INTRUSION DETECTION: A GAME THEORETIC APPROACH

MONA MEHRANDISH

A THESIS
IN
THE DEPARTMENT
OF
COMPUTER SCIENCE AND SOFTWARE ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE
CONCORDIA UNIVERSITY
MONTRÉAL, QUÉBEC, CANADA

JUNE 2006

Your file Votre référence
ISBN: 978-0-494-20779-6
Our file Notre référence
ISBN: 978-0-494-20779-6

Abstract

Intrusion Detection: A Game Theoretic Approach

Mona Mehrandish

In this thesis, we consider the problems of detecting intrusions initiated by cooperative malicious nodes and multiple malicious packets initiated by a smart intruder. Detection is accomplished by sampling a subset of the transmitted packets over selected network links or router interfaces. Given a total sampling budget, our framework aims at developing a network packet sampling strategy to effectively reduce the success chances of an intruder. We consider two different scenarios: 1) A well informed intruder divides her attack over multiple packets in order to increase her chances of successfully intruding a target domain. 2) Different cooperating intruders distribute the attack among themselves each sending the attack fragments to the target node. Each of the packets containing a fragment of the attack is transmitted through a different path using multi-path routing, where each path is selected with a different probability. To the best of our knowledge, there has not been any work done for the case where the attack is split over multiple packets or distributed over cooperative intruders using game theory. We formulate the game theoretic problem, and develop optimal sampling schemes.

Acknowledgments

I am deeply indebted to my advisors, Dr. Chadi Assi and Dr. Mourad Debbabi, for their constant support. Without their help, this work would not be possible. I would also like to thank the members of my defense committee: J. William Atwood, Peter Grogono and Amr youssef. Their advice and patience is appreciated. A special thank also goes to Hadi Otrok, who helped me a lot both as a colleague and a friend. Without his help, this work could have taken many more years. I would like to thank all my friends at CIISE for the inspiration and support. I would also like to thank Jacques Labelle, who helped me in reading and editing the whole thesis. I want to thank all my friends, for all the nights out and emotional support. I would like to thank (in alphabetical order) Abdelghani, Ahmad, Ali, Amir, Andre, Elaheh, Hassan, Ivetta, Lamia, Luay, Mahya, Maryam, Mazdak, Mehdi, Payam, Payam, Rebab, Sarah, Siamak, Siavash, Shahab, Sirin, and Yosr. I also want to thank Aryan for the food support.

Lastly, I would like to thank my family for their support. I am greatly indebted to my mother for staying with me in the cold winter to help me out.

I dedicate this thesis to Nahid, Hamid and Maral.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Systems and networks are subject to continuous electronic attacks. Frequent attempts to violate information security requirements for data protection are increasing everyday. Therefore, some tools have been developed as solutions to this problem such as vulnerability-assessment tools and intrusion detection systems. Both of these tools allow organizations to protect themselves from losses related to network security problems. Vulnerability-assessment tools check systems and networks for system problems and configuration errors that represent security vulnerabilities. Intrusion detection systems (IDS) collect information from a variety of points within computer systems and networks (depending on the IDS information source type) and analyze this information for signs of security violations [10].

In a nutshell, intrusion detection systems do exactly as the name suggests: they detect possible intrusions. In other word, intrusion detection system tools gather information, depending on the information source (i.e., network or hosts), analyze the information to detect computer attacks and/or computer misuse, and then alert the proper individuals upon detection. An IDS installed on a network functions the same as a burglar alarm system installed in a house. Through various methods, both detect the existence of an intruder/attacker/burgler, and they both subsequently issue some kind of warning or alert [25].

Even though IDSs may be used in conjunction with firewalls, they should not be considered as the same thing. Firewalls aim at regulating and controlling the flow of information into and out of network acting as prevention tools. They protect a network and attempt to prevent intrusions, while IDS tools detect whether if the network is under attack. On the other hand, IDS tools form an integral part of a thorough and complete security system,

when used with security policy, vulnerability assessments, data encryption, user authentication, access control, and firewalls. Using the previous example, firewalls can be thought of as a fence or a door positioned in front of a house and IDSs can be thought as a security camera inside the house [25].

There are three fundamental security functions that intrusion detection systems serve: monitoring, detecting, and responding to unauthorized activities by insider or outsider intruders. Intrusion detection systems use rules to characterize certain events that, if detected (or not detected) will issue an alert. More specifically, if a particular event (or non existence of the particular event) is considered to violate the system security, an alert will be issued if that event is detected (or not detected). The intrusion response [11] varies for different kinds of IDSs. Most intrusion detection systems have the capability of sending out alerts, so that the network administrator receives a notification of a possible security violation in the form of a page, email, or SNMP trap. Some other intrusion detection systems not only recognize a particular incident and issue an appropriate alert, but also respond automatically to the event. The intrusion response might be logging off a user, disabling a user account, or launching of scripts [25].

Unlike what is believed by most of people, the vast majority of the security incidents that occur on a network is launched by an insider attacker. The insider attackers can be authorized users who are disgruntled employees. The rest of the attacks come from the outside, in the form of denial of service attacks or attempts to penetrate a network infrastructure. The IDSs are the only proactive means of detecting and responding to threats that can be initiated by both insider and outsider intruders [25].

The February 2000 denial of service attacks [50] against Amazon.com, Yahoo, CNN, and E-Bay(amongst others) illustrated the need for effective intrusion detection, especially within on-line retail and e-commerce. Studies [25] show that almost all large corporations and most medium-sized organizations have installed some form of intrusion detection tool. Nevertheless, it is apparent that given the growing frequency of security incidents, any entity that can be accessed through Internet should have some form of IDS running as a line of defense. The motivation for network attacks and intrusions can be financial, political, military, or personal reasons. Reasonably, if one has a network, she is a potential target, and should have some form of IDS installed.

As we mentioned before, intrusion detection is the process of monitoring computers or networks for unauthorized access, activity, or file modification. Depending on what kind of information the IDS monitors the intrusion detection systems can be divided into two basic

categories: host-based and network-based [11]. Each of these types has a distinct approach to monitoring and securing data, and thus each has distinct pros and cons. In particular, the former inspects data held on individual computers that serve as hosts, while the later inspects data transmitted between computers.

Since the intrusion detection system is a critical component of security tools, it is important to make sure it is functioning as much as expected by the organization deploying it. The system administrator needs to be able to trust and act on the information provided by the system. In case of unsound information (false alarm) [34], the system would be more endangered. In other words, the forensic value of information from faulty systems is not only negated, but potentially misleading.

Knowing that how risky the failure of an ID component can be, it is rational to consider the ID systems as targets for the attackers. A well informed intruder might likely attack the IDS first, immobilizing it or feeding false information to it and then attack the actual system. For example, the intruder might distract security personnel from the actual attack in progress, or frame someone else for the attack.

In order for a software component to defend against the attack, it must consider the specific means by which it can be attacked, both in the design and implementation phase. Unluckily, very little information is accessible to IDS designers to document the traps. Besides, the dynamic characteristic of the attacks makes it impossible for the IDS designer to plan the IDS considering all the possible attacks. However, most of the IDSs have the feature that new attacks can be added to their databases as new attacks are discovered. Furthermore, the majority of commercially available IDSs are not open source (they have proprietary, secret designs), making the independent third-party security analysis of such systems a difficult task.

ID systems that consider incoming packets independently of each other, and thus retain no state information across packets, can easily be evaded [36]. In case of stateless IDSs, an intruder can avoid detection by spreading elements of the attack over multiple packets (for example, through IP fragmentation or TCP segmentation). Even stateful IDSs can sometimes be evaded by exploiting slight differences in the way the packet is handled by the IDS on one hand and the targeted system at the other. Therefore, it is crucial for the system administrators to deploy a stateful IDS for networks that need more security (e.g., e-banking facilities).

1.2 Objective of The Thesis

Noting that how essential the stateful ID systems are, we take into consideration two scenarios where stateful detection is needed. In the first scenario, a well informed intruder divides her attack over multiple packets in order to increase her chances of successfully intruding a target domain. We use game theory as a tool to analyze the possibility of an intrusion going undetected. More specifically, we build a game theoretic framework to model network intrusions through multiple packets. Detection is accomplished by sampling a portion of the packets transiting through selected network links (or router interfaces). Given a total sampling budget, our work in this thesis then aims at developing a network packet sampling strategy to effectively reduce the success chances of an intruder. As mentioned before, the intruder launches the attack over multiple packets. Each of the packets containing a fragment of the attack is transmitted through a different path using multi-path routing, where each path is selected with a different probability [42].

The second scenario considers a distributed cooperative attack. Here, a group of cooperating intruders distribute the attack over multiple packets. Then, each intruder sends a fragment of the attack to the target node. Again, a stateful IDS is needed to detect the attack. The attack is successful if a certain number of the malicious fragments reach the target node without being detected. We formulate this problem using a game theoretic framework and then present sampling strategies for the IDS in order to maximize the probability of detection. The sampling budget constraint holds for this case as well. To the best of our knowledge, there has not been any work done for any of these scenarios.

The rest of this thesis is organized as follows. In Chapter 2, we present a survey of intrusion detection systems. We introduce different taxonomies for IDSs, and we illustrate them in more detail. Chapter 3 gives the background about game theory. It presents a taxonomy of games and explains the games in each category with examples. Furthermore, it discusses some famous examples in game theory. Chapter 4 brings in some studies including game theory and intrusion detection and carries our contributions, which are followed by the concluding remarks of Chapter 5.

Chapter 2

Intrusion Detection

2.1 Introduction

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network [12]. An intrusion detection system (IDS) inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. There are several ways to classify an IDS, e.g., misuse detection vs. anomaly detection [11]. In misuse detection, the IDS analyzes the information it gathers and compares it to large databases of attack signatures. Fundamentally, the IDS looks for a particular attack that has already been documented. Like a virus detection system, the functionality of misuse detection software is dependent on the database of attack signatures that it uses to compare packets against. On the other hand, in anomaly detection, the system administrator defines the normal state of the network's traffic load, breakdown, protocol, and typical packet size. The anomaly detector monitors network segments to compare their state to the normal behavior and look for anomalies. As mentioned in the previous chapter, another way of classifying the IDSs is network-based vs. host-based systems [11]: in a network-based system, or NIDS, the individual packets flowing through a network are analyzed. The NIDS can detect malicious packets that are designed to be overlooked by a firewall's basic filtering rules. In a host-based system, as one can guess from the name, the IDS examines the activity on each individual computer or host. Furthermore, ID systems also can be classified as passive system vs. reactive system [11]. In a passive system, the IDS

detects a potential security violation, logs the information and reports an alert while in a reactive system, the IDS response is not only raising an alert but attempting to stop the suspicious activity by logging off a user or by reprogramming the firewall to block network traffic from the suspected malicious source.

Though commonly mistaken for each other, an IDS differs from a firewall in that a firewall prevents the intrusions by filtering the traffic between inside and outside of the network, while an IDS looks out for intrusions in order to stop them from happening. A firewall doesn't consider the probability of existence of an insider intruder and it is not capable of detecting an attack that is launched by an insider intruder. An IDS, on the other hand, evaluates a suspected intrusion once it has taken place and signals an alarm. Furthermore, an IDS also watches for attacks that originate from within a system. An IDS should be considered as a tool to be used in conjunction with the other standard security products such as anti-viruses and firewalls in order to increase the system's security.

Intrusion detection is the art and science of sensing when a system or network is being used improperly or without authorization. An intrusion-detection system monitors system and network resources and activities. Then, it analyzes the information gathered from these sources, in order to detect whether an intrusion is in progress or not. When an intrusion is sensed, the IDS notifies the authorities by raising an alarm.

2.1.1 Motivation

Considering the potential security threats of intrusions in contemporary businesse computer networks, the creation and design of intrusion detection systems are exemplified in a simple manner. The following lists various threats that would promote intrusion detection systems for businesses [24].

1. **Loss of Financial Assets:** Financial transactions are processed every day across the globe. This increasing trend opens financial institutions to subversion from within or from across the computer networks. The risk is not limited solely to financial institutions since online banking has rendered the customer's financial information liable to be intercepted via the Internet.

2. **Loss of Intellectual Property:** In the competitive market, an important threat is crucial to be considered: intellectual property threat. The concept of intellectual property protects the business's ownership of ideas and products. Theft of intellectual property can severely hinder any local or global company as the competitive edge is

taken away. It is easy to understand that any protection against such threats is vital to a company.

3. **Loss of Computing Resources:** The company's inability to access and process crucial information can cripple its financial standings. Missed business transactions, disrupted e-mail system and lost customer confidence in the online service are perfect examples on how a loss of computing resources affects the company.

4. **Loss of Privacy:** The protection of sensitive information concerns both citizens and companies. Thefts of personal information can lead to invasion of privacy for the citizens and lawsuits for the company's lack of protection of personal information. The sources of these types of attack are wrongfully presumed to come from outside of the company when in fact, employees who have direct access to the computer systems are more likely the culprit of these attacks. This misconception has left companies focusing on firewalls and web servers instead of preventing malicious employees or corporate espionage. The followings are the scenarios that lead to security problems [24].

   (a) **Misplaced Trust:** The authenticity of information and processes of a computer system are usually taken for granted by the user as trust is put into the computer system's intent. Accuracy, confidentiality and correctness of information is a typical misplaced trust when viewing and accessing a web page. Other types of misplaced trust can be illustrated when you enter a password to gain access to a service or system, you trust that is has not been stored to be used at a later time and when receiving e-mails, you trust the origins of the sender in the sender heading.

   (b) **Malicious Code:** Businesses experiences great losses due to the effect of computer viruses on business productivity. The danger experienced by computer viruses is not due from actual viral damage but by the containment of the virus.

   (c) **Strong Security With a Weak Link:** Security for a computer network can be as simple and complex as desired but complexity is futile if a single flaw leaves an open space for intrusions.

   (d) **Exploitation of Critical Infrastructure Elements:** Computer networks consist of elements of varied importance. A critical service or hardware that acts as a nexus to other elements is an optimal target to cause severe damage to the entire network.

(e) **Mis-configured Software and Hardware:** Configurations of simple and complex software or hardware contain potential network security threats. If not properly configured, the software or hardware would enable an intrusion or an attack.

(f) **Excessive Privilege for Simple Tasks:** An improperly designed code that is executed with high privileges can cause serious security issues. A seemingly inoffensive bug in a program could cause major repercussions in these situations. This is common as it is frequent to see code being executed with unnecessary privileges.

(g) **Being Used as a Springboard to Attack the Next Victim:** A computer system can be used to relay an attack to a third party.

Existence of these threats leads to the development of new security techniques and methods. Three indispensable tools are discussed: firewalls, encryption and security auditing tools.

1. **Firewalls:** A firewall consists of a system that acts as a traffic controller between two networks. Its common utilization is to control traffic between the Internet and an intranet. Using policy enforcement, permissions established by the firewall prevent outside attacks on the computer systems but do not prevent insider attacks or stop systems from becoming a springboard as discussed earlier. Obviously this is mainly useful when the limits of the Internet and the intranet are well defined. New concepts such as extranet (multi-partner intranets) have increased the ambiguity of the intranet limits and therefore the implementation of a firewall has increased in difficulty.

2. **Encryption:** Encryption uses mathematical algorithms to render data unreadable and unmodifiable unless specifically authorized by the algorithm. It also authenticates the identity of the sender of the data enabling confidence in the origin of a message or data.

   The core of any cryptographic algorithm is the encryption key. While the value of the key is known, any encrypted messages of this key can be decrypted, altered and retransmitted. A great advantage of this system is that even if the algorithm is known, without the key, the decryption cannot be done. This method is not infallible since it is still prone to insider attack such as a third party copying the key.

Data has to go through an encryption process to be considered secured, leaving any unencrypted data unsecured but with this in mind, encryption is uniquely a valuable part for a security system.

3. **Security Auditing Tools:** Computer systems and networks can contain a series of vulnerabilities that can be used for an intrusion or an attack. Security auditing tools scan for any vulnerabilities and generates a report including recommendations to remedy the situation. This concept, if done frequently, can be a vital tool against intrusions or attacks. The frequency aspect of the tool greatly reduces the potential dangers so if run in a continuous fashion, this leads us to the concept of the Intrusion Detection System.

An Intrusion Detection System tries to alleviate security threats to computer systems and computer networks. A large number of threats occurs for several reasons:

1. Valuable information is found on computer networks that can interest people who have rightful claim to the information or malicious people who profit from this information.

2. Evolving technology entails evolving threats and it becomes virtually impossible to keep track of all the existing and possible threats. The rate of evolution overwhelms the measures to detect and fix all possible threats.

3. The Internet is a great medium to deliver information to an individual or to groups of individuals. With this attribute, the Internet has been filled with hackers trying to demonstrate their abilities by creating numerous threats to computer networks across the Internet.

4. Since a large number of threats come from hackers, the method of intrusion has been simplified by creating tools. These tools are often made freely available and thus even novice hackers gain the ability to breach a security system. This means that one with no security specialty to write hacks and code tools can launch an attack. Consequently, the number of attacks has increased greatly over the last few years. Figure 2.1 shows the number of incidents reported to CERT (Computer Emergency Response Team)[1] since 1989.

