OUTBOUND NETWORK TRAFFIC MONITRING

By

AMAL ALI AM ALGEDIR

Bachelor of Science in Computer Engineering

Alfateh University

Tripoli, Libya

2002

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 2012

OUTBOUND NETWORK TRAFFIC MONITORING

Thesis Approved:

Dr. George Scheets

Thesis Adviser

Dr. Qi Cheng

Dr. Nazanin Rahnavard

Dr. Sheryl A. Tucker

Dean of the Graduate College

TABLE OF CONTENTS

Chapter Page

I. INTRODUCTION 1

1.1 Internet security 1
1.2 Attacks Overview and Classification 2
1.3 Motivation of This Research 5
1.4 Overview of this Dissertation 7

II. REVIEW OF LITERATURE 8

2.1 Encryption and Decryption –Cryptographic Algorithms. 8
2.1.1 Symmetric key Encryption Algorithm. 8
2.1.2 Public key Algorithm. 11
2.2 Data Authentication and Hash Function. 12
2.3 Secure Socket Layer 14
2.4 Internet Protocol (IPv4 and IPV6 security) 14
2.5 Firewalls. 15
2.6 Intrusion Detection Systems. 18
2.6.1 Host Based Intrusion Detection 19
2.6.2 Network Based Intrusion Detection 19
2.6.3 Signature Based Intrusion Detection 20
2.6.4 Anomaly Based Intrusion Detection 20

III. METHODOLOGY 23

3.1 Overview of the Work 23
3.2 The Flow Charts of the Code 24
3.2.1 The Flow Chart of the Main Program 24
3.2.2 The Flow Chart of Remove Duplicated Class 26
3.2.3 The Flow Chart of Sorting Class 28
3.2.4 The Flow Chart of Query Website Class 26
3.3 The Program Graphic Interface 34

Chapter Page

IV. FINDINGS....36

Results Analysis....36

V. CONCLUSION....45

REFERENCES....47

APPENDICES....51

APPENDIX A- JAVA CODE OF THE FLOW CHARTS IN CHAPTER III……...51

APPENDIX A1.JAVA Code of Main Class……51

APPENDIX A2.JAVA Code of Remove Duplication IP Class……...……54

APPENDIX A3.JAVA Code of Sorting IP Class……55

APPENDIX A4.JAVA Code of Query Website Class ……60

APPENDIX A5.JAVA Code of the Detailed Information Class…….……69

LIST OF TABLES

Table Page

IV-1. The top ten countries; 9:00 am capture 37
IV-2. The top ten countries; 3:00 pm capture 38
IV-3. The top ten countries; 9:00 pm capture 40
IV-4. The top ten countries; 3:00 am capture 42
IV-5. The percentage of IP addresses of common websites 44

LIST OF FIGURES

Figure Page

II-1. Time Line of Firewall Architecture 16
III-1. The main program's flow chart 26
III-2. The Remove duplicate class's flow chart 27
III-3. The sort class's flow chart 29
III-4. The query website class's flow chart (part A, B) 32
III-5. Graphic user interface 35
IV-1. The percentage of IP destinations by continent, 9:00 am capture 38
IV-2. The percentage of IP destinations by continent, 3:00 pm capture 39
IV-3. The percentage of IP destinations by continent, 9:00 pm capture 41
IV-4. The percentage of IP destinations by continent, 3:00 am capture 42
IV-5. The number of the location in each country during different time 43

CHAPTER I

INTRODUCTION

1.1 Internet Security

Interconnected computer systems allow for improved information exchange between the users within the same or different organizations, allowing companies to achieve greater efficiencies.

With roots in ARPANET whose development began in the late 1960's [1], the Internet has become one of the most important, if not the most important, communication networks on the planet. Today, the Internet pervades almost every aspect of life and business. Along with its exponential growth comes the critical need to secure information in attached systems from unauthorized disclosure, transfer, modification, or destruction. With its wide reach and the potential ability to "reach out and touch" any attached node should that node's Internet address be known, the need for network security to guard network computer systems and protect electric data that is either stored in computer networks or transmitted over the networks, has become evident.

