the Bot-Master and Bots is to use IRC protocol for their communication. In normal communication the time differences between IRC requests and replies is several seconds, while during a Botnet attack Bots reply to the request of the Bot-Master extremely quickly. Humans (normal users) need to open interface programs to work with IRC, and their response time is great while they are communicating with a server (e.g. entering simple commands by a normal user through the keyboard takes at least 2 or 3 seconds). Instead, the automated program, which is installed on Bot, will reply to the request of the Bot-Master in milliseconds. Clustering the response time of Bots and comparing it with the average length of response time in normal traffic will highlight the differences.

During distributed attacks, whenever the second engine finds similarities from the previous communication of Bots with any particular machines (Bot-Master), the system will generate a report and send it to the administrator.

IV. CONCLUSIONS AND FUTURE WORKS

The main goal of the proposed technique is to have real-time and unsupervised NIDS. We have found the weaknesses and limitations of current unsupervised NIDS (for instance [27]) and apply new features (calculating the Mahalanobis distance for DBSCAN, Botnet detection engine etc.) to enhance the detection rate of network intrusions. To decrease the computation burden, the system will monitor the volume of traffic. In the event of facing significant changes in the volume of network traffic (which can be caused by network attacks), the network measurement formula will trigger the system to start the detection process. Dividing the process of detection using multi-stage engines decreases the load of the computation process. Clustering network traffic distinguishes normal traffic from outliers and detects attacks inside normal or encrypted communications. The first engine will detect attacks using a small attack-window size, such as DOS, DDOS and Scanning. It has been discovered [21, 22, 23] that the Bot-Master communicates with Bots in a same way. During distributed attacks (such as DDOS) the second engine will cluster and merge the previous actions of distributed attackers (suspicious Bots) to find the similarities of their previous connections to detect the suspicious Bot-Master.

We will implement our proposed model and test it with NSL-KDD [32] and ISCX [33] traffic sample to demonstrate the improved rate of intrusion detection on different types of network attacks such as DOS, DDOS, Botnet and etc. In the meanwhile we will create several networks (and sub networks) in a simulated environment, such as NS3 (Network Simulator 3) and connect it to public network and install our proposed model on the gateway. Afterward, we will simulate different types of network attacks to check and demonstrate the realtimeness of the proposed model. We are also undertaking research in order to propose a dynamic self-tuning mechanism to suggest a more optimal size of 'α' and 'β' for the DBSCAN clustering algorithm in order to increase the rate of intrusion detection.

ACKNOWLEDGMENTS

We wish to acknowledge the sponsorships of CIMO (Centre for International Mobility) in Helsinki, Finland and COMAS (Doctoral Program in Computing and Mathematical Sciences) by the University of Jyväskylä, Finland which have made it possible to undertake this research.

REFERENCES

[1] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, B. Stiller, "An overview of IP flow-based intrusion detection," Communications Surveys & Tutorials, IEEE , vol.12, no.3, pp.343-356, Third Quarter 2010

[2] V. Engen, "Machine learning for network based intrusion detection: an investigation into discrepancies in findings with the KDD cup '99 data set and multi-objective evolution of neural network classifier ensembles from imbalanced data," PhD Thesis, Bournemouth University, 2010

[3] P. Vahdani Amoli, A.R. Ghobadi, G. Taherzadeh, R. Karimi, S. Maham, "New Detection Technique Using Correlation of Network

Flows For NIDS," Proceedings of the 2011 International Conference on Security Management, SAM 2011, Las Vegas, Nevada, USA, 2011

[4] A. Lakhina, M. Crovella, C. Diot, "Characterization of network-wide anomalies in traffic flows," Proc. of the 4th ACM SIGCOMM conference on Internet measurement, pp.201–206, ACM, New York, 2004

[5] G. Tedesco, U. Aickelin, "An Immune Inspired Network Intrusion Detection System Utilising Correlation Context," Proceedings of the Workshop on Artificial Immune Systems and Immune System Modelling (AISB '06), Bristol, 2006

[6] T. Peng, C. Leckie, K. Ramamohanarao, "Proactively Detecting Distributed Denial of Service Attacks Using Source IP Address Monitoring," Proceedings of the Third International IFIP-TC6 Networking Conference (Networking 2004), pp.771-782, 2004

[7] A.L. Mark, M. Crovella, C. Diot, "Characterization of Network-Wide Anomalies in Traffic Flows," IMC '04 Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, pp.201-206, New York, NY, USA, 2004

[8] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101 (Proposed Standard), [Online]. Available: http://www.ietf.org/rfc/rfc5101.txt, Jan. 2012

[9] R. Koch, G.D. Rodosek, "Security System for Encrypted Environments (S2E2)," RAID 2010, LNCS, vol. 6306, pp.505-507, Springer, Heidelberg, 2010

[10] R. Koch, G.D. Rodosek , "Command Evaluation in Encrypted Remote Sessions," *Network and System Security (NSS), 2010 4th International Conference on* , vol., no., pp.299-305, 1-3 Sept. 2010

[11] M. Augustin, A. Balaz, "Intrusion detection with early recognition of encrypted application," *Intelligent Engineering Systems (INES), 2011 15th IEEE International Conference on* , vol., no., pp.245-247, 23-25 June 2011

[12] F. Alserhani, M. Akhlaq, I.U. Awan, A.J. Cullen, P. Mirchandani, "MARS: Multi-stage Attack Recognition System," Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on , vol., no., pp.753-759, 20-23 April 2010

[13] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," Computers & Security, vol. 28, Issues 1–2, pp. 18-28, February–March 2009