[1] http://www.cert.org/

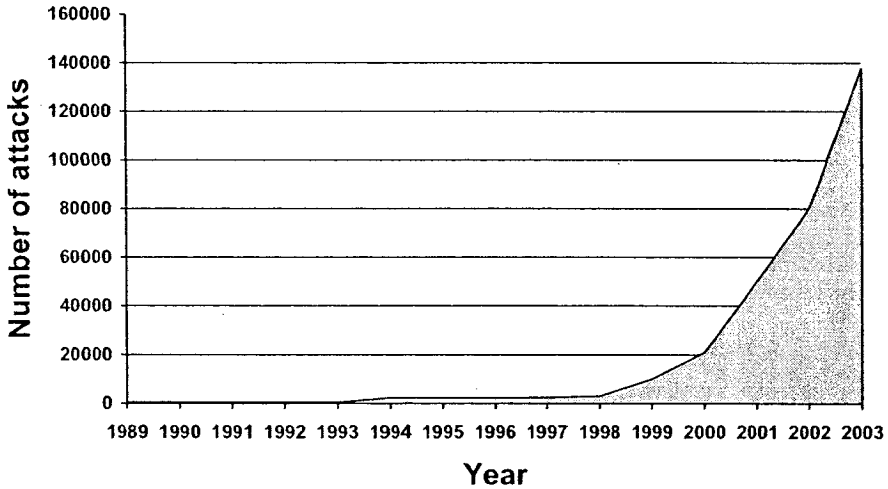

Figure 2.1: Number of incidents reported to CERT (Computer Emergency Response Team) [1]

5. The low risk involving Internet Based attacks encourages hackers to practice without fear of repercussions of being identified.

6. The accessibility of the Internet allows a high range for the attacks as networks are often open to the Internet through some services. Even though a network can be reached from the outside, insider attacks still constitute the highest proportion of attacks.

7. Security systems are not infallible therefore they allow threats to occur.

8. Threats pass undetected due to the level of traffic through computer networks of these days. Past techniques such as log monitoring are inadequate for the traffic level that occurs presently.

2.2 The Intrusion Detection Problem

Significantly, it is impossible for any one person to continually monitor the networks manually, due to the enormous amount of information flowing in the network and the level of activity on most corporate servers. Traditional network management and system monitoring tools do not address the issue of helping to ensure that systems are not misused and

abused [24]. In case of theft of a company's critical data, the traditional systems are incapable of detecting the threat. Mostly corporation's intellectual property resides on server machines. Therefore, theft of information is can be really expensive for them. This makes a tool that could detect security-related threats and attacks as they occur, a necessity for them.

To make it clearer, we can think of an intrusion detection system as a security camera inside the house, and the fence and front door as firewalls and other security tools [25]. Once an intruder bypasses all the defense lines, the IDS is like the closed circuit TV cameras that security guards monitor in order to obstruct the attack.

Intrusion Detection is the art of detecting inappropriate, incorrect, or anomalous activity [2]. An intrusion detection system monitors a server machine, a whole network, or even an application (such as a database or web server) in order to detect patterns of misuse or anomalies that may correspond to security breaches. The monitoring is automatic and even on all the systems on which the IDS is deployed. Consequently the IDS imposes a low overhead on the systems and network. This overhead should be low enough so that it does not disrupt the system normal activities.

In order to attack a system, an attacker first looks for any security holes that make the system vulnerable to subversion. After identifying a vulnerability to exploit, the attacker will then generate an attack script and send it to the victim. The attack scripts are frequently just shell scripts or simple programs that perform a series of fixed steps to exploit the vulnerability. In case of normal intruders, the script has already been written and is available on a web page in which case the intruder just downloads it [24].

Most of the attacks are simply variations of each other. As soon as an attacker identifies a weak spot and releases an attack script for it, many others who are inspired by her work find similar weaknesses in other pieces of software. Therefore, attacks can be classified into groups with common patterns in each group. Thus, after codifying an attack, it is useful to introduce *detection templates*, such that any pattern that is analogous to this template can be marked as an intrusion. This helps with detecting an unknown intrusion, for which we have a variation of it in the database of attacks.

2.3 Evaluating Intrusion Detection Systems

Before classifying the intrusion detection systems, we explain some concepts used as standard measures for evaluating IDSs. These measures are shown in Figure 2.2.

Standard metrics		Predicted connection label	
		Normal	Intrusions (Attacks)
Actual connection label	Normal	True Negative (TN)	False Alarm (FP)
	Intrusions (Attacks)	False Negative (FN)	Correctly detected intrusions - Detection rate (TP)

Figure 2.2: Measures for evaluating IDSs

1. The detection rate is the ratio between the number of correctly detected attacks and the total number of attacks.

2. The false alarm (false positive) rate is the ratio between the number of normal connections that are incorrectly misclassified as attacks (False Alarms in Table) and the total number of normal connections. In other words, the false positive rate is the frequency with which the IDS reports malicious activities while the activities are normal. A high false positive rate can be extremely dangerous in a way that it may either cause administrators to block normal activities which leads to denial of service or either cause administrators to ignore the system's output when legitimate alerts are raised. In general, increasing the sensitivity of an intrusion-detection system results in a higher false positive rate and consequently decreasing it lowers the false positive rate.

3. The False negative rate is the rate with which the IDS cannot detect an intrusion. In other words, it is the ratio of the malicious activities for which the IDS fails to raise an alert to total number of malicious activities. The main concern of an IDS is to detect any malicious activity. Therefore, it is clear why these are the most dangerous types of errors, as they represent undetected attacks on a system. Additionally, as one might expect, false negative rates change in an inverse proportion to false positive rates.

4. The Crossover Error Rate (CER) [33] is a point at which the system is tuned so that both kinds of false responses occur with the same frequency. Furthermore, CER provides us with a metric to be able to compare different IDSs with each other. Since the sensitivity of systems has an influence on the false positive/negative rates, it is important to have some common measure that may be applied across the board. The

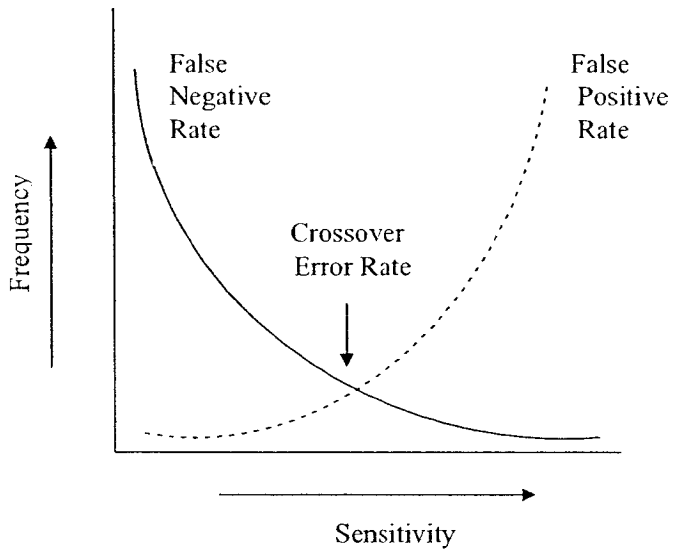

Figure 2.3: Crossover Error Rate (CER)

CER for a system is determined by regulating the system's sensitivity until the false positive rate and the false negative rate are equal, as shown in the Figure 2.3.

Selecting an IDS depends on the main concerns of the administrators. If one is interested in achieving a balance between false positives and false negatives, she may simply select the system with the lowest CER. On the other hand, if detecting every single attack is of the utmost priority, you may still wish to select the system with the lowest false negative rate noting that this selection may increase the administrative overhead associated with false positive reports.

2.4 Classification of Intrusion Detection Systems

Intrusion detection can be classified according to any of the following options [37]:

1. Information source: Host-based ID, network-based ID, wireless-network ID, application logs, or sensor alerts.

2. Time aspects in analysis: Real-time analysis vs. off-line analysis.

3. Architecture: Single centralized vs. distributed & heterogeneous.

4. Continuity: Continuous analysis vs. periodic analysis.

5. Analysis strategy: Anomaly detection vs. misuse detection.

2.4.1 Information source

The source of the information might be the information obtained from a single host (e.g. system log data, system calls data), monitored traffic in the network to which the hosts are connected, traffic between mobile nodes, database logs, web logs, alarms generated by other IDSs and etc.

Generally, the IDSs are divided into two major groups depending on the source of information. The set of IDS tools that use information derived from a single host (system) are called host based IDS (HIDS), and those IDSs that use information gained from a network are called network based IDS, i.e., NIDS [11].

The following systems [28] can be distinguished as HIDS:

1. Systems that monitor incoming connection attempts. These systems examine host-based incoming and outgoing network connections. Furthermore, they look for illegitimate connection attempts to TCP or UDP ports and can also detect incoming *portscans*.

2. Systems that examine network traffic and look for packets that attempt to access the host. Furthermore, they protect the host by intercepting suspicious packets and looking for abnormal payloads.

3. Systems that examine the network layer of their protected host and monitor login activity onto the host. They monitor log-in and log-out attempts, in order to detect any unusual activity including an event occurring at unexpected times or in particular network locations. As an example, we can mention detecting multiple failed login attempts.

4. Systems that monitor actions of a super-user, e.g., root, who has the highest privileges. These systems inspect any unusual activity related to access control such as increased super-user activity.

5. Systems that monitor file system integrity. Tools that are capable of checking the integrity allow the IDS to detect any changes to the files that are critical for the

operating system. Therefore, any illegal changes in the system would be logged and would raise an alert to notify the system administrator.

6. Systems that monitor the Kernel. These systems examine the status of critical operating system files and streams. Furthermore, they are capable of blocking a part of the actions undertaken by the super-user.

As one may say from the name, a host based intrusion detection system inhabits a particular computer (host) and provide protection for a specific computer system. In addition to monitoring the host for misused patterns or anomalies, they might respond to an intrusion as well.

On the other hand, the network-based type of IDS (NIDS) reassembles and analyzes network packets that reach the network interface card for signs of intrusions. Unlike the host based model, they do not only deal with packets going to a specific host. Having an NIDS deployed in a network segment, all the machines in that segment can benefit from the NIDS. Network-based IDS can also be installed on active network elements such as routers [28].

Some of the classical attacks, e.g., DOS [50], need statistical data on the network load in order to be detected. Therefore, a certain group of NID systems can be introduced to monitor the network traffic and collect statistical data. These types of intrusion detection systems do not analyze the captured packets but only focus on creating network statistics.

There is also another class of IDSs called Network Node IDS (NNIDS), which is basically a mixture of HIDS and NIDS. An NNIDS has its agents deployed on every host within the network being protected and a single agent usually processes the network traffic directed to the host it runs on [28]. The main reason to bring in such hybrid IDS was packet encryption where only the source and destination could see decrypted network traffic which made it impossible for the other nodes to analyze the traffic.

2.4.2 Time aspects in analysis

Intrusion detection systems are categorized to real-time and off-line in terms of analysis time aspects [37]. In the former the intrusion detection system analyzes the data while the sessions are in progress (e.g. network sessions for network intrusion detection, login sessions for host based intrusion detection). It raises an alarm immediately when the attack is detected. On the other hand, the IDS analyzes the data when the information about the sessions are already collected in case of an off-line intrusion detection system. Afterwards, it

analysis the data. Off-line analysis is very useful for understanding the attacker's behavior.

2.4.3 Architecture

The IDS can operate either as a stand-alone centralized application or an integrated application that creates a distributed system [28]. In the former, data analysis is performed in a fixed number of locations, independent of how many hosts are being monitored while the latter has a particular architecture with autonomous agents that are sometimes able to move over the network. There, data analysis is performed in a number of locations proportional to the number of hosts that are being monitored [22]. Furthermore, a distributed architecture is necessary for detection of distributed/coordinated attacks targeted at multiple networks/machines.

2.4.4 Continuity

The IDS may monitor the data continuously or in periodic intervals. In continuous monitoring, the IDS performs a continuous, real-time analysis by acquiring information about the events instantly after they occur [37]. Continuous monitoring can be expensive due to transferring the audit data and processing them in real time. In Periodic Analysis, the IDS periodically takes a snapshot of the environment, i.e., monitored system, and then analyzes the data snapshot. Afterward, it looks for any sign of misuse patterns or anomalies. Obviously, due to high cost of continuous monitoring, periodic analysis is widely used by system administrators, but not satisfactory to ensure high security, since the security exposure between two consecutive runs is sufficient for an intruder with enough knowledge about the system.

2.4.5 Analysis Strategy

Intrusion detection systems usually use signature based approaches, anomaly based approaches or a hybrid of the two [12]. In a signature based approach, we have a database of attack signatures and intrusion patterns. The IDS would examine the packets and compare them against this database to find any misuse attack. On the other hand, we have some patterns or rules of normal activities in an anomaly based approach. In this case, the IDS examines the traffic. Any behavior, i.e., monitored traffic, deviating from normal pattern would be labeled as an anomaly, which alerts the system administrator. Figure 2.4 shows

the classification of analysis strategies each with an example of an existing IDS [9]. We will discuss them later in detail as intrusion detection techniques.

2.5 Intrusion Detection Approaches

2.5.1 Data Mining and Machine Learning Approaches

In the data mining approach, we are able to detect new types of attacks. Here, anomalies are detected using predefined rules. In order to be able to update the system with the appropriate rules, the system supervisor should know the behavior pattern for a certain anomaly to make the system adaptive. The system administrator should design several rule sets for various attack patterns. Here, the rule generation methodology is done using data mining techniques. An association rule (item set) [23] is defined with the following generic form: $X \rightarrow Y, c, s$ where X and Y are the item sets for the rule and $X \cap Y = \emptyset$ is the relation between them. $s = support(X \cup Y)$ where s is the support value for the rule and $c = \frac{support(X \cup Y)}{support(X)}$ is the confidence for the rule. The system keeps these rules for a period of time and uses them as the pattern for the event and behavior model for the users. As an example [38], an association rule for the shell command history file (which is a stream of commands and their arguments) of a user is: $trn \rightarrow rec.humor$, 0.3, 0.1, which indicates that 30 % of the time when a user invokes *trn*, he or she is reading the news in rec.humor, and reading this newsgroup accounts for 10% activities recorded in her command history file.

There is another rule called frequent episode rule [46]: $X, Y \rightarrow Z, c, s, window$ where X and Y are the item sets for the rule and $X \cap Y = \emptyset$ is the relation between them. $s = support(X \cup Y \cup Z)$ where s is the support value for the rule and $c = \frac{support(X \cup Y \cup Z)}{support(X \cup Y)}$ is the confidence for the rule and *window* is the sampling window interval. Applying proper subintervals, the system will reduce the length of the user records. At the same time, the system will keep the historical records for the activities in its database (data reduction). Using the user records, the system will generate a rule set for the activities within the network. At this stage, the system can notice the irregularities and identify them (if they are known).

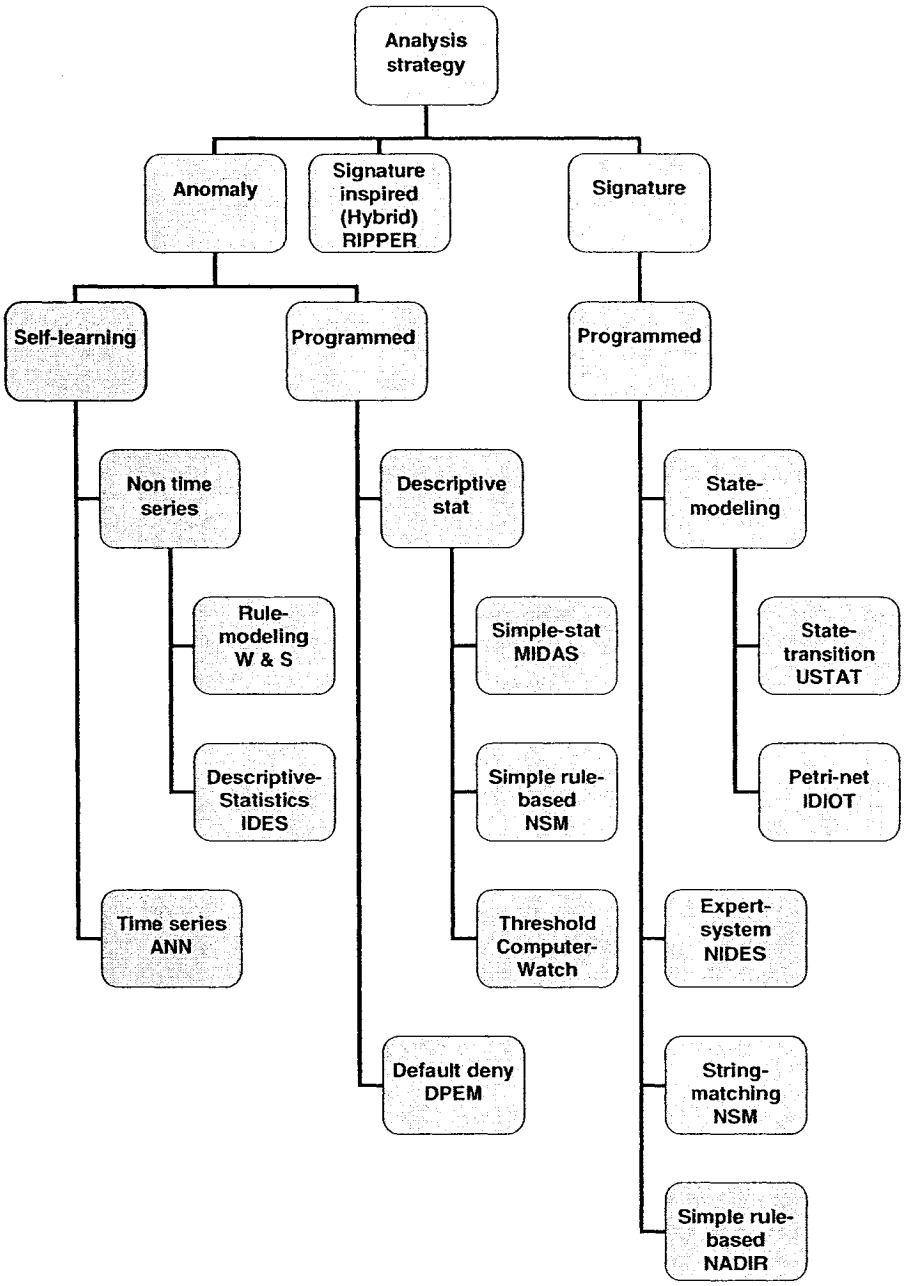

Figure 2.4: Taxonomy of analysis strategies with examples [9]

2.5.2 Artificial Intelligence Approaches

With many applications of artificial intelligence in intrusion detection, researchers have proposed several approaches in this regard. Data mining using the association rule is also one of the approaches which we already discussed in the previous subsection. Other AI approaches propose application of the fuzzy logic concept into the intrusion detection problem area. In [20], the authors propose a method of combining multiple decision trees based on fuzzy logic [27], especially the fuzzy integral. In order to improve detection performance of intrusion detection system, they divide a great large dataset into several sub-datasets, and mine on sub-datasets separately to construct different sub-decision trees. Then, they detect data by different sub-decision trees, and finally they nonlinearly combine the results from multiple sub-decision trees by fuzzy integral. In another study [53], Hidden Markov Model (HMM) has been deployed to detect intrusions. In that approach a Markov-chain model represents a profile of computer-event transitions, which is generated by historic data. Then, the observed activities of the system are analyzed to infer the probability that the Markov-chain model of the norm profile supports the observed activities. The lower this probability is, the more likely the observed activities are anomalies resulting from cyber-attacks.