Internet security is a special branch of computer security. The main objective of Internet security is to protect the channel that is carrying the information and the storage devices that contain that information against various types of attacks. Even if you don't think you have anything worth protecting on your computer, it's still important that you keep it protected.

1.2 Attacks Overview and Classification

Due to Internet growth and the increase in the number of application programs using it for transport, the Internet has become a critical infrastructure. Attackers can utilize vulnerabilities to endanger your computer system's security. The attack's goal will be to impact or destroy one of the security requirements of the system such as integrity, confidentiality, availability, authorization, reliability, or accountability.

According to RFC 2828, network attacks are classified as either passive attacks or active attacks.

Passive attacks do not affect the network resources. They attempt to steal information stored in a system by eavesdropping (wiretapping). Consequently, it is difficult to detect passive attacks because they do not involve alteration of the data. According to Stallings, passive attacks are subdivided into two classes; release of message and traffic analysis [3]. With a release of message attack, the attacker merely needs to be able to observe unencrypted traffic and looks for clear text passwords and sensitive information that can be used in other types of attacks or may otherwise be of use.

In a traffic analysis attack, even if the messages are encrypted; the attacker might still be able to observe the patterns of those messages and determine the location and the identity of the communicating hosts. This information might be useful in defining the nature of the communication.

On the other hand, an active attack attempts to alter system resources or effect the system's operation. Additionally, it may attempt to delete or modify the data that is stored

on the computer; this is thus considered one of the most serious forms of attack. According to Stallings, the active attack can be subdivided into four categories: masquerade, replay, modification of message and denial of service [2].

A masquerade is a type of attack where the attackers pretend to be an authorized user of a system in order to gain access to it or to gain greater privileges than they are authorized. The masquerade may be attempted through the use of stolen logon IDs and passwords.

The next category is a replay attack, also known as a man-in-the-middle attack. The hacker in the replay attack tries to capture message parts and then resend a message sometime later with changes. Even though the messages may be encrypted, and the attacker may not know what the actual keys and passwords are, the retransmission of valid logon messages may be sufficient to gain access to the network.

The modification of message attack means that some portions of a legal message are altered or those messages are delayed or reordered to produce an unauthorized effect.

One more type of the active attack is the Denial of Service attack (DOS) which prevents network resources from providing services by limiting or denying access to those resources for the victims.

Lately, another form of DOS attack has appeared, known as the Distributed Denial of Service (DDOS) attack, sometimes called a botnet. The attackers attack the service from multiple sources, to make the job of protecting network resources by blocking the source much harder, and sometimes impossible.

The common Denial of Service Attacks are ICMP (Internet Control Message Protocol) Ping Flood, Smurf Attack, and TCP SYN Attack.

An ICMP ping flood is a simple Denial of service attack where the attacker overwhelms the victim with ICMP Echo Request packets. It only succeeds if the attacker has more bandwidth than the victim. The attacker hopes that the victim will respond with ICMP Echo Reply packets, thus consuming outgoing bandwidth as well as incoming server Bandwidth [3].

A Smurf attack is a DOS attack caused by ping flooding. The attacker sends a large number of ICMP echo or ping traffic to IP broadcast addresses, with each having a spoofed source address of the intended victim. If the routing device delivering traffic to those broadcast addresses delivers the IP broadcast to all hosts, most hosts on that IP network will take the ICMP echo request and reply to it with an echo reply. On a multi access broadcast network, hundreds, or even thousands of machines might reply to each packet, overloading the target and causing it to go offline or crash [7]. To prevent this attack, the user can configure the routers so that they do not forward the packets to the broadcast addresses. Another method is to organize host and routers so they do not respond to ping requests.

SYN flood is a form of denial of service attack in which an attacker sends a succession of SYN requests to a target's system. Primarily, when a client attempts to start a TCP connection to a server, the client and server exchange a series of messages called the TCP three-way handshake, which is the foundation for every connection, established using the TCP protocol.