[14] M.N.M. Sap, A.H. Abdullah, S. Srinoy, S. Chimphle, W. Chimphle, "Anomaly Intrusion Detection Using Fuzzy Clustering Methods," Jurnal Teknologi Maklumat, FSKSM, UTM, Jurnal Teknologi Maklumat, vol.18, pp.25-32, 2006

[15] T.P. Fries, "A Fuzzy-Genetic Approach to Network Intrusion Detection," Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation, Atlanta, GA, USA, pp.2141-2146, 2008

[16] T.T.T. Nguyen, G. Armitage, "A survey of techniques for internet traffic classification using machine learning," Communications Surveys & Tutorials, IEEE , vol.10, no.4, pp.56-76, Fourth Quarter 2008

[17] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita. "An effective unsupervised network anomaly detection method," In Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI '12). ACM, pp.533-539, New York, NY, USA, 2012

[18] H.R. Zeidanloo, Bt Manaf, P. Vahdani Amoli, F. Tabatabaei, M. Zamani, "Botnet Detection Based on Traffic Monitoring," International Conference on Networking and Information Technology (ICNIT), vol., no., pp.97 – 101, Manila, Philippines, 2010

[19] H.R. Zeidanloo, M.J.Z. Shooshtari, P. Vahdani Amoli, M. Safari, M. Zamani,"A taxonomy of Botnet detection techniques,"3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), vol.2, no., pp.158 – 162, Chengdu, China , 2010

[20] H.R. Zeidanloo, F. Tabatabaei, P. Vahdani Amoli, A. Tajpour, "All about Malwares (Malicious Codes)," Proceedings of the 2010 International Conference on Security Management, SAM 2010, pp.342-348, Las Vegas Nevada, USA, 2010

[21] F.F. Etemad, P.Vahdani Amoli, "Real-Time Botnet Command and Control Characterization at the Host Level," 6th International Symposium on Telecommunication with emphasis on Information and Communication Technology (IST'2012), Tehran, Iran, 2012

[22] A. Karasaridis, B. Rexroad, D. Hoeflin, "Wide-scale botnet detection and characterization," Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, pp.7-7, Cambridge, MA, USA, 2007

[23] H.C. Lin, C.M. Chen, J.Y. Tzeng, "Flow Based Botnet Detection," Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on , vol., no., pp.1538-1541, 2009

[24] W. Hong, G. Zhenghu, G. Qing, Wang Baosheng, "Detection Network Anomalies Based on Packet and Flow Analysis," Seventh International Conference on Networking, 2008. ICN 2008., vol., no., pp.497-502, 2008

[25] Y. Waizumi, H. Tsunoda, M. Tsuji, Y. Nemoto, "A Multi-Stage Network Anomaly Detection Method for Improving Efficiency and Accuracy," Journal of Information Security, vol.3 no. 1, pp.18-24, 2012

[26] M. Amini, R. Jalili, H.R. Shahriari, "RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks," Computers and Security, Elsevier Inc, vol.25, Issue 6, pp.459-468, 2006

[27] P. Casas, J. Mazel, P. Owezarski, "Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge," Computer Communications, vol.35, Issue 7, pp.772-783, 2012

[28] G. Cormode, S. Muthukrishnan, "What's new: finding significant differences in network data streams," IEEE/ACM Transactions on Networking (TON), vol.13, Issue 6, pp.1219-1232, 2005

[29] Cisco.com, "Cisco IOS NetFlow Configuration Guide, Release 12.4," http://www.cisco.com, Sep. 2012

[30] M. Ester, H.P. Kriegel, J. Sander, X. Xu , "A density-based algorithm for discovering clusters in large spatial databases with noise," Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), AAAI Press. pp.226–23, 1996

[31] P.C. Mahalanobis, "On the generalised distance in statistics," Proceedings of the National Institute of Sciences of India 2 (1) : pp.49–55, 1936

[32] M. Tavallaee, E. Bagheri, Lu Wei, A.A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," Computational Intelligence for Security and Defense Applications, CISDA 2009. IEEE Symposium on , vol., no., pp.1,6, 8-10 July 2009

[33] A. Shiravi, H. Shiravi, M. Tavallaee, A. A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Computers & Security, vol.31, Issue 3, May 2012, pp.357-374, ISSN 0167-4048, 2012

PIII

DISTRIBUTED AGENT BASED MODEL FOR INTRUSION DETECTION SYSTEM BASED ON ARTIFICIAL IMMUNE SYSTEM

by

Farhoud Hosseinpour, Sureswaran Ramadass, Andrew Meulenberg,
Payam Vahdani Amoli and Zahra Moghaddasi 2013

International Journal of Digital Content Technology and
its Applications (JDCTA), Vol. 7, No. 9, pp. 206-214

PIV

ARTIFICIAL IMMUNE SYSTEM BASED INTRUSION DETECTION: INNATE IMMUNITY USING AN UNSUPERVISED LEARNING APPROACH

by

Farhoud Hosseinpour, Payam Vahdani Amoli,
Fahimeh Farahnakian, Juha Plosila and Timo Hämäläinen 2014

International Journal of Digital Content Technology and
its Applications (JDCTA), Vol. 8, No. 5, pp. 1-12

PV

UNSUPERVISED NETWORK INTRUSION DETECTION SYSTEMS FOR ZERO-DAY FAST-SPREADING ATTACKS AND BOTNETS

by

Payam Vahdani Amoli, Timo Hämäläinen, Gil David,
Mikhail Zolotukhin and Mahsa Mirzamohammad (Accepted Nov/2015)

International Journal of Digital Content Technology and its Applications
(JDCTA)