Some researchers [35] have tried to use the Bayesian methodology [21] to solve the intrusion detection problem. The main idea behind this approach is the unique feature of the Bayesian methodology. For a given consequence, using probability calculations, the Bayesian methodology can move back in time and find the cause of the events. This feature is suitable for finding the reason for a particular anomaly in the network behavior. Using Bayesian algorithm, system can somehow move back in time and find the cause for the events. Although using the Bayesian methodology for the intrusion detection or intruder behavior prediction can be very appealing, however, there are some issues that one should be concerned about them. Since the accuracy of this method is dependent on certain presumptions, distancing from those presumptions will decrease its accuracy. Usually these presumptions are based on the behavioral model of the target system. Selecting an inaccurate model may lead to an inaccurate detection system. Therefore, selecting an accurate model is the first step towards solving the problem. Unfortunately due to the complexity of the behavioral model within this system finding such a model is a very difficult task.

Furthermore, Artificial Neural Network (ANN) has been proposed by some researchers for intrusion detection purpose. The main goal of using the ANN approach is to provide an unsupervised classification method to overcome the curse of dimensionality for a large

number of input features. Normally in an Intrusion detection system, the system is complex and input features are numerous. Therefore, clustering the events can be a very time consuming task. Using the Principal Component Analysis (PCA) [14] or Singular Value Decomposition (SVD) [47] methods can be an alternative solution. However, if not used properly both of these methods can become computationally expensive algorithms. At the same time, reducing the number of features will lead to a less accurate model and consequently it will reduce the detection accuracy. In the computer networks intrusion detection problem area, the size of the feature space is obviously very large. Once the dimensions of the feature space are multiplied by the number of samples in the feature space, the result will surely present a very large number. This is why some researchers either select a small sampling time window or reduce the dimensionality of the feature space. Since the processing time is an important factor in the timely detection of the intrusion, the efficiency of the deployed algorithms is very important. Time constraint may sometimes force us to have the less important features pruned (dimensionality reduction). However, the pruning approach is not always possible.

2.5.3 Embedded Programming and Intrusion Detection

In this approach [41], the goal is to preprocess the network information using a preprocessor hardware (front-end processor). Here, some parts of the processing are performed prior to the IDS. This preprocess will significantly reduce the processing load on the IDS and consequently the main CPU. For example, programming the Network Interface Card (NIC), can have many properties including lower computational traffic and higher performance for the main processor. Since the NIC is performing the major part of the processing while the main processor only monitors the NIC operation, implementing this approach will make it easier to detect a variety of attacks such as Denial of Service (DoS) attack.

2.5.4 Agent Based Intrusion Detection

Another approach is the distributed or the agent based computing [22]. In this approach not only the workload will be divided between the individual processors, but also the IDS will be able to obtain an overall knowledge of the network working conditions. Having an overall view of the network will help the IDS to detect the intrusion more accurately and at the same time it can respond to the threats more effectively. In this approach, servers can communicate with one another and can alarm each other. In order to respond to an attack,

sometimes it can be sufficient enough to disconnect a subnet. In this type of system in order to contain a threat, the distributed IDS can order severs, routers or network switches to disconnect a host or a subnet. One of the concerns with this type of system is the extra workload that the IDS will enforce on the network infrastructure. The communication between the different hosts and servers in the network can produce a significant traffic in the network. The distributed approach can increase the workload of the network layers within the hosts or servers and consequently it may slow them down. There are two approaches in implementing an agent based technology.

In the first approach, autonomous distributed agents are used to both monitor the system and communicate with other agents in the network. A Multiagent based system will enjoy a better perception of the world surrounding it. In this way, the complex system will be broken down into much simpler systems and will become easier to manage. In the second approach, mobile agents are used to travel through the network and collect information or to perform some tasks.

2.5.5 Software Engineering and Intrusion Detection

As the complexity of the IDS increases, the problem of developing the IDS becomes more and more difficult. A programming language dedicated to developing IDSs can be useful for the developer community. Such a programming language with its special components will improve the programming standard for the IDS code. IDS developers can enjoy the benefits of a new language dedicated to the IDS development. Such a language will improve both the programming speed and the quality of the final code.

We illustrate this by presenting two studies. In [51] the main attention is focused on the software engineering aspect of the IDS. Issues such as object-oriented programming, component reusability and the programming language for the IDS are discussed in this paper. A new framework called State Transition Analysis Technique (STAT) is introduced in this paper. In their implemented framework, the authors propose a type of state machine system called STAT that follows the state transition of the attack patterns. This framework is for developing signature based IDSs. There is a STAT-Response class that holds response modules. These response modules include a library of actions that are associated with the pattern of the attack scenarios. All together, this language will produce an encapsulated object-oriented code with a high reusability in the code. There is an event provider module that will provide the framework with the events occurring on the network.

Another approach in programming languages for the IDS is to provide means to follow

the state change in the system. In this way, the IDS will have the ability to have its behavior altered if necessary. Including this feature in the IDS will make it adaptive and reconfigurable. The possibility to alter the behavior of the IDS will provide us with a dynamically reconfigurable IDS. In [48] a State Machine Language (SML) approach was implemented. It is based on the Extended Finite State Automata (EFSA) to model the correct or expected behavior of the network. Using a well designed program in SML, the state machine will be able to follow up with the events within the network and to produce appropriate outputs. If no irregularities detected, then the anomaly detection part of the process will analyze the outputs and will detect the anomalies.

2.6 Intrusion Detection Techniques

There are several taxonomies for intrusion detection depending on the analysis strategy. Here, we have used the taxonomy proposed by [9].

2.6.1 Signature-based Techniques

Signature based intrusion detection (misuse detection) is one of the frequently used and precise techniques of intrusion detection. In this approach there is a database of signatures that any monitored activity is matched upon it. The signature can have many forms such as a special string in an attack code, e.g., *su* in buffer overflow attacks, a threshold on some metrics, e.g., number of unsuccessful attempts to log on a system, or etc. Signature based techniques are very effective once the attack is known, but they can not detect the attack in case of a new attack. In order to keep a signature-based ID system effective, security specialists study the attacks and once a new attack is observed, they analyze and codify it and finally add it to the signature database. The IDS should be updated regularly to recognize the new attack patterns and to respond to them. The main shortcoming of this approach is the fact that it can not detect novel attacks. Once the attack pattern is slightly altered, this approach will not detect the altered versions of the old attacks.

All signature-based techniques are programmed, i.e., the system is programmed with an explicit decision rule. The detection rule is uncomplicated in the sense that it includes a straightforward coding of what can be anticipated to be labeled as an intrusion and what can not be. In a signature based IDS, the patterns that uniquely lead to an intrusion should be clearly stated.

The programmed signature-based techniques can be categorized as the following [9]:

1. State-modeling: In state modeling there is a set of states that should be transited in order for an intrusion to be accomplished. These states should be transited in order of time, i.e., they are time series. State modeling can be classified into two groups: state transition and Petri net [13]. In the former all the states have to be traversed in order for the intrusion to fulfill while in the latter, they are a set of paths that can be traversed so that the intrusion occurs.

2. Expert-system: In an expert system, signatures are a set of rules. The IDS checks the state of the system by applying the security rules looking for any sign of misuse behavior. Forward-chaining, production-based tools are mostly utilized in order to make the system capable of managing new information about attacks. Therefore, expert systems are often flexible, but this often comes at a cost of execution speed when compared with simpler methods.

3. String matching: This if often a simple method which looks for a substring in a monitored packet. The substring search is usually case sensitive. This method is easy to apply since they are many efficient algorithms for string processing. This approach works having the assumption that in order for an attack to fulfill its goal, it should contain a special instruction (string) in the body of the attack code.

4. Simple rule-based: Like expert system, in these system signatures are a set of rules. Here, the rules are much less complicated that the expert system method and as a result it is less expensive in terms of execution speed.

2.6.2 Anomaly-based Techniques

In anomaly based intrusion detection, the system looks for what is called abnormal behavior. Unlike signature based approach, there is no database of attack signatures. Instead, the attack is detected once the network behaves out of its regular way. Obviously, normal behavior varies for different kind of networks. Therefore, in this approach we use a training process where the network behavior is observed for a certain amount of time and logged. Analyzing the network behavior and removing any malicious behavior from this logged information, the normal behavior can be defined. This behavior is different for different systems depending on various factors such as times of the day, the date, different working conditions, etc., which should be considered during creating the normal behavior baseline. Any monitored behavior that deviates significantly from this baseline, would be labeled as

an anomaly. The disadvantage of this approach is that not every anomaly indicates an intrusion. This method will lead to a high rate of false positives, especially when the system is dynamic. On the other hand, this method is capable of detecting unknown attacks, which make it desirable to many system administrators. Anomaly based systems are classified [9] to two subclasses, self-learning systems and programmed systems, which are explained in the next two subsections.

Self-learning Systems

This kind of systems, as the name suggests, are capable of building the normal baseline by observing the network traffic for a period of time. They can be either time series or non-time series [9].

1. Non-time series: These systems construct the normal system behavior using a stochastic model that does not take time series behavior into consideration. They can be either rule modeling or descriptive statistics.

   (a) Rule modeling: In ruled base method, the system studies the system behavior and creates a set of rules to describe the normal behavior of the system in the training mode. In order to detect an intrusion, the system applies these set of rules on the monitored traffic. The system raises an alarm if the observed traffic significantly deviates from this set of rules.

   (b) Descriptive statistics: In descriptive static method the system collects statistics for different system parameters. Then it considers a normal profile of the system as a vector of all these parameters. In order to detect an attack, the distance vector of the system profile and the observed behavior is calculated. If this distance is greater than a specified threshold, the observed behavior is labeled as an anomaly.

2. Time series: This approach takes time series behavior into consideration, which makes it really complex. Hidden Markov model (HMM) [53] and artificial neural network (ANN) [16] are some examples of this approach.

Programmed Systems

In the programmed approach, the system is not capable of building the normal profile. Instead, an individual, e.g., IDS administrator, programs the system, in a way that she

generates the normal profile of the system for the IDS. The IDS uses this profile to compare against the monitored behavior. Programmed systems are classified as follows [9]:

1. Descriptive statistics: In these systems, a profile of normal statistical behavior is created by using system parameters. Here, descriptive statistics on a number of parameters are collected to build the normal behavior profile. Such parameters can be the number of unsuccessful logins, the number of network connections, the number of commands with error returns, etc. These systems can use simple statistics approach, simple rule based approach or threshold approach.

   (a) Simple statistics: These systems are capable of making more abstract intrusion detection decisions since the collected statistics are used by higher level components..

   (b) Simple rule-based: In this approach, the normal system profile is a set of compound rules. The intrusion detection system applies these sets of rules on the collected statistics in order to detect an anomaly.

   (c) Threshold: Here, the normal system profile is just a set of predefined thresholds. This is the simplest intrusion detection approach. The system basically compares the collected statistics against the thresholds and raises an alert if a collected statistic is more than a predefined threshold. A good example of this method is raising a security alarm after a certain number of unsuccessful logins.

2. Default deny: In this approach a complete profile of normal behavior is built. This profile states explicitly all possible normal behaviors of the system. Any monitored behavior is flagged as an anomaly, unless it falls within the profile. The state series modeling is usually used for default deny. In state series modeling, the system normal profile is programmed as a set of states. Unlike expert systems where the transitions between states are explicit, here state transitions are implicit. Like all the state machines, as the monitored action carries on, the system transits from one state to another. Once the monitored action takes the system to any implied state that is not explicitly mentioned, the system raises an alarm. The monitored actions that can trigger transitions are usually security relevant actions such as file accesses (reads and writes), the opening of 'secure' communications ports, etc

2.6.3 Hybrid Techniques

Since hybrid intrusion detection systems analyze the system considering both the normal behavior of the system and the intrusive behavior of the intruder, they provide us with a more accurate result than any of anomaly or signature-based techniques alone. In other words, these detectors have a greater true positive rate and a lower false positive rate in the supervised system, since they know both the patterns of intrusive behavior and the normal behavior of the system.

These systems automatically are capable of finding out what represents an intrusive event and normal behavior by being presented with examples of normal behavior combined with intrusive behavior. One example of this system, RIPPER [39], operates by automatically determining what observable features are interesting when forming the intrusion detection decision, isolating them, and using them to form the intrusion detection decision later.

RIPPER [39] is a classification rule learning tool inspired by data mining for the automatic and adaptive construction of intrusion detection models. RIPPER uses auditing programs to monitor the system to set all the system parameters and features. These parameters describe the system behavior. Then, it uses data mining approaches to extract rules that accurately capture the behavior of intrusive and normal activities. These rules are then used for anomaly and signature based detection.

2.6.4 Honey Pots (HPs)

Honey Pots (HP) have recently gained their popularity in the academic and industrial community primarily due to their practical heuristic method of intrusion detection. HP uses the concept of bait and trap as a deception to trap undetected intruders. The passive approach of HP compliments the IDS as undetected intrusions could succumb to the bait. This combined effort has contributed greatly to reducing the amounts of undetected intrusions. HP's contribution to the IDS is not limited only to intrusion detections as any access to the bait is considered malicious and the activity is monitored. The following describes key benefits of HP:

1. The illusionary nature of the bait keeps the intruder busy and once the intrusion is confirmed, the system's security officer is alerted of the activity. This holds several other benefits:

(a) Valuable information on attacker tendencies can be gained depending on the bait used. Similar to studying mice in a maze, intruder activities are logged as the emulated environment is accessed.

(b) By delaying the attacker, HP will enable countermeasures to be applied within the response latency time. Usually attacks occur too quickly for any countermeasures to be applicable or to even trace the source of the intrusion.

(c) A believable emulated environment as bait helps divert the attention and time of the intruder. While busy, other resources are left untouched by the attacker.

2. HP increases the dependability of the IDS with its unbiased detection method.

3. A major advantage of the HP system is its processing and resource consumption. IDS monitors and analyzes every packet transactions thus consuming large processing resources and utilizes other systems involved in the data transfers. In comparison to the IDS, HP will behave like an interrupt IO, consuming resources solely when the HP system is utilized.

2.7 Conclusion

In this chapter, we presented the main issues in intrusion detection system. First, we gave some scenarios to illustrate the importance of network security. Then we demonstrated the fact that firewalls and other prevention tools can not be enough and there is a need for another line of defense when the actual intrusion is happening or has happened. We identified intrusion detection systems as the detection tool when an intruder bypasses the prevention tools. Afterwards, we presented some technical definitions and we discussed different categories of intrusion detection systems and how they function. Moreover, we talked about the state of the art in intrusion detection systems and the different approaches such as data mining techniques, artificial intelligence, agent based techniques, software engineering methods and the embedded system approaches. Besides, we classified all the different methods of analyzing the monitored data and discussed them in detail. Finally, we presented the new concept of honey pots, which can work in conjunction with intrusion detection systems to improve the intrusion detection performance.

Chapter 3

Introduction to Game Theory

3.1 Introduction

Game Theory, the formal modeling of conflict and cooperation, first emerged as a recognized field with a publication of John von Neumann and Oskar Morgenstern's *Theory of Games and Economic Behaviour* in 1944. Since then, game-theoretic thinking about choice of strategies and the interdependence of people's actions has influenced all the social sciences.

Game theory is the science of strategic decision making [30]. It is a powerful tool in understanding the relationships that are made and broken in the course of cooperation and competition. Games are characterized by a number of players or decision makers who interact, possibly threaten each other and form coalitions, take actions under uncertain conditions, and finally receive some benefit or reward or possibly some punishment or monetary loss. In this chapter, we first introduce some concepts in game theory. Then, we classify the games and discuss each one in detail. Finally, we illustrate the concept more by providing some examples.

3.2 What is game theory?

Game theory is a formal way to analyze interactions among a group of rational agents that behave strategically [15]. We discuss the terms used in this definition in the following:

1. **Group**: The decision makers in any game are called the players. The set of players in a game is referred to as group. If there is only one player in a game, the game becomes a decision problem.

2. **Interaction**: The decision each player makes in a group has an effect on at least one other player. Otherwise the game is simply a series of independent decision problems.

3. **Strategy**: Each player has a set of actions to choose from. This set of actions is referred to as the strategy of the player. Each strategy the player chooses affects the game.

4. **Rational**: Each player chooses her best action, considering the other players' decision . This condition can be weakened and we can assume that agents are *boundedly* rational. Behavioral economics analyzes decision problems in which agents behave *boundedly* rational. Evolutionary game theory is game theory with *boundedly* rational agents.