During a TCP SYN attack, a malicious client does not send this last ACK message to server. If these half-open connections bind resources on the server, it may be possible to take up all these resources by flooding the server with SYN messages. Once all resources set aside for connections are exhausted, no new connections can be made, resulting in denial of service [8]. Internet Request For Comments (RFC) 4987 discusses many defense techniques that have been used to prevent this attack [9].

It's worth mentioning one other type of attack called the brute force attack which is related to encryption systems. The attackers in brute force try to find the weakness in an encryption algorithm. The strategy that the attacker uses is to attempt all possible keys in an encryption algorithm until the correct key is found.

1.3 Motivation of this Research

Measuring and monitoring the behavior of the network traffic is considered an important aspect of building, managing, and improving a large and complex computer system. This holds for computer networks, and is especially true for the Internet.

The network security administrator considers the network perimeter as a crucial point of any network. Therefore, companies spend a lot of time guarding against traffic that might enter their networks, monitoring it for unusual behavior which may be indicative of an intrusion. On the other hand they appear to not spend much time guarding against traffic that might leave their networks. Knowledge of where outbound communications are terminating, and why they are terminating there, may provide network administrators with an additional tool useful for uncovering malicious behavior on their network. To be useful, a network intruder must communicate with the outside

world. If an intruder has successfully established themselves, unnoticed, on some network, it may still be possible to detect their presence via unusual behavior with regard to the outbound traffic.

The long term goal of this research is to develop a tool to sample a network's outbound traffic, determine the traffic's physical destination, estimate the probable purpose of this communication, and alert a human operator in the event the communication is deemed suspicious or unusual. For example, a Pentagon computer found to be communicating with another computer in North Korea might meet this criterion. It is unknown as to how useful this proposed technique will prove to be as a knowledgeable attacker would likely hide behind another remotely controlled device resulting in a serious security breach possibly appearing to be an innocent communication. However as few publications regarding outbound traffic monitoring appear in the open literature, this is an area that needs to be explored in order to determine its usefulness.

The process of developing this tool can be subdivided into several stages with different objectives. The first objective of the research is to use Internet based search engines to find the probable physical location of outbound traffic IP addresses and define the owner of that IP address. The second objective of this research is to identify the probable type of system (router, server, user PC) at the observed destination IP address. After knowing the location, the owner, and the type of the system, the last objective would be to estimate the probable purpose of the communication between the two parties.

This Masters' dissertation focuses on the first objective and begins the development of the above monitoring tool. Outbound traffic captured over a 24 hour period at Oklahoma State University is used as a test sample. Additionally, changes in the traffic pattern over this 24 hour period are examined and analyzed.

1.4 Overview of this Dissertation

In the second chapter, user security approaches are surveyed, and the reader is given an overview of security approaches. Advantages and disadvantages of these methods are mentioned.

The third chapter explains work done to accomplish objective one. Code was written in JAVA to parse TCP Dump files provided by OSU IT in order to extract the destination IP addresses. Additional JAVA programs were developed that were used to query Internet databases in order to find information that is related to each destination IP address.

Chapter 4 presents and discusses the results of analyzing the outbound traffic of Oklahoma State University through one day in four different time periods.

Finally, in chapter 5 conclusions are presented with some suggestion for future research.

CHAPTER II

REVIEW OF LITERATURE

The purpose of this chapter is to introduce and give a brief summery about network security approaches. In general, network security is accomplished by different approaches as explained in the following sections.

2.1 Encryption and Decryption –Cryptographic Algorithms

Computer cryptography was created to protect confidential data in digital forms, and it has flourished in the Internet era. Different methods have been used to protect the transfer of the confidential data in the networks, including encryption and decryption algorithms. The encryption algorithm is used to change data from that which is readable (clear text) to that which is unreadable (cipher text). The opposite algorithm, which retrieves the original clear text from cipher text, is called decryption. The most known cryptographic algorithms are symmetric and public key algorithms [2, 10].