3.3 Taxonomy of games

There are several ways to categorize different types of games. Here, the games are classified as follows [30]:

1. Games of Skill
2. Games of Chance
    (a) Games Involving Risk
    (b) Games Involving Uncertainty
3. Games of Strategy
    (a) Two-Person
        i. Cooperative
            A. Purely Cooperative
            B. Minimal Social Situation
        ii. Mixed-Motive
        iii. Zero-Sum
    (b) Multi-Person
        i. Non-Cooperative

ii. Cooperative

The taxonomy of games is illustrated in Figure 3.1.

3.4 Games of skill

Games of skill involve solely one rational agent and consider nature as a certainty. Since, nature does not constitute a genuine second player, as in the case of games of chance, they are not really regarded as genuine games. A crossword puzzle, for example, is a game of skill. To elaborate, two fields that use the concept of games of skill are discussed in detail: Linear programming, optimization and basic results and the Lagrange method of partial derivatives.

3.4.1 Linear programming, optimization and basic results

In the mathematical domain of linear programming [8], a function is optimized in retrospect to the set Ω, the constraint set. This function would be designed to maximize and minimize respectively its output and its input. The constraint set would be sorted in an order according to criteria. Let $f : \Omega \rightarrow \mathbb{R}$ represent the constraint set where the maximizer $\omega \in \Omega$ is determined in order to maximize or minimize $f(\omega)$. Since optimization involves finding the local maxima and local minima of functions (collectively called optima), differential calculus is often a strong candidate as an instrument for solving problems.

3.4.2 The Lagrange method of partial derivatives

Multivariable functions can be optimized using Lagrange's method of partial derivative [18]. This method can be explained in the following manner: *Lagrangianfunction*, Λ is expressed as:

$$\Lambda(x,y,\lambda) = f(x,y) + \lambda[c - g(x,y)]$$

where c is constant; $f(x,y)$ is the function to optimize and $g(x,y)$ is the constraint function. All partial derivatives should be equal to zero, so that the optimization solution could be found, which is analogous to the first-order test for stationary points:

$$\frac{\delta(\Lambda)}{\delta(x)} = \frac{\delta(\Lambda)}{\delta(y)} = \frac{\delta(\Lambda)}{\delta(\lambda)} = 0$$

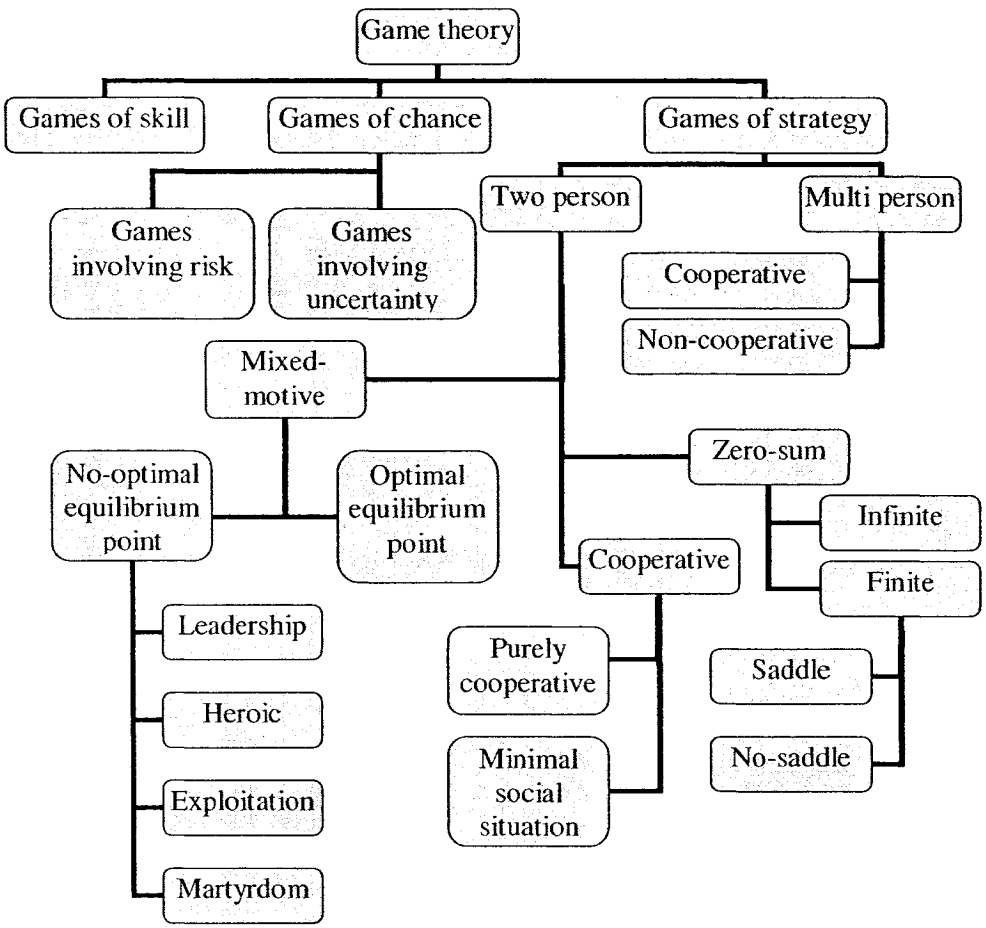

Figure 3.1: Taxonomy of games [30]

In other words, the following equations must be solved for x, y and λ:

$$\frac{\delta f}{\delta x} = \lambda \frac{\delta g}{\delta x}$$

$$\frac{\delta f}{\delta y} = \lambda \frac{\delta g}{\delta y}$$

$$g(x,y) = c$$

in which case all the values that produce maxima and minima for $f(x,y)$, subject to $g(x,y) = c$, will be contained in the solution set.

3.5 Games of chance

Still consisting of a single rational agent, in games of chance [43] unpredictability is added to the situation and affects the outcome. Games of chance are also referred to as decision theory [30]. They either involve *risk* (decision theory under certainty), where the probability of nature's response is known; or involve *uncertainty* (decision theory under uncertainty), where the probability of nature's response is not known.

A decision problem $(A, \preceq)$ consists of a finite set of outcomes $A = \{a_1, a_2, ..., a_n\}$ and a preference relation $\preceq$. The expression $a \preceq b$ should be interpreted as "b is at least as good as a". We expect the preference relation to fulfill two simple axioms:

Axiom 3.5.1 *Completeness. Any two outcomes can be ranked, e.g.* $a \preceq b$ *or* $b \preceq a$.

Axiom 3.5.2 *Transitivity implies that if* $a \preceq b$ *and* $b \preceq c$ *then* $a \preceq c$.

The axioms guarantee that all outcomes are ordered gapless and cycleless. Albeit convenient, it often better to use a utility function $u : A \rightarrow \mathbb{R}$, since only n real numbers $\{u_1, u_2, ..., u_n\}$ need to be monitored.

Definition 3.5.1 *Utility function* $u : A \rightarrow \mathbb{R}$ *is consistent with the preference relationship of* $(A, \preceq)$ *if for all* $a, b \in A$: $a \preceq b$ *iff* $u(a) \leq u(b)$.

Using the previous knowledge, we can now clearly identify a rational agent:

Definition 3.5.2 *A rational agent as an agent faced with problem* $(A, \preceq)$ *maximizes the utility function by selecting* $a^* \in A$. *This is better expressed as for each* $a \in A$, *we have* $a \preceq a^*$).

3.5.1 Games of chance involving risk

Games of chance involving risk are resolved based on outcome probabilities. This gives rise to the notions of utility theory and expected utility value.

Utility theory guides the rational agent according to valuable outcomes instead of objective outcomes.

Let $U(c)$ be the expected utility value of a choice c, for a continuous distribution function:

$$U(c) = \Sigma_{i=1}^{n} p_i u_i$$

where u_i is called Von Neumann-Morgenstern utility function [45] and represents the player's preferences among her expected values, and p_i is the probability of choice i.

3.5.2 Game of chance involving uncertainty

Games of chance involving uncertainty are unpredictable to the rational agent. Three principles have been developed for such games: the maximax principle, the maximin principle and the minimax principle.

1. Maximax Principle: the rational agent will make a decision based on the greatest profit while disregarding the risks involved.

2. Maximin Principle (Wald,1945): the rational agent's decision will avoid the worst scenario that has the minimum utility.

3. Minimax Principle (Savage, 1954): combines the previous two principles to dictate that the rational agent avoids the largest regret. Regrets are defined as the possible alternatives that could have been chosen if unpredictability were not present.

3.6 Games of strategy

Also considered as the normal form games, games of strategy consist of the following elements:

1. A player list $D = \{1,2,...,I\}$. We mostly consider games with just two players. As an example consider two people A and B, who want to meet.

2. Each player i can choose actions from a strategy set S_i. To continue our example, each of the players has the option to go to Spot C or D . So the strategy sets of both players are $S_1 = S_2 = \{C,D\}$.

3. The outcome of the game is defined by the 'strategy profile' which consists of all strategies chosen by the individual players. For example, in our game there are four possible outcomes - both players meet at C, (C,C), they mis-coordinate, (C,D) and (D,C), or they meet at D (D,D). Mathematically, the set of strategy profiles (or outcomes of the game) is defined as

$$S = S_1 \times S_2 \tag{1}$$

   In our case, the order of S is 4. If player 1 can take m possible actions, and player 2 can take n possible actions, the set of profiles has order $m \times n$.

4. Players have preferences over the outcomes of the play. You should realize that players can not have preferences over the actions. In a game the payoff depends on the set of actions. In the example above, players just want to be able to meet at the same spot. They do not care if they meet at C or at D. As long as the players choose the same spot, they are fine. If one chooses one spot and the other player chooses the other spot, then they are unhappy. So what matters to players are the outcomes, not the actions (of course their actions influence the outcome - but for each action there might be many possible outcomes - in our example there are two possible outcomes per action). Recall, that we can represent preferences over outcomes through a utility function. Mathematically, preferences over outcomes are defined as:

$$u_i : S \rightarrow \mathbb{R} \tag{2}$$

   In our example, $u_i = 1$ if both agents choose the same action, and 0 otherwise.

All this information can be conveniently expressed in a game matrix as shown in Figure 3.2. A more formal definition of a game is given below:

Definition 3.6.1 *A strategic game G consists of [43]:*

1. *A finite set of agents* $D = \{1,2,...,I\}$.

2. *Strategy sets* $S_1, S_2, ..., S_I$.

	E	C
E	(1,1)	(0,0)
C	(0,0)	(1,1)

Figure 3.2: Example of a general two player game

3. *Payoff functions* $u_i : (S_1 \times S_2 \times ... \times S_I) \rightarrow \mathbb{R}$.

We will write $S = S_1 \times S_2 \times ... \times S_I$ and we call $s \in S$ a strategy profile where $s = (s_1, s_2, ..., s_I)$. We present the strategy choices of all players except player i with s_{-i} which is $(s_1, s_2, ..., s_{i-1}, s_{i+1}, ..., s_I)$.

3.7 Two person cooperative games

In cooperative two-person games, both players have common goals and therefore there is no conflict. These games are interesting because of their decision making process. The two rational agents can be considered as a single rational agent since they cannot be distinguished by their desired outcomes. Such games can be divided into two classes: purely cooperative games and minimal social situation games [26].

3.7.1 Purely cooperative games

In purely cooperative games, the interests of both players coincide perfectly. The rational agents agree on the preference order of the outcomes. Each rational agent formulates strategies based on the other's strategies, which can be obtained either by direct information or by extrapolation. The games can be ones of either perfect or imperfect information. In games of complete information, all players know the rules of the other game and the preferences of the other players. On the other hand, the condition of complete information does not apply in games of incomplete information. In these types of games, no unique best combination of strategies exists and a solution must be sought through informal analysis.

3.7.2 Minimal social situation games

Minimal social situation games imply that the rational agents are left ignorant of all information other than the available choices. It is important to note that simultaneous and sequential decisions differ in minimal social situation games as sequential decisions never lead to a mutually beneficial outcome.

The authors in [29], proposed a principle of rational choice for minimal social games known as the "win-stay, lose-change" principle. This principle states that, if a player makes a choice which produces a positive pay-off, the player will repeat that choice. On the other hand, if a player makes a choice that produces a negative pay-off, the player will change strategy. Thus, the strategies that produce positive pay-offs are reinforced and those that produce negative pay-offs are not.

3.8 Zero-Sum Games

The two rational agent zero-sum game [43] is strictly competitive as the pay-offs sum up to either zero or a constant. An important aspect of this type of game is that there will always be exactly one winner and one loser.

Game theory is particularly well-suited to the analysis of zero-sum games and application to everyday life. Actually a "constant-sum-game" would be a better title, since in some circumstances, the pay-offs do not add up to zero because the game is unfair. However they do sum to a constant, which is the prevalent feature of these strictly competitive games. The term zero-sum is used even in these instances, for the sake of simplicity.

A two person zero-sum game is defined as a 3-tuple (X,Y,M), where X and Y are sets of possible strategies for the two players and M is real valued utility function defined on the Cartesian product $X \times Y$. The set X is called the set of admissible pure strategies of player 1 and set Y is called the set of admissible pure strategies of player 2. The function M is called the pay-off function of player 1. Player 1 chooses a strategy x of the set X while player 2 chooses a strategy y of the set Y. The choices are done simultaneously and independently and the chosen x and y determine the pay-off $M(x,y)$ to player 1 and $-M(x,y)$ to player 2. The data of the game (X,Y,M) are known to both players.

A pair (x^*,y^*) of strategies is called an equilibrium (a saddle-point) if the following conditions hold:

$$M(x,y^*) \leq M(x^*,y^*) \leq M(x^*,y) \quad forany \quad (x,y) \in X \times Y.$$

	H	T
H	1	-1
T	-1	1

Figure 3.3: Example of matrix game

The pair (x^*,y^*) is called an equilibrium, the strategies x^* and y^* of the players are called optimal, and $M(x^*,y^*)$ is called the value of the game. It is clear that if player 1 plays x^* then she can win at least $M(x^*,y^*)$ no matter what player 2 plays. Likewise if player 2 plays y^* then she can win at most $M(x^*,y^*)$ no matter what player 1 plays.

The special case in which X and Y are finite is called a finite game or a matrix game. In this case, the function M for the game (X,Y,M) is described as a pay-off matrix A whose rows are labeled by the elements of X (usually dented as $1,...,n$) and the columns by the elements of Y (denoted as $1,...,m$). An example of matrix game is illustrated in Figure 3.3. The game can be described as follows. Two players choose tail or head. If both of them choose the same side of the coin then player two pays one dollar to player 1. Otherwise player 1 pays one dollar to player 2. Not every game has a saddle point. In fact the above game has none.

In view of the non-existence of the value for matrix games, it is suggested that a player can sometimes do better by choosing her strategy randomly. For example if a player chooses head and tail with equal probabilities, then her expected pay-offs will be zero, independently of the behavior of the other player. This motivates the following definition.

The mixed extension of a matrix game (X,Y,M) is the game $(\overline{X},\overline{Y},\overline{M})$ where:

$$\overline{X} = \{x \in R^n : x_i \geq 0 \quad for\, i \in [1,n], \sum_{i=1}^{n} x_i = 1\}$$

$$\overline{Y} = \{y \in R^m : y_i \geq 0 \quad for\ \ i \in [1,m], \sum_{i=1}^{m} y_i = 1\}$$

and

$$\overline{M}(x,y) = \sum_{i=1}^{n}\sum_{j=1}^{m} A_{i,j}x_i,y_j$$

So, the strategy set of players in the mixed extension is the set of probability distributions on

	R	P	S
R	(0,0)	(-1,1)	(1,-1)
P	(1,-1)	(0,0)	(-1,1)
S	(-1,1)	(1,-1)	(0,0)

Figure 3.4: Rock-Paper-Scissors game

the strategy set of the original game. The elements of these sets are called mixed strategies. The pay-off in the mixed extension is just the expected pay-off of the player. It is clear that the extreme points of set $\overline{X}$ or $\overline{Y}$ can be identified with the strategy sets X, Y.

3.8.1 Nash equilibrium

Definition 3.8.1 *Strategy profile s^* is defined as a pure strategy Nash equilibrium of G if and only if:*

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*) \tag{3}$$

Definition 3.8.2 *A pure strategy Nash equilibrium is defined as strict if the following inequality is true:*

$$u_i(s_i^*, s_{-i}^*) > u_i(s_i, s_{-i}^*) \tag{4}$$

3.8.2 The advantage of mixed strategies

Randomizing and playing mixed strategies prevents the enemy's extrapolation of a rational agent's strategic pattern and thus favoring victory.

Consider the following Rock-Paper-Scissors (RPS) game as shown in Figure 3.4. Note that RPS is a zero-sum game. This game has no pure-strategy Nash equilibrium. Whatever pure strategy player 1 chooses, player 2 can beat him. A natural solution for player 1 might be to randomize amongst her strategies.

Another example of a game without pure-strategy NE is matching pennies, see Figure 3.5. As in RPS, the opponent can exploit her knowledge of the other player's action, fearing what might the opponent do. One solution is to randomize and play a mixed strategy. Each player could flip a coin and play H with probability $\frac{1}{2}$ and T with probability $\frac{1}{2}$.

	H	T
H	1	-1
T	-1	1

Figure 3.5: Matching penneis

Note that each player cannot be taken advantage of.

Definition 3.8.3 *Let G be a game with strategy spaces $S_1, S_2, .., S_I$. A mixed strategy σ_i for player i is a probability distribution on S_i, i.e., for S_i finite a mixed strategy is a function $\sigma_i : S_i \to \mathbb{R}^+$ such that $\sum_{s_i \in S_i} \sigma_i(s_i) = 1$.*

Several notations are commonly used for describing mixed strategies.

1. Function (measure): $\sigma_1(H) = \frac{1}{2}$ and $\sigma_1(T) = \frac{1}{2}$.
2. Vector: If the pure strategies are $s_{i_1}, .., s_{i_N}$ write $(\sigma(s_{i_1}), ..., \sigma(s_{i_N}))$, e.g., $(\frac{1}{2}, \frac{1}{2})$.
3. $\frac{1}{2}H + \frac{1}{2}T$

3.8.3 Mixed strategy Nash equilibrium (MSNE)

Define Σ_i (also $\Delta(S_i)$) for the set of probability distributions on S_i.

Define Σ for $\Sigma_1 \times ... \times \Sigma_I$. A mixed strategy profile $\sigma \in \Sigma$ is an I-tuple $(\sigma_1, ..., \sigma_I)$ with $\sigma_i \in \Sigma_i$.

Definition 3.8.4 *A mixed strategy NE of G is a mixed profile $\sigma^* \in \Sigma$ such that*

$$u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(\sigma_i, \sigma_{-i}^*) \tag{5}$$

for all i and all $\sigma_i \in \Sigma_i$.

3.8.4 Testing for MSNE

The definition of MSNE makes it cumbersome to check that a mixed profile is a NE. The next result shows that it is sufficient to check against pure strategy alternatives.

Proposition 3.8.1 *σ_i^* is a Nash equilibrium if and only if*

$$u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(s_i, \sigma_{-i}^*) \tag{6}$$

for all i and $s_i \in S_i$.

Example 3.8.1 *The strategy profile $\sigma_1^* = \sigma_2^* = \frac{1}{2}H + \frac{1}{2}T$ is a NE of Matching Pennies.*

Because of the symmetry it is sufficient to check that player 1 would not deviate. If he plays her mixed strategy he gets expected payoff 0. Playing her two pure strategies gives him payoff 0 as well. Therefore, there is no incentive to deviate.

Note: Mixed strategies can help us to find MSNE when no pure strategy NE exists.

3.9 Two-person mixed-motive games of strategy

Two-person mixed-motive games [30] can be considered as the middle ground between cooperative and zero-sum games as the sum of the pay-offs vary dependently on the strategies. In a mixed-motive game, the sum of the pay-offs differs from strategy to strategy, so they are sometimes called *variable-sum* games, although the term is not strictly accurate since cooperative games are also variable. They rarely produce pure solutions, but they are interesting for the real-time situations they represent and for providing an insight into the nature of conflict resolution.

3.9.1 Mixed-motive games and the Nash equilibrium

Represented in an ordinal fashion, mixed-motive games display the two rational agents' pay-offs as coordinate pairs and are defined as the following:

1. The first rational agent has a finite set of strategies: $S_1 = \{r_1, r_2, ..., r_m\}$ where $|S_1| = m$

2. The second rational agent has a finite set of strategies: $S_2 = \{c_1, c_2, ..., c_m\}$ where $|S_2| = n$

3. The rational agents pay-offs are the utility functions u_1 and u_2. The outcome of r and c defines the pay-off for the first rational agent as $u_1(r, c) \in S_1 \times S_2$.

The domination aspect of strategy r_i over another strategy r_j for the first rational agent is determined by the following relationship: $u_1(r_i,c) \geq u_1(r_j,c)$, $\forall c \in S_2$

The dominance is considered *strict* if: $u_1(r_i,c) > u_1(r_j,c)$, $\forall c \in S_2$ The dominance is considered *weak* if: $u_1(r_i,c) \geq u_1(r_j,c)$, $\forall c \in S_2$

A pair of strategies $(r_N,c_N) \in S_1 \times S_2$ is considered as an *Nash equilibrium* if the following holds true:

1. $u_1(r_N,c_N) \geq u_1(r,c_N)$, $\forall r \in S_2$
2. $u_2(r_N,c_N) \geq u_2(r_N,c)$, $\forall c \in S_2$

Strategic pairs can be determined if a unique saddle-point exists. In the situation where none exist, mixed strategies are used and in the situation where multiple saddle-points exist, we classify each equilibrium point according to a set of archetypes.

Archetype1-leadership games

In such games, no dominant strategies can give the most profitable outcome to each rational agents. This fact leads to the conclusion that the minimax principle is inapplicable as regret is generated when the worst-case pay-off is chosen and the opponent's choice is known. The leadership concept is understood as the outcome has two equilibrium points in relation to the leader. The choice of the first rational agent will inadvertently affect the optimal choice of the second rational agent to be the opposite choice. Extrapolation is a key factor as the game has no constant value therefore open communication is favored for all rational agents to deviate from the minimax principle. Figure 3.6 shows an example of leadership games. It can be seen from the matrix that there are no dominant or inadmissible strategies. The minimax principle fails too because both candidates should choose their first strategy so as to avoid the worst pay-off (1,1). Yet, if they do this, both candidates regret it once the other's choice becomes known. Hence, the minimax strategies are not in equilibrium and the solution (2,2) is not an equilibrium point. It is unstable and both players are tempted to deviate from it, although it should be pointed out that the worst case scenario is when both deviate from it (1,1).

Yet, there are two equilibrium points. If player 1 chooses the second column, player2 can do no better than choosing first column; and if player 1 chooses the second column, player2 can do no better than choosing first column. So, there are two equilibrium points, those with pay-offs (4,3) and (3,4).

(2,2)	(3,4)
(4,3)	(1,1)

Figure 3.6: An example of leadership games

Archetype 2- heroic games

Same as the leadership games but the choice of the rational agent is made to be convincing as it displays deeply unselfish motives by benefiting the opponent more.

Archetype 3- exploitation games

Similar to the previous archetypes, the rational agent deviating from the minimax principle stands to be the sole benefactor in such decisions and embarks both rational agents towards disaster if the choice is incorrect.

Archetype 4- martyrdom games

The mutual benefiting intent causes both rational agents to deviate from the minimax principle but a defecting rational agent will guarantee success. The most famous example here is the *prisoner's dilemma* game, as shown in Figure 3.7, so-called in 1950 by A. W. Tucker. It is the most famous and most analyzed game in game theory and the example below is a variation on that well-known theme. This game is a genuine paradox. The minimax strategies intersect at (2,2). Unlike the other three prototype games above, this minimax solution does form an equilibrium point. It can be seen that for both players the second strategy dominates.

However, this dominant solution is worse than the strategy (3,3). It appears that there is a conflict between individual self-interest and collective self-interest. Furthermore, the latter strategy where both players optimize their collective pay-offs, (3,3), is unstable itself since each player is tempted to deviate from it.

Games such as this are called *martyrdom* games because if both players deviate from

(3,3)	(1,4)
(4,1)	(2,2)

Figure 3.7: An example of martyrdom games

the minimax strategy, they are doing so to benefit others as much as self. And yet, the martyr who defects from this mutuality of martyrdom will always win.

3.10 Multi-person games

With more than two rational agents, multi-person games can be greatly affected by the formation of coalitions between rational agents. In the event of coinciding interests in coalitions, the game can be seen as a two-person cooperative game but in case of zero-sum multi-person games saddle points are determined.

Non-cooperative multi-person games and a more realistic situation such as partially cooperative and mixed-motive game are discussed in detail in the following:

3.10.1 Non-cooperative multi-person games

In non-cooperative multi-person games [30], coalition formation is not an option as communication between rational agents can be unwise or even impossible depending on the situation.

Finite non-cooperative multi-person games were proved by Nash in 1951 to containe minimally one Nash saddle-point in pure or mixed strategies. These points represent the outcomes that produces no regret to any rational agent involved and are frequently non-equivalent (different pay-offs) and non-interchangeable (unique to a particular rational agent.

3.10.2 Mixed-motive multi-person games

A mixed motive multi-person game [30] with n rational agents can be described as follows:

1. Each rational agent i contains a finite set of strategies S_i, $\forall i \in \{1,2,...,n\}$
2. Each rational agent i contains a pay-off utility function $u_i \in S_1 \times S_2 ... \times S_n \rightarrow \mathbb{R}$

Each rational agent i picks at the same time a strategy $s_i \in S_i$ and produces the pay-off u_i. Each pay-off is based on all n strategies therefore strategies $S_1, S_2, ..S_n$ and pay-off functions $u_1, u_2, ..., u_n$ are required to be understood beforehand.

A Nash saddle-point for mixed-motive multi-person games is defined as the set of strategies $\{s_{N1}, s_{N2}, ...s_{Nn}\}$ where

$u_i(s_{N1}, s_{N2}, ...s_{Nn}) \geq u_i(s_1, s_2, ...s_n)$, $\forall s_i \in S_i$

In this situation, Nash equilibrium strategy is best suited for all rational agents if all rational agents follow the same type of strategy and that strategies are made public.

Nash saddle-points have the solution $s_{N1}, s_{N2}, ...s_{Nn}$ as:

$\frac{\delta u_i}{\delta s_i}(s_{N1}, s_{N2}, ...s_{Nn}) = 0$

and $\delta u_i / \delta s_i = 0$ contains only a single solution, then there exist a unique Nash saddle-point. Also, each s_{Ni} will be the only stationary point of the function u_i and

$\frac{\delta^2 u_i}{\delta^2 s_i} < 0$, $\forall i$

will indicate a local maximum.

3.11 Some classical examples

3.11.1 Matching Pennies

Zero-sum games are true games of conflict. Any gain on one side comes at the expense of the opponents. Think of dividing up a pie. The size of the pie doesn't change - it's all about redistribution of the pieces between the players (tax policy is a good example). The simplest zero sum game is matching pennies, Figure 3.8. This is a two player game where player 1 get a Dollar from player 2 if both choose the same action, and otherwise loses a Dollar.

3.11.2 Battle of the Sexes

This game is interesting because it is a coordination game with some elements of conflict. The idea is that a couple want to spend the evening together. The wife wants to go to the

	H	T
H	(1,-1)	(-1,1)
T	(1,-1)	(1,-1)

Figure 3.8: Matching pennies

	F	O
F	(2,1)	(0,0)
O	(0,0)	(1,2)

Figure 3.9: Battle of sexes

Opera, while the husband wants to go to a football game. Each gets at least some utility from going together to at least one of the venues, but each wants to go their favorite one (the husband is player 1 - the column player), see Figure 3.9.

3.11.3 Chicken or Hawk versus Dove

This game is an anti-coordination game. The story is that two teenagers drive home on a narrow road with their bikes, and in opposite directions. None of them wants to go out of the way. Whoever 'chickens' out loses her pride, while the tough guy wins. But if both stay tough, then they break their bones. If both go out of the way, none of the them is too happy or unhappy. The game is shown in Figure 3.10.

	T	C
T	(-1,-1)	(10,0)
C	(0,10)	(5,5)

Figure 3.10: Chicken or hawk versus dove

3.12 Conclusion

We discussed game theory concepts to provide knowledge background for the next chapter where we introduce a game theoretic approach for intrusion detection. Here, we presented a complete taxonomy of different types of games, and we discussed each subclass in detail. Moreover, we introduced the concept of mixed strategies and Nash equilibrium. Mixed strategies come in handy when there is no answer to a problem in fixed strategies. We use these concepts in the next chapter.

Chapter 4

A Game Theoretic Model for Detecting Network Intrusions under Different Scenarios

4.1 Introduction

Security of computer and network systems is becoming increasingly important as more and more sensitive information is being stored, transmitted, and manipulated online [31]. Two key areas of concern in system security are intrusion detection and intrusion prevention which have been extensively investigated in the research community over the past decade. Currently, Intrusion Detection Systems (IDSs) [54] have become a critical technology for protecting and defending networks and computer systems against malicious attacks. Network intrusion takes many forms including denial of service attacks (DoS) [50], viruses introduced into the networks, etc. Typically, in an intrusion problem, the intruder attempts to gain access to a particular file server or web site in the network. A stylized intrusion problem is where an intruder attempts to send a malicious packet to a particular node in the network and the network then attempts to detect this intrusion [32].

Most of the earlier work on intrusion detection relies on ad-hoc schemes and experimental work [6]. Therefore, in order to address issues like attack modeling, analysis of detected threats, and decision on response actions, there is a need for a quantitative decision and control framework. Currently, various tools have been developed within the game theory discipline to address problems where multiple players with different objectives compete and interact with each other on the same system. These tools are successfully used

in many disciplines including economics, political science, and control. Now, given the continuous struggle between attackers who aim to penetrate the deployed systems and security administrators trying to protect these systems, these interactions can be modeled as a non-cooperative game [43], where the players are the intruders and the intrusion detection system. Therefore, game theory is a strong candidate to provide the much-needed mathematical framework for analysis, modeling, decision, and control process for information security and intrusion detection [7]. As a result, game theory has been lately proposed by several studies for a theoretical analysis of IDS [3, 6, 7, 32, 40].

Recently, the authors of [32] have studied the problem of intrusion detection through packet sampling and formulated the problem using game theory. They considered an attack wherein the intruder uses only one packet to carry out her task. However, a well trained intruder may choose to split her attack over multiple packets each possibly traversing a different route. To the best of our knowledge, none of the previous studies have considered this more practical problem. Our contributions in this chapter are the following:

1. To build a game theoretic framework to model network intrusions through multiple packets. Detection is accomplished by sampling a portion of the packets transiting selected network links while not exceeding the budget constraint.

2. To investigate the case where we have a group of cooperative intruders. The intruders initiate the attack by sending a series of malicious packets from different nodes. We build a game theoretic model in order to detect these network intrusions by sampling a subset of the transmitted packets over selected links. To the best of our knowledge, there has not been any study for the case where the attack is distributed over multiple intruders using game theory.

Our work aims at developing a network packet sampling policy to effectively reduce the success chances of an intruder by finding the value of the game using a min-max strategy [43]. Non-cooperative game theory will be used to formally express our problems, where the players are: (1) the cooperative intruders or a smart intruder (depends on which scenario we are solving) and (2) the intrusion detection system. This game theoretic model will guide the IDS to have an optimal sampling strategy in order to detect the malicious packets. The strategy for each intruder is the probability of choosing each possible path to send its malicious packet to the victim node. Consequently, the optimal strategy for the IDS is to assign the sampling rates to each link to maximize the probability of detection while not exceeding the total predetermined budget.

The rest of this chapter is organized as follows. Section 4.2 overviews the related work. In Section 4.3, we consider the first scenario where a malicious node distributes the attack over multiple packets. We present the problem statement and then illustrate the assumptions. Next, we introduce the game and discuss the constraints and objective of the game. Sections 4.4 and 4.5 present the game formulation and solution respectively. Furthermore, a case study is done to show how the game formulation works in a practical network. In Section 4.6, we investigate the case where at "least half" of the malicious packets are needed to detect the attack. Sections 4.7 and 4.8 present the second scenario where a distributed attack is launched via cooperative malicious nodes. First, a game theoretic framework is built and then the solution of the game is introduced providing strategies for both the IDS and the intruders. A case study is presented as well. Section 4.9 discusses the game results through simulations, which is followed by the concluding remarks of Section 4.10.

4.2 Related Work

In Alpcan and Basar [7], the authors presented a game-theoretic analysis of intrusion detection in access control systems. In order to establish a quantitative mathematical framework, they modeled the interaction between the attacker and the IDS as both finite and continuous-kernel non-cooperative security games. They modeled the imperfect flow of information from the attacker to the IDS through a virtual sensor network based on software agents. The interaction between the attacker and the IDS was formulated as a non-cooperative non-zero sum game with the virtual sensor network as a third fictitious player. Existence of a unique Nash equilibrium and best-response strategies for players under specific cost functions was also investigated. Then, the authors extended the model to take the dynamic characteristics of the sensor network [5] into account. They modeled this through analyzing the interaction between the players over a time period using repeated games. Finally, they discussed properties of the resulting dynamic system and repeated games both analytically and numerically where through these numerical studies, some basic strategies for the IDS and the attacker were proposed.

In [6], the authors aimed at demonstrating the suitability of game theory for development of various decision, analysis, and control algorithms in intrusion detection. They accomplished this by addressing some of the basic network security tradeoffs, and giving illustrative examples in different platforms. Therefore, they proposed two different

schemes, based on game theoretic techniques. They considered a generic model of a distributed IDS with a network of sensors. First, the authors devised a flexible scheme using intrusion warning levels with the IDS being able to operate in different modes at each security level, and to switch automatically between the different levels. To deal with this issue they used cooperative game theory and Shapely value [17]. The security warning system is simple and easy to implement, and it has given system administrators an intuitive overview of the security situation in the network. Furthermore, the authors modeled the interaction between the attacker and the IDS as a two-person, non-zero sum, single act, finite game with dynamic information. They proposed two specific sub-games and Nash equilibrium solutions in closed forms were obtained for these specific sub-games. Nash equilibrium solutions were derived analytically and analyzed for the defined security game in the two special cases.

In [40], Liu et al. proposed a game theoretic approach for estimating the attacker's intent, objective, and strategies (AIOS). They developed a game theoretic AIOS formalization which could capture the inherent inter-dependency between AIOS and defender objectives and strategies in a way that AIOS could be automatically inferred. They presented an incentive-based conceptual framework for AIOS modeling. Then, the authors developed a game theoretic formalization of the conceptual framework. They used the concept of utilities since it was capable of integrating incentives and costs in such a way that attacker objectives could be practically modeled. Finally, they used a specific case study to show how AIOS could be inferred in real world attack-defense scenarios.

In [3], Agah et al. pointed out the insufficiency of resources in sensor networks as the motivation of their study. They proposed a game theoretic framework for defending nodes in a sensor network since they believed it would consume fewer resources. Their main concern was finding the most vulnerable node in a sensor network and protecting it. They applied three different schemes for defense; game theory, Markov Decision Process (MDP) and an intuitive metric (node's traffic). In the first scheme they formulated the attack-defense problem as a two-player, nonzero-sum, non-cooperative game between an attacker and a sensor network. They showed that this game achieved Nash equilibrium and thus leading to a defense strategy for the network. They considered many risk factors like reliability of a sensor node, different types of attacks, and past behavior of the attacker. Simulation results showed that using a game theoretic framework significantly improved the chance of intrusion detection.

The work of Kodialam and Lakshman [32] has inspired and motivated our study. They

have considered the problem of detecting intruding packets in a network by means of network packet sampling. Since packet sampling and examination in real-time could be expensive, the network operator had to devise an effective sampling scheme to detect intruding packets injected into the network by an adversary. They take into consideration the scenario where the adversary has significant information about the network and can either pick paths to minimize chances of detection or could pick suitable network ingress-point if only shortest path routing was allowed. They have formulated the problem in a game-theoretic framework and the solution to this problem was a max-flow problem from which the stable operating points were obtained. However, their approach is not practical when a practiced intruder or cooperative intruders divide the attack over multiple packets and transmits them through possibly different routes. We particularly take into account these special problems, and use the same network model as in [32]. We then build our game-theoretic models and formulate the sampling problem. We solve the games using min-max approach to find the optimal sampling strategy for the IDS in order to detect these intrusion packets that are launched either by a smart intruder or by cooperative intruders.

4.3 Problem Statement

The problem set-up is outlined in four steps. First, we discuss the assumptions in the network. Then, we introduce the game defining the adversaries in a game theoretic framework. Afterward, we describe the objective of the game that is played between the adversaries and finally we introduce strategies for the two players.

4.3.1 Network Model and Assumptions

The network is modeled as a directed graph, $G = (N, E)$ where N is the set of nodes and E is the set of unidirectional links. It is also assumed that there are k nodes and l links in the network. The capacity of link $e \in E$ is denoted by c_e and the amount of traffic flowing on link e is represented by f_e. Given two nodes u and v in the network, let ρ_u^v represent the set of paths from u to v in G. We present the maximum flow between u and v with $MF_u^v(c)$, where c is the capacity vector. Corresponding to the maximum flow between nodes u and v, there is a minimum cut [49] consisting of a set of links in the network. The set of links in this minimum cut will be represented by $Mincut_u^v$. We also introduce the maximum flow among all links in all paths in ρ_u^v by $max_u^v(f) = Max\{f_e | \forall e \in P, \forall P \in \rho_u^v\}$. In the first scenario, a malicious user can split an attack over n packets each containing a fragment

of the attack. Here, we call these packets *a-fragments*. We have a distributed intrusion detection system that is detecting attacks over multiple packets and the intrusion is being detected if a fraction of the *a-fragments* is being sampled. The IDS can detect the intrusion if m *a-fragments* are being detected where $m \leq n$. For the second scenario, we introduce Ω ,be the set of cooperating intruders, each sending a packet to the target node t, in order to initiate the attack, where $|\Omega|$ is the number of intruders. Furthermore, we introduce s_e to b the sampling rate on link e. It is obvious that $s_e \leq f_e$, i.e., the sampling rate on a link is less or equal than the actual flow on the very link. Knowing that in practical networks the sampling rate is an integer value, we state that $s_e < f_e - 1$.

4.3.2 Introducing the Games

In the first scenario, we assume that the game is played on an infrastructure-based network between two players: The IDS and the intruder. The objective of the intruder is to inject n *a-fragments* from some attacking node $a \in N$ with the intention of attacking a target node $t \in N$. An intrusion is successful when at least m *a-fragments* out of the n *a-fragments* reach the desired target node, t, without detection. In order to detect and prevent the intrusion, the IDS is allowed to sample packets in the network. Without loss of generality, it is assumed that sampling takes place on the links in the network. The game is pictorially illustrated in Figure 4.1-a. In the second scenario, the game is played on an infrastructure-based network between the IDS and the cooperating intruders. Assuming the set of cooperative intruders as one player, we model the game as a zero-sum game: the IDS and the intruders. The objective of each intruder $x \in \Omega$ is to send an *a-fragment* to the target node t. An intrusion is successful when at least m *a-fragments* out of the $|\Omega|$ *a-fragments* reach the desired target node t without detection. In order to detect the intrusion, the IDS samples packets in the network via its agents. Furthermore, the agents sample the traffic on each link in the network as shown in Figure 4.1-b.

4.3.3 Game objectives and constraints

Sampling all the packets flowing on a link and examining these packets can be fairly expensive to perform in real time. Therefore, we assume that the IDS has a sampling budget of B_s packets per second over the entire network. This sampling effort can be distributed arbitrarily over the links in the network. Here, we assume a distributed agent based IDS [22]. The IDS samples the packets on each link via the agents while not exceeding the sampling

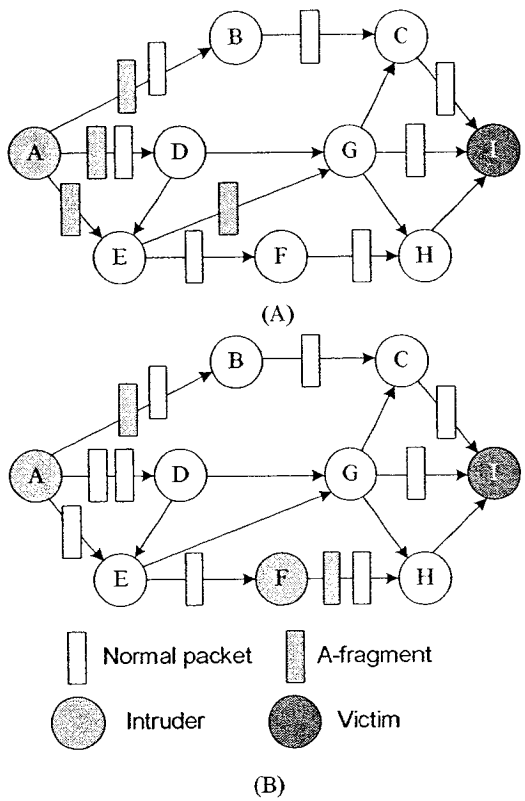

Figure 4.1: Single intruder and cooperative intruder games

budget, B_s. The sampling bound can be viewed as the maximum rate at which the intrusion detection system can process packets in real time. If a link e, with traffic f_e flowing on it, is sampled at rate s_e, then the probability of sampling a malicious fragment on this link is given by $p_e = s_e/f_e$. Therefore, we have the sampling constraint $\sum_{e\in E} s_e \leq B_s$. The game theoretic problem that we are going to discuss in the next sections, is formulated in terms of p_e. We assume that all the players have complete information about the topology of the network and all the link flows in the network.

4.3.4 Strategies for the two players

In the case of the intruder, in the two scenarios, a pure strategy would be to pick a path $P \in \rho_x^t$ for the malicious packet to traverse from x to t. The intruder, in our case, can use a mixed strategy. In the case of a mixed strategy, the intruder has a probability vector $q_x = (q(P_{1_x}), ..., q(P_{z_x}))$ over the set of paths in $\rho_x^t = \{P_{1_x}, P_{2_x}, ..., P_{z_x}\}$ such that $\sum_{P\in\rho_a^t} q(P) = 1$. Moreover, let $V_x = \{q : \sum_{P\in\rho_x^t} q(P) = 1\}$ represent the set of feasible probability allocations over the set of paths between x and t. The intruder, x, then picks a path $P \in \rho_x^t$ with probability $q_x(P)$ for each malicious packet. The strategy for the IDS is to choose the sampling rate s_e on link e such that $\sum_{e\in E} s_e \leq B_s$. We also introduce $U = \{p : \sum_{e\in E} f_e p_e \leq B_s\}$ to represent the set of detection probability vectors $p = (p_{e_1}, ..., p_{e_l})$ that satisfy the sampling budget constraint. The strategy for the IDS is to pick a set of detection probabilities at the links which belongs to the set U.

4.4 Game Formulation: Single Intruder with Multiple Packets

Having the intruder and the IDS each chosen their strategies, (i.e., their probability distributions, (1) q over the set of paths in ρ_a^t, and (2) p a set of detection probabilities at the links for the intruder and IDS respectively). The payoff for both the IDS and the intruder depends on the probability of the intrusion being detected as it goes from a to t. The probability of sampling an *a-fragment* traversing from node a to node t is the sum of probability of taking each path times the probability of sampling the *a-fragment* on that particular path over all possible routes from a to t. Denote α to be this probability then we have:

$$\alpha = \sum_{P \in \rho_a^t} q(P)[1 - \prod_{e \in P}(1 - p_e)] \tag{7}$$

Therefore, the probability of sampling exactly *m a-fragments* is,

$$\alpha^m \times (1 - \alpha)^{n-m}. \tag{8}$$

Notice that the IDS will detect the intrusion if at least *m a-fragments* are sampled. Hence, the IDS will detect the intrusion with probability,

$$\sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i} \tag{9}$$

Accordingly, the IDS will choose a strategy that maximizes the detection probability:

$$\max_{p \in U} \sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i}, \tag{10}$$

where,

$$U = \{p : \sum_{e \in E} f_e p_e \leq B_s\}.$$

On the other hand, the objective of the intruder is to choose a distribution q and number of fragments n that minimize this maximum value. In other words, the objective of the intruder is:

$$\min_{n \in \mathbb{N}, q \in V} \max_{p \in U} \sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i}. \tag{11}$$

Using a similar argument, the objective of the IDS becomes:

$$\max_{p \in U} \min_{n \in \mathbb{N}, q \in V} \sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i}. \tag{12}$$

This is a classical two person zero-sum game. According to minmax theorem [52], there exists an optimal solution to the intrusion detection game where the following noted min-max result holds,

$$\theta = \max_{p \in U} \min_{n \in \mathbb{N}, q \in V} \sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i} \tag{13}$$

$$= \min_{n \in \mathbb{N}, q \in V} \max_{p \in U} \sum_{i=m}^{n} \alpha^i \times (1 - \alpha)^{n-i},$$

and θ is the value of the game.

4.5 Solution of the game: Single Intruder with Multiple Packets

We first consider the case where the distributed IDS needs all the *a-fragments* to detect the intrusion, $m = n$. Later, we investigate the game for a more general case, where at least half of the *a-fragments* are needed to detect the intrusion. Replacing m with n and recalling Equation 8, the problem will reduce to the following:

$$\max_{p\in U} \min_{n\in\mathbb{N},q\in V} \sum_{P\in\rho'_a} q(P)[1-\prod_{e\in P}(1-p_e)]^n \tag{14}$$

$$= \min_{n\in\mathbb{N},q\in V} \max_{p\in U} \sum_{P\in\rho'_a} q(P)[1-\prod_{e\in P}(1-p_e)]^n,$$

Before solving the game, we first prove the following lemma:

Lemma 1 *The following inequality holds in our game,*

$$\prod_{e\in E} f_e(1-p_e) \geq \sum_{e\in E} f_e(1-p_e).$$

Proof. Using the assumption in Section 3 (1) we have,

$$f_e > s_e + 1. \tag{15}$$

Given that $p_e = \frac{s_e}{f_e}$, we replace s_e by $p_e f_e$, which will give us the following:

$$f_e > p_e f_e + 1 \tag{16}$$

rephrasing (16), we will have,

$$f_e(1-p_e) > 1 \tag{17}$$

Notice that the sum of any number of variables each greater than one is less than or equal to the product of them. Knowing that for all the links $f_e(1-p_e) > 1$, the following holds,

$$\prod_{e\in E} f_e(1-p_e) \geq \sum_{e\in E} f_e(1-p_e). \tag{18}$$

■

Consider the intruders problem:

$$\min_{n\in\mathbb{N},q\in V} \max_{p\in U} [\sum_{P\in\rho_a^t} q(P)[1-\prod_{e\in P}(1-p_e)]]^n. \tag{19}$$

For a fixed $q \in V$ and n the inner maximization problem can be written as,

$$\max_{p\in U} [\sum_{P\in\rho_a^t} q(P)[1-\prod_{e\in P}(1-p_e)]]^n. \tag{20}$$

For a fixed n to maximize the expression above we have to maximize the following,

$$\max_{p\in U} [\sum_{P\in\rho_a^t} q(P)[1-\prod_{e\in P}(1-p_e)]], \tag{21}$$

which is equal to

$$\max_{p\in U} [\sum_{P\in\rho_a^t} q(P) - \sum_{P\in\rho_a^t} [q(P)\prod_{e\in P}(1-p_e)]], \tag{22}$$

or alternatively one can minimize the following,

$$\min_{p\in U} \sum_{P\in\rho_a^t} [q(P)\prod_{e\in P}(1-p_e)]. \tag{23}$$

Note that for any positive function, $f(x)$, in order to minimize that function, it is sufficient to minimize $\ln f(x)$. This follows from the strictly increasing characteristic of the logarithmic function. Since we are solving for a fixed q, we can now minimize the following,

$$\min_{p\in U} \sum_{P\in\rho_a^t} [q(P)\ln\prod_{e\in P}(1-p_e)], \tag{24}$$

which is equal to,

$$\min_{p\in U} \sum_{P\in\rho_a^t} [q(P)\sum_{e\in P}\ln(1-p_e)]. \tag{25}$$

It is evident that $ln(1-p_e) \leq 0$, since $0 \leq p_e \leq 1$. We introduce p'_e to be $-\ln(1-p_e)$ and rewrite the minimization problem in terms of p'_e.

$$\min_{p \in U} \sum_{P \in \rho'_a} [q(P) \sum_{e \in P} -p'_e] \tag{26}$$

Or alternatively,

$$\max_{p \in U} \sum_{P \in \rho'_a} [q(P) \sum_{e \in P} p'_e]. \tag{27}$$

Similarly, we have to rewrite the constraints in terms of the new variable, p'_e. The sampling constraint is,

$$\sum_{e \in E} f_e p_e \leq B_S \tag{28}$$

Replacing p_e with $1 - e^{-p'_e}$, we get,

$$\sum_{e \in E} f_e (1 - e^{-p'_e}) \leq B_s, \tag{29}$$

which is,

$$\sum_{e \in E} f_e - B_s \leq \sum_{e \in E} f_e e^{-p'_e}. \tag{30}$$

Using Lemma 1, and substituting p_e for p'_e we have,

$$\prod_{e \in E} f_e e^{-p'_e} \geq \sum_{e \in E} f_e e^{-p'_e}. \tag{31}$$

Therefore, applying log to both sides and using equation (30), we get,

$$\sum_{e \in E} \ln(f_e e^{-p'_e}) \geq \ln(\sum_{e \in E} f_e - B_s). \tag{32}$$

Or,

$$\sum_{e \in E} \ln f_e - \sum_{e \in E} p'_e \geq \ln(\sum_{e \in E} f_e - B_s). \tag{33}$$

Therefore, the game simplifies to the following:

$$\max_{p \in U} \sum_{e \in E} [p'_e \sum_{P \in \rho'_a, e \in P} q(P)], \tag{34}$$

subject to the constraints,

$$\sum_{e\in E} p'_e \leq \sum_{e\in E} \ln f_e - \ln(\sum_{e\in E} f_e - B_s),$$

$$p'_e \geq 0.$$

Associating a dual variable λ [44], we obtain the following dual optimization problem with the corresponding constraints:

$$\min(\sum_{e\in E} \ln f_e - \ln(\sum_{e\in E} f_e - B_s))\lambda, \tag{35}$$

$$\forall e \in E, \lambda \geq \sum_{P\in\rho'_a, e\in P} q(P),$$

$$\lambda \geq 0,$$

$$\sum_{P\in\rho'_a} q(P) = 1.$$

Interpreting $q(P)$ as a flow on path P, the constraint

$$\sum_{P\in\rho'_a, e\in P} q(P) \leq \lambda,$$

restricts the flow for all links to be at most λ. Hence, λ can be interpreted as the maximum capacity of all links in ρ'_a. The constraint $\sum_{P\in\rho'_a} q(P) = 1$ enforces one unit flow to be sent from node a to node t.

The objective of the game is therefore to determine the smallest λ so that a flow of one unit can be sent from node a to node t. This can be done as follows:

1. Assume that link e has capacity f_e and determine the maximum flow from a to t, $MF'_a(f)$, using these capacities.

2. Assume that link e has capacity f_e and determine the maximum flow among all links in ρ'_a, $max'_a(f)$, using these capacities.

3. Scale the capacities by $MF'_a(f)^{-1}$ so that a flow of one unit can be sent from node a to node t.

4. λ will be $MF'_a(f)^{-1} max'_a(f)$.

5. The value of the game is $\theta = (\sum_{e\in E} \ln f_e - \ln(\sum_{e\in E} f_e - B_s))MF'_a(f)^{-1} max'_a(f)$.

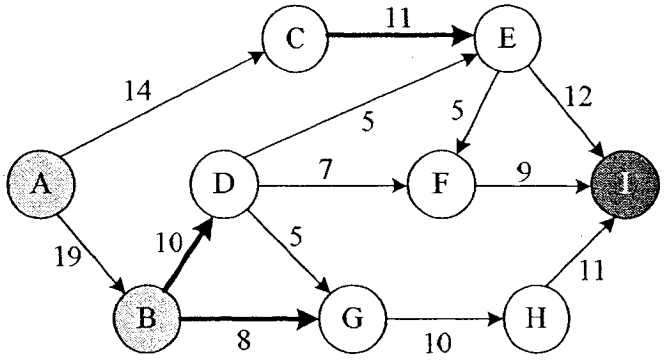

Figure 4.2: Single intruder with multiple *a-fragments*

From the network flow duality [19], corresponding to the maximum flow value there is a minimum cut. The IDS computes the maximum flow from a to t using f_e as the capacity of the link e. Let $e_1, e_2, ..., e_r$ denote the arcs in the corresponding minimum cut with flows $f_1, f_2, ..., f_r$. From duality [19] $\sum_{i=1}^{r} f_i = MF_a^t(f)$. The IDS samples link e_i at rate $B_s f_e MF_a^t(f)^{-1}$. The intruder, on the other hand, has a fragmentation budget constraint, B_i, depending on the difficulty of launching the attack over multiple packets. He then chooses the number of fragments, n, to be equal to B_i and uses the standard flow decomposition techniques to decompose the maximum flow into flow on paths $P_1, P_2, ..., P_l$ from node a to node t with flows of $m_1, m_2, ..., m_l$ respectively (note that $\sum_{i=1}^{l} m_i = MF_a^t(f)$). The intruder transmits each malicious fragment packet along the path P_i with probability $m_i MF_a^t(f)^{-1}$.

We now illustrate the results in Section 5 on the example shown in Figure 4.2. The numbers next to the links are the flows on the links. Suppose that there is a sampling budget B_s of 12 units for the IDS. Assume, also that the intruder's fragmentation budget constraint, B_i, is equal to 3. The nodes, $a = A$ and $t = I$ are the attack and target nodes respectively. The links (C,E), (B,D) and (B,G) belong to the minimum $a-t$ [49] cut which are shown in thick lines. The minimum cut (and hence the maximum flow) has a value of 29 units. The maximum flow on the links in ρ_a^t, $max_a^t(f)$, is 18.

The intruder will launch the attack over 3 fragments and the min-max strategy is the following:.

For each fragment,

1. Transmit the malicious fragment along the path $A-C-E-I$ with probability $11/29$.

2. Transmit the malicious fragment along the path $A-B-G-H-I$ with probability

8/29.

3. Transmit the malicious fragment along the path $A-B-D-F-I$ with probability 7/29.

4. Transmit the malicious fragment along the path $A-B-D-G-H-I$ with probability 2/29 .

5. Transmit the malicious fragment along the path $A-B-D-E-F-I$ with probability 1/29.

Correspondingly, the IDS's strategy is the following:

1. Sample link (C,E) with the sampling rate $s_e = (12\times 11)/29$.

2. Sample link (B,G) with the sampling rate $s_e = (12\times 8)/29$.

3. Sample link (B,D) with the sampling rate $s_e = (12\times 10)/29$.

Note that the total sampling budget is equal to 12.

4.6 Analyzing the game: A more practical case

We now analyze the game for a practical case, where the intrusion detection system needs at least half of the *a-fragments* to detect the intrusion. Using Equation 4 from Section 4, we introduce Γ to be this probability for $n = 2m$.

$$\Gamma = \sum_{i=m}^{n} \alpha^i \times (1-\alpha)^{n-i} \tag{36}$$

First, we show that Γ is concave, therefore having a local maximum on $\alpha \in [0,1]$. Then we aim at maximizing the function Γ. In order to prove that the function is concave, it is sufficient to show that the second derivative of the function with respect to p_e is always negative.

We know that Γ is the following:

$$\Gamma = \sum_{i=m}^{2m} \alpha^i \times (1-\alpha)^{2m-i} \tag{37}$$

Therefore, the first derivative of the function is as follows:

$$\frac{\delta}{\delta p_e}\Gamma = \sum_{i=m}^{2m} i\alpha^{i-1} \times (1-\alpha)^{2m-i} \frac{\delta}{\delta p_e}\alpha \tag{38}$$

$$-\sum_{i=m}^{2m} (2m-i)\alpha^{i} \times (1-\alpha)^{2m-i-1} \frac{\delta}{\delta p_e}\alpha$$

Factorizing $\frac{\delta}{\delta p_e}\alpha$ and $\alpha^{i-1} \times (1-\alpha)^{2m-i-1}$, Equation (38) simplifies to the following:

$$\frac{\delta}{\delta p_e}\Gamma = \frac{\delta}{\delta p_e}\alpha \sum_{i=m}^{2m} (i-n\alpha) \times \alpha^{i-1} \times (1-\alpha)^{2m-i-1} \tag{39}$$

where,

$$\frac{\delta}{\delta p_e}\alpha = \sum_{P \in \rho'_a} q(P) \prod_{e' \in P, e' \neq e} (1-p_{e'}) \tag{40}$$

and e' is any edge on path P except for e. Therefore, the second derivative of α with respect to p_e is zero, $\frac{\delta^2}{\delta p_e}\alpha = 0$. Thus, the second derivative of Γ reduces to the following:

$$\frac{\delta^2}{\delta p_e}\Gamma = \frac{\delta}{\delta p_e}\alpha \times \frac{\delta}{\delta p_e} \sum_{i=m}^{2m} (i-n\alpha) \times \alpha^{i-1} \times (1-\alpha)^{2m-i-1} \tag{41}$$

Applying δ on the summation, we will have:

$$\frac{\delta^2}{\delta p_e}\Gamma = (\frac{\delta}{\delta p_e}\alpha)^2 \times \sum_{i=m}^{2m} \{-n \times \alpha^{i-1} \times (1-\alpha)^{2m-i-1} \tag{42}$$

$$+(i-n\alpha) \times (i-1) \times \alpha^{i-2} \times (1-\alpha)^{2m-i-1}$$

$$-(i-n\alpha) \times \alpha^{i-1} \times (2m-i-1) \times (1-\alpha)^{2m-i-2}\}$$

By factorizing $\alpha^{i-2} \times (1-\alpha)^{2m-i-2}$, Equation (42) simplifies to the following:

$$\frac{\delta^2}{\delta p_e}\Gamma = (\frac{\delta}{\delta p_e}\alpha)^2 \sum_{i=m}^{2m} \{\alpha^{i-2} \times (1-\alpha)^{2m-i-2} \times \tag{43}$$

$$[(i-n\alpha)^2 - n\alpha^2 + 2\alpha - i]\}$$

For Γ to be concave, $\frac{\delta^2}{\delta p_e}\Gamma$ has to be negative. Since $(\frac{\delta}{\delta p_e}\alpha)^2$ is always positive, it is

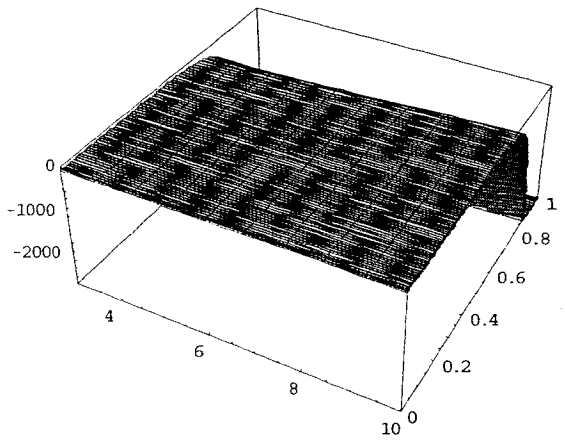

Figure 4.3: The graph for χ

sufficient to show the function χ,

$$\chi = \sum_{i=m}^{2m} \alpha^{i-2} \times (1-\alpha)^{2m-i-2} \times [(i-n\alpha)^2 - n\alpha^2 + 2\alpha - i] \tag{44}$$

is negative everywhere in the interval $\alpha \in [0,1]$. In order to prove that the function χ is negative, we have graphicly illustrated it in Figure 4.3. Proving that the function Γ has a local maximum, we can solve the game. Now, we set the first derivative to zero in order to maximize the probability of detection. Therefore, the IDS assigns a sampling probability, p_e, to each link, e, such that the following holds:

$$\forall e \in \rho_a^t, \sum_{P \in \rho_a^t} q(P) \prod_{e' \in P, e' \neq e} (1 - p_{e'}) = 0 \tag{45}$$

since $p_{e'}$ $0 \leq p_{e'} \leq 1$, therefore, $(1-p_{e'})$ is always positive, i.e., $\prod_{e' \in P, e' \neq e}(1-p_{e'}) \geq 0$. Thus, we can see that Equation (45) is the summation of a number of positive terms. To set this to zero, all the terms should be equal to zero. So, we should sample all the packets on one link per path, for all the paths in ρ_a^t. To minimize the sampling, we take the set of links to be sampled to be the minimum cut between nodes a and t. Note that according to each minimum cut, there exists a maximum flow with the same value. However, we know that the sampling budget B_s might be less than the maximum flow from a to t, $MF_a^t(f)$. In this case we distribute the sampling budget among the links in minimum cut, according to their flows. Thus, the strategy of the IDS would be to sample each link, e, in the minimum cut

with sampling rate, $s_e = B_s \frac{f_e}{MF_a^t(f)}$.

4.7 Game Formulation: Cooperative Intruders

Here, we have a distributed agent based IDS. The IDS sample the packets on each link via the agents while not exceeding the sampling budget, B_s. The game is played on the infrastructure network between the IDS and the cooperating intruders. Assuming the set of cooperative intruders as one player, we model the game as a zero-sum game: the IDS and the intruders. The objective of each intruder $x \in \Omega$ is to send a malicious packet to the target node t. An intrusion is successful when at least m malicious packets out of the $|\Omega|$ packets reach the desired target node t without detection. In order to detect the intrusion, the IDS samples packets in the network via its agents. Furthermore, the agents sample the traffic on each link in the network as presented in section 4.3.

The intruders and the intrusion detection system (IDS) each should choose their strategies, which are the probability distributions: q_x over the set of paths in ρ_x^t, and p a set of detection probabilities at the links for the intruders and the IDS respectively. Then, the payoff for both the IDS and the intruders depends on the probability of the intrusion being detected as it goes from the intruding nodes to the target node t. For any node $x \in \Omega$, the probability of sampling a packet traversing from node x to node t is the sum of probability of taking each path times the probability of sampling the packet on that particular path over all possible routes from x to t. We introduce α_x to be the probability of detecting the intrusion, when intruder x is attacking node t, which is given by:

$$\alpha_x = \sum_{P \in \rho_x^t} q(P)[1 - \prod_{e \in P}(1 - p_e)] \tag{46}$$

Next, we define the function Φ to be the mean value of detecting the intrusion through sampling:

$$\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x \tag{47}$$

The main goal of the IDS is to maximize Φ. In other words, the IDS aims at maximizing the following:

$$\max_{p_e \in U} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{48}$$

where:

$$U = \{p : \sum_{e \in E} f_e p_e \leq B_s\}$$

On the other hand, the cooperative intruders aim at minimizing Equation (48). The intruders will fulfil this objective by assigning probabilities for all possible routes to the target node:

$$\min_{q \in V_x} \max_{p \in U} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{49}$$

where:

$$V_x = \{q : \sum_{P \in \rho_x'} q(P) = 1\}$$

Using a similar argument, the objective of the IDS becomes:

$$\max_{p \in U} \min_{q \in V_x} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{50}$$

This is a mixed strategy zero-sum game, for which the following min-max theorem holds:

$$\beta = \max_{p \in U} \min_{q \in V_x} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x)$$

$$= \min_{q \in V_x} \max_{p \in U} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{51}$$

where β is the value of the game.

4.8 Solution of the game: Cooperative Intruders

In this section, we propose our solution to the min-max problem formulated in section 4.7. First, we consider the intruders' problem:

$$\min_{q \in V_x} \max_{p \in U} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{52}$$

For a fixed q the problem reduces to the following:

$$\max_{p \in U} (\Phi = \frac{1}{|\Omega|} \sum_{x \in \Omega} \alpha_x) \tag{53}$$

In order to maximize the previous equation, it is sufficient to maximize all the terms. Therefore, the problem simplifies to the following:

$$\max_{p \in U} \alpha_x \tag{54}$$

We know that each node is sending one packet to the target node. Therefore, we can divide the budget, B_s among the $|\Omega|$ intruders. Thus, each intruder, x, will have the budget constraint $\frac{B_s}{|\Omega|}$ or $B_s|\Omega|^{-1}$. Replacing α_x, using Equation (46) and rewriting Equation (54) the problem can be written as follows:

$$\max_{\pi^x \in U_x} \sum_{P \in \rho'_x} q(P)[1 - \prod_{e \in P}(1 - \pi_e^x)] \tag{55}$$

where,

$$\sum_{x \in \Omega} \pi_e^x = p_e,$$

$$U_x = \{\pi^x : \sum_{e \in E} f_e \pi_e^x \leq B_s|\Omega|^{-1}\}$$

Having the sampling constraint:

$$\sum_{e \in E} f_e \pi_e \leq B_s|\Omega|^{-1} \tag{56}$$

Thus, using the same approach as in Section 5, the subgame reduces to the following:

$$\min(\sum_{e \in E} \ln f_e - \ln(\sum_{e \in E} f_e - B_s|\Omega|^{-1}))\lambda \tag{57}$$

Subject to:

$$\lambda \geq \sum_{P \in \rho'_x, e \in P} q(P), \forall x \in \omega, \forall e \in E \tag{58}$$

$$\lambda \geq 0 \tag{59}$$

$$\sum_{P \in \rho'_x} q(P) = 1 \tag{60}$$

Next, we calculate the maximum flow from x to t, $MF_x^t(f)$. Knowing that the maximum flow is equal to the summation of the flows on all the paths from x to t, we then normalize the flows in the network (with respect to the $MF_t^x(f)$). Therefore, the normalized flow on each path can be interpreted as $q(P)$ and constraint (58) holds. Furthermore, interpreting

$q(P)$ as the flow on path P suggests $\sum_{P\in\rho_x^t, e\in P} q(P)$ to be the normalized flow on link e. Hence, in order to minimize λ, satisfying constraint (58), we introduce λ to be the maximum flow among all the flows in ρ_x^t; that is, $\lambda = max_x^t(f)$. Therefore, the value of the game is:

$$[\sum_{e\in E} \ln f_e - \ln(\sum_{e\in E} f_e - B_s|\Omega|^{-1})]max_x^t(f) \tag{61}$$

The game would guide us to the following strategies satisfying the budget constraint. The intruder x's strategy is:

1. Calculate the maximum flow from x to t using f_e as the capacity of the link e.
2. Use the standard flow decomposition techniques [4] to decompose the maximum flow into flows on paths $P_1, P_2, ..., P_{l_x}$ from node x to node t with flows of $m_1, m_2, ..., m_{l_x}$ respectively, knowing that $\sum_{i=1}^{l_x} m_i = MF_x^t(f)$ and $|\rho_x^t| = l_x$.
3. Transmit the malicious packet along the path P_i with probability $m_i MF_x^t(f)^{-1}$.

Consequently, the IDS's strategy is:

1. For each node $x \in \Omega$ find the minimum cut.
2. Let $Mincut_x^t$ denote the set of arcs in the corresponding minimum cut.
3. Sample link e at rate:

$$\sum_{x\in\Omega, e\in Mincut_x^t} B_s|\Omega|^{-1} MF_x^t(f)^{-1} f_e \tag{62}$$

Note that, $\sum_{e\in E}\sum_{x\in\Omega, e\in Mincut_x^t} B_s|\Omega|^{-1} MF_x^t(f)^{-1} f_e = B_s$ and therefore, satisfying the budget constraint.

Now, we illustrate the game with an example as shown in Figure 4.4, where nodes A and E are the cooperative intruders and node I is the target. In other words, $\Omega = \{A, E\}$ and $|\Omega| = 2$. The budget constraint, B_s, is 60. Therefore, $B_s|\Omega|^{-1} = 30$. The maximum flow from A to I is 99 and the maximum flow from E to I is 54 as shown in Figure 4.4 in bold links. $Mincut_A^I = \{AB, DG, EG, EF\}$ and $Mincut_E^I = \{EG, EF\}$. Hence the IDS will sample the links as follows:

1. AB with sampling rate $30 * 30/99 \simeq 9.09$
2. DG with sampling rate $30 * 15/99 \simeq 4.54$

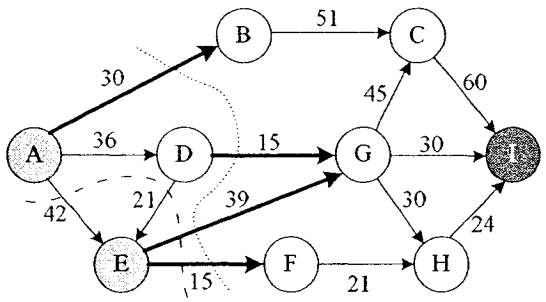

Figure 4.4: Cooperative multi-intruder attack

3. EG with sampling rate $30*39/99+30*39/54 \simeq 33.47$

4. EF with sampling rate $30*15/99+30*15/54 \simeq 12.87$

Note that the total sampling is $59.97 \leq 60$. The intruders send the malicious packet as follows:

1. Node A sends the malicious packet through path $ABCI$ with probability 30/99.

2. Node A sends the malicious packet through path $ADGI$ with probability 15/99.

3. Node A sends the malicious packet through path $ADEGCI$ with probability 21/99.

4. Node A sends the malicious packet through path $AEGCI$ with probability 18/99.

5. Node A sends the malicious packet through path $AEFHI$ with probability 15/99.

6. Node E sends the malicious packet through path $EGCI$ with probability 39/54.

7. Node E sends the malicious packet through path $EFHI$ with probability 15/54.

4.9 Experimental Results

Testing and evaluation will be achieved by comparing our game theoretic framework results with two different approaches: random and uniform models. Random model is a model where sampling is done on random links. While uniform model is achieved through dividing the sampling effort equally over all the links. Note that, all the models must satisfy

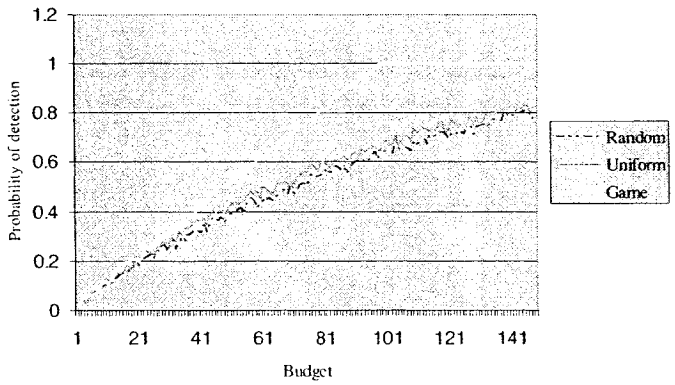

Figure 4.5: One intruder sending *a-fragments*

the sampling budget constraint. Moreover, simulation among all the models will be done taking into consideration the same graph as shown in Figure 4.4.

First, we consider the scenario where a single intruder transmits the *a-fragments* to a target node in order to launch the attack. Detection is fulfilled if half of the *a-fragments* are sampled. Here, we assume that node A is the attacker and node I is the target. Figure 4.5 shows the detection probability as a function of the budget, where the budget varies from 1 to 150 (packets/second). From the case study in Section (8), we know that the maximum flow between these two nodes is 99. As it is shown in Figure 4.5, as soon as the budget reaches the maximum flow, the probability of detection becomes 1, no matter how many packets are being sent. This is because we are not sampling randomly or uniformly on all the edges. Instead, we focus all the budget on the minimum cut edges, where every packet transmitted from the attacker to the target has to traverse at least one of the links in the minimum cut set. From the minimax theorem [49], we know that the summation of flows in the minimum cut is equal to the maximum flow. Therefore, if the sampling budget is equal to or greater than the maximum flow between the attacker and the target, we can sample with a rate equal to the actual flow on each link in the minimum cut. Thus, any packet either normal or malicious would be sampled ensuring that the intrusion is being detected. We can see that the game results are much better than the other two approaches. For very small sampling budgets, all the three approaches have small detection probabilities. As the budget increases, we can see that the detection probability increases, for all approaches. The reason is that we sample with higher sampling rates on the links, and thus greater sampling probabilities on each link. Therefore, the total probability of detection increases. As shown in the figure, our game approach has a greater slope; this is due to the fact that

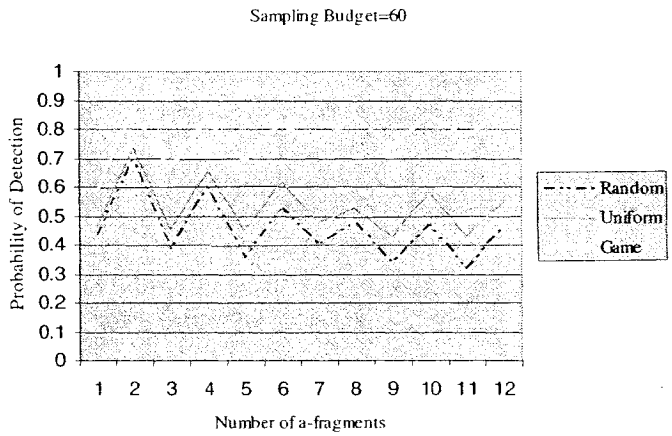

Figure 4.6: One intruder sending *a-fragments*

the sampling is done on the critical links where any traffic has to be transmitted through them (i.e., minimum cut). In other words, the budget is distributed over a set of critical links instead of all the links in the network, while all the traffic is still traversing through these links. This results in better detection rate.

Figure 4.6, illustrates the results of another scenario, where an intruder A transmits different numbers of *a-fragments* to a target node I having a constant sampling budget equal to 60. The attacker transmits the a-fragments through different paths. Note that there are 12 paths from A to I that could be selected randomly by the intruder. Here, the detection probability is demonstrated as a function of the number of *a-fragments*. As we can see the detection probability for odd number of *a-fragments* is less than the even ones for two consecutive numbers of *a-fragments*. This is due to the reason that the IDS needs at least half of the *a-fragments* which is *one* more for the case of odd numbers. In case of larger networks, this difference between odd and even number of packets would be neglected. The results are illustrated in Figure 4.6. Using the same terminology as in the previous scenario, our game theoretic framework presents better results than the other two models.

Finally, we illustrate the multi-intruder scenario, where m cooperating intruders distribute the attack over m *a-fragments*, and where each intruder sends one *a-fragment* to a common target node. The attack is successful if at least half of these *a-fragments* reach the target node without being detected. The simulation results are shown in Figure 4.7. The detection probability decreases as the number of intruders increases. This is because the IDS has to divide the budget over the number of intruders. When the number of intruders is less than 60% of the total number of nodes in the network, focusing the sampling budget

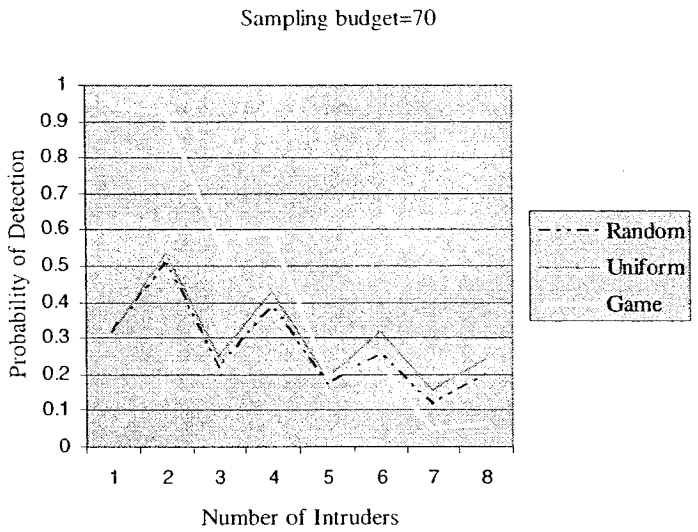

Figure 4.7: Multi-intruders sending one malicious packets each

on the union of the minimum cuts for each intruder and the target node, helps in increasing the detection probability. In this case, the number of links in the union of minimum cuts is still much less than the total number of links in the network. Therefore, we distribute the total sampling budget over a smaller number of links and consequently the sampling rate increases on each link leading to better results. As the number of intruders increases, more and more links are added to the union of critical edges (union of minimum cut sets for each intruder and the target node). Thus, the set of the links becomes comparable to the total number of links. Here, the *a-fragments* are almost over all the links. In this case, the sampling budget is divided by the number of attackers, which becomes a relatively small number. The sampling rate on the other hand would be multiplied by this small sampling budget and divided by the maximum flow. Thus, the sampling probability decreases. For random and uniform strategy, the budget is independent of the number of attackers. In other words, they continue to sample almost with the same rate for any number of attackers. This shows why the uniform and random methods provide better results than the game approach in the case where the number of intruders is greater than 60% of the total number of nodes in the network.

4.10 Conclusions

We considered the problem of intrusion over multiple packets in a network by means of network packet sampling. Given a total sampling budget, we developed a network packet sampling strategy to effectively reduce the success chances of an intruder. We considered two different scenarios where the adversary(s) has(have) considerable information about the network and can pick paths to minimize chances of detection. In case of a single intruder, we formulated the intrusion detection problem as a zero-sum two-player game between the IDS and the attacker. First, we considered the case where the IDS needs all the malicious fragments to detect the intrusion, and we have done a case study to gain more insights about the solution of the game. Then, we analyzed the case where the IDS needs at least half of the *a-fragment*s to detect the intrusion. Furthermore, we considered the problem of multiple cooperating intruders. We considered the scenario where the attackers pick paths to minimize chances of detection. We formulated the intrusion detection problem as a zero-sum non-cooperative game between the IDS and the set of attackers. Moreover, we solved the game to bring up strategies for both the IDS and the set of intruders. Finally, we evaluated our game solutions via simulation.

Chapter 5

Conclusion and Future Work

In this study, we showed that intrusion detection systems are needed as a second line of defense, since intrusion prevention tools can not be enough. The intrusion detection systems monitor the system for any sign of intrusive action and raise an alarm as soon as they find a misuse pattern or anomaly. The system administrator then responds to that intrusion. Furthermore, we explained that there is a need for a quantitative decision and control framework in order to address issues like attack modeling, analysis of detected threats, and decision on response actions. We showed that game theory were a strong candidate to provide the much-needed mathematical framework for analysis, modeling, decision, and control process for information security and intrusion detection. Given the continuous struggle between attackers, who aim to penetrate the deployed systems and the IDS trying to protect these systems, we modeled these interactions as a non-cooperative game, where the players were the intruders and the intrusion detection system. In order to provide the much needed background, we studies intrusion detection systems and game theory in chapters two and three respectfully.

We presented a complete taxonomy of intrusion detection systems and we discussed different techniques for intrusion detection. We also, discussed the state of the art of intrusion detection systems and the different approaches such as data mining techniques, artificial intelligence, agent-based techniques, software engineering methods and the embedded system approaches. Besides, we explained all the different approaches researchers use for intrusion detection.

Additionally, we presented a survey of different kinds of games. We presented taxonomy of games with concrete examples of each in order to illustrate each kind of game. We talked about zero sum games and Nash equilibrium in detail. Furthermore, we explained

mixed strategies.

After providing the reader with necessary background about intrusion detection and game theory, we proposed our game theoretic model for intrusion detection. We considered two scenarios, where in the first one a well informed intruder distributes her attack over multiple packets to evade the intrusion detection system and in the second one a group of cooperative attackers distribute the attack among themselves. We formulated both problems and using game theory we presented optimal sampling strategies to maximize the probability of detection in each scenario. Finally we simulated our contributions and compared them with two other sampling methods: random sampling and uniform sampling. Our game theoretic framework showed better results in almost all cases.

In future, we are planning to extend our study for the case, where the intruder(s) and the target can be any node in the network. We would formulate the game and provide optimal sampling strategies. This could be more feasible by introducing a set of nodes to be possible intruders via assigning reputation to each node depending on its previous behavior. Furthermore, we would consider the problem of cooperation enforcement. We believe that we can enforce cooperation by using game theory and mechanism design.

Bibliography

[1] `http://www.cert.org`. Carnegie Mellon University's Computer Emergency Response Team.

[2] Ajith Abraham, Kate Smith, Ravi Jain, and Lakhmi Jain. Network and information security: A computational intelligence approach. *Journal of Network and Computer Applications*, 2005.

[3] Afrand Agah, Sajal K. Das, Kalyan Basu, and Mehran Asadi. Intrusion detection in sensor networks: A non-cooperative game approach. In *IEEE 3rd International Symposium on Network Computing and Applications (NCA'04)*, pages 343–346, August 2004.

[4] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows : Theory, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[5] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38:393–422, 2002.

[6] Tansu Alpcan and Tamer Basar. A game theoretic approach to decision and analysis in network intrusion detection. In *IEEE 42nd Conference on Decision and Control (CDC)*, December 2003.

[7] Tansu Alpcan and Tamer Basar. A game theoretic analysis of intrusion detection in access control systems. In *IEEE 43rd Conference on Decision and Control (CDC)*, volume 2, pages 1568–1573, December 2004.

[8] J. S. Arora. *Introduction to Optimum Design*. McGraw-Hill, 1989.

[9] Stefan Axelsson. Intrusion detection systems: A survey and taxonomy. Technical Report 99-15, Chalmers University, march 2000.

[10] Rebecca Bace. An introduction to intrusion detection and assessment for system and network security management. April 1999.

[11] Rebecca Gurley Bace. *Intrusion Detection*. Sams, 1999.

[12] Rebecca Gurley Bace and Peter Mell. *Intrusion Detection Systems*. NIST Computer Security Division, 2001.

[13] E. Badouel and P. Darondeau. Theory of regions. In *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*, pages 529–586. Springer, 1999.

[14] Yacine Bouzida and Sylvain Gombault. Intrusion detection using principal component analysis. In *7th World Multiconference on Systemics, Cybernetics and Informatics (WMSCI'03)*, Orlando, Florida, July 2003.

[15] G. M. Campbell and C. F. Dolan. Using game theory to introduce ethics in decision sciences. *The Decision Sciences Journal of Innovative Education*, 2:229, July 2004.

[16] James Cannady. Artificial neural networks for misuse detection. In *21st National Information Systems Security Conference (NISSC'98)*, pages 441–454, Arlington, Virginia, 1998.

[17] Vincent Conitzer and Tuomas Sandholm.

[18] Kevin R. Coombes, J. M. Rosenberg, and Ronald L. Lipsman. *Multivariable Calculus and Mathematica*. Springer, may 1998.

[19] Thomas Epping, Winfried Hochstättler, and Marco E. Lübbecke. MaxFlow-MinCut duality for a paint shop problem. In U. Leopold-Wildburger, F. Rendl, and G. Wäscher, editors, *Operations Research Proceedings 2002*, pages 353–358, Berlin, 2003. Springer.

[20] Jun feng Tian, Yue Fu, Ying Xu, and Jian ling Wang. Intrusion detection combining multiple decision trees by fuzzy logic. In *6th International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*, pages 256–258, 2005.

[21] A. Gelman, J.B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, London, 1995.

[22] Vaibhav Gowadia, Csilla Farkas, and Marco Valtorta. Paid: A probabilistic agent-based intrusion detection system. *Computers and Security*, 24(7):529–545, October 2005.

[23] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[24] Hewlett-Packard Development Company, LP. *HP-UX Host Intrusion Detection System Administrator's Guide: Software Release 2.2*, 3 edition, 2003.

[25] Paul Innella and Oba McMillan. An introduction to intrusion detection. December 2001.

[26] B. Wyckoff Jb Sidowski and L. Tabory. The influence of reinforcement and punishment in a minimal social situation. *J Abnorm Psychol*, 52(1):115–119, January 1956.

[27] A. Kaufmann and M. Gupta. *Introduction to Fuzzy Arithmetic: Theory and Applicatioons*. International Thomson Computer Press, London, 1991.

[28] Przemyslaw Kazienko and Piotr Dorosz. Intrusion detection systems. part ii. classification, methods, and techniques. *IT FAQ*, 4:17–22, 2003.

[29] H.H. Kelley, J.W. Thibaut, R. Radloff, and D. Mundy. The development of cooperation in the minimal social situation. *Psychological Monographs*, 76(19), 1962.

[30] Anthony Kelly. *Decision Making using Game Theory : An Introduction for Managers*. Cambridge University Press, March 2003.

[31] Joseph M. Kizza. *Computer Network Security*. Springer-Verlag, New York, 2005.

[32] Murari Kodialam and T. V. Lakshman. Detecting network intrusions via sampling: A game theoretic approach. In *22nd Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM)*, MARCH 2003.

[33] Micki Krause and Harold F. Tipton. *Handbook of Information Security Management*. Auerbach Publications, 1999.

[34] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Bayesian event classification for intrusion detection, 2003.

[35] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Bayesian event classification for intrusion detection. In *IEEE 19th Annual Computer Security Applications Conference (ACSAC '03)*, December 2003.

[36] Christopher Kruegel, Fredrik Valeur, Giovanni Vigna, and Richard Kemmerer. Stateful intrusion detection for high-speed networks. In *IEEE Symposium on Research on Security and Privacy (RSP'02)*, Oakland, CA, May 2002. IEEE Press.

[37] Aleksandar Lazarevic, Jaideep Srivastava, and Vipin Kumar. *Data Mining for Intrusion Detection.* Tutorial at the Pacific-Asia Conference on Knowledge Discovery in Databases, 2003.

[38] W. Lee, S. Stolfo, and K. Mok. Mining audit data to build intrusion detection models. In *4th International Conference on Knowledge Discovery and Data Mining*, 1998.

[39] Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. A data mining framework for building intrusion detection models. In *IEEE Symposium on Security and Privacy*, pages 120–132, 1999.

[40] Peng Liu, Wanyu Zang, and Meng Yu. Incentive-based modeling and inference of attacker intent, objectives, and strategies. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):78–118, 2005.

[41] A. Ghoting G. Li S. Narravula M. Otey, S. Parthasarathy and D. Panda. Towards nic-based intrusion detection. pages 723–728, 2003.

[42] Mona Mehrandish, Chadi Assi, and Mourad Debbabi. A game theoretic model to handle network intrusions over multiple packets. In *IEEE Internationl Conference on Communications (ICC'06)*, Istanbul, Turkey, June 2006.

[43] Peter Morris. *Introduction to Game Theory.* Springer, July 1994.

[44] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization.* PH, 1982.

[45] Robert A. Pollak. Additive von neumann-morgenstern utility functions. *The Econometric Society*, pages 485–494, 1967.

[46] Min Qin and Kai Hwang. Frequent episode rules for internet anomaly detection. In *IEEE 3rd International Symposium on Network Computing and Applications (NCA'04)*, pages 161–168, August 2004.

[47] Sanjay Rawat, Arun K. Pujari, and Ved Prakash Gulati. On the use of singular value decomposition for a fast intrusion detection system. *Electronic Notes in Theoretical Computer Science*, 142:215–228, 2006.

[48] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou. Specification-based anomaly detection: a new approach for detecting network intrusions. In *9th ACM Conference on Computer and Communications Security (CCS'02)*, November 2002.

[49] Mechthild Stoer and Frank Wagner. A simple min-cut algorithm. *Journal of the ACM*, 44(4):585–591, 1997.

[50] Bennett Todd. Distributed denial of service attacks. March 2002.

[51] Giovanni Vigna, Fredrik Valeur, and Richard A. Kemmerer. Designing and implementing a family of intrusion detection systems. In *9th European software engineering conference held jointly with 11th ACM SIGSOFT international symposium on Foundations of software engineering (ESEC/FSE 2003)*, pages 88 – 97, Helsinki, Finland, September 2003.

[52] Michel Willem. *Minimax Theorems (Progress in Nonlinear Differential Equations and Their Applications)*. Birkhauser, 1997.

[53] Nong Ye, Yebin Zhang, and Connie M. Borror. Robustness of the markov-chain model for cyber-attack detection. *IEEE Transactions on Reliability*, 53(1):116–123, 2004.

[54] Feng Zhao, Qing-Hua Li, and Yan-Bin Zhao. Real time approaches for time-series mining-based ids. In *3rd International Conference on Grid and Cooperative Computing (GCC'04)*, pages 879–882, 2004.