2.1.1 - Symmetric Key Encryption Algorithm.

The symmetric or secret key algorithm is also referred to as a conventional encryption algorithm. In this algorithm, both the encryption and decryption keys are the same, and the sender and receiver must agree on the key before any secured communication can take place between them. Basically, the symmetric key algorithms can be divided into two categories based on the used technique: Stream algorithms

(Stream ciphers) which encrypt the bits of the message one at a time, and Block algorithms (Block ciphers) which take a number of bits and encrypt them as a single unit [2, 10].

The most commonly used symmetric algorithms are the Data Encryption Standard (DES) and the Advance Encryption Standard (AES). Both of them are block cipher algorithms.

In 1977, the Data Encryption Standard (DES) was published by the US National Bureau of Standards (NBS). “The length of a DES encryption key is 56 bits, which was sufficient to resist brute force attacks in the period from 1970’s to 1980's. However, the 56-bit key length was no longer secure in the late 1990's due to advancements of computer technologies and algorithms [10].” In 1998, Electronic Frontier Foundation (EFF) built a special-purpose supercomputer, named "DES Cracker", to attack the DES algorithm, and they announced that the DES algorithm had been broken. Consequently, DES Cracker proved that the Data Encryption Standard is insecure [2, 10].

In 1998, a new algorithm called triple DES (3DES) extended the life of DES. Basically, triple DES involves repeating DES three times using either two or three different keys to each data block so that this process makes the effective key length 112 or 168 bits long [2].

Generally, applying DES multiple times can effectively extend the length of encryption keys without modifying the basic DES algorithm but this effort just slightly extended the life of DES. Triple DES also has some drawbacks. The main drawback of 3DES is that the algorithm is relatively slow in software and the secondary drawback is

that both DES and 3DES use a 64 – bit block size which is considered too short for reasons of the security and efficiency, and it is vulnerable to the meet-in-the-middle attack. A meet-in-the-middle attack is a cryptographic attack which targets block cipher cryptographic functions. This attack is use to reduce the number of brute force permutations required to decrypt text that has been encrypted by more than one key. In a meet-in-the-middle attack brute force techniques are applied to both the plaintext and cipher text of a block cipher by attackers. By applying various keys to encrypt the plaintext; the attackers try to get intermediate cipher text which has been encrypted by the first key. At the same time, they attempt to decrypt the cipher text according to various keys to find a block of intermediate cipher text that is the same as the one achieved by encrypting the plaintext. If two intermediate cipher texts match, the key used to encrypt the plaintext and the key used to decrypt the cipher text are the two encryption keys used for the block cipher [2, 10, 26].

Because of the drawbacks of 3DES, it was not a practical candidate for long term use. Therefore, in 1997 the U.S. National Institute of Standards and Technology (NIST) issued a call for proposals for a new Advanced Encryption Standard (AES). In 2001, NIST selected the Rijndael algorithm which was devised by Belgium cryptographers Joan Deamen and Vincent Rijmen. AES was issued as Federal Information Processing Standard (FIPS 197). AES divides the plaintext string into 128-blt blocks, and it can use encryption keys of three different key lengths of 128, 192, and 256 bits [10].

Regarding the attack methods, the symmetric key algorithms are attacked by using cryptanalytic attacks and the brute force attack. Cryptanalytic attacks rely on analyzing and deciphering codes, ciphers and cryptograms, and the brute force attack is a

strategy used to break the encryption of data. It involves traversing the search space of possible keys until the correct key is found [2, 10].

2.1.2 Public key Algorithm.

Before public key cryptography (PKC) was invented, the users involved in a communication must first agree on using the same secret keys over a secure channel. Due to this reason, a major breakthrough in computer cryptography was the invention of the public key cryptography. PKC addresses the major drawback in the previous algorithms which is the need for a secure key distribution channel. Thus, unlike symmetric key algorithms, a public key algorithm does not require a secure initial exchange of one or more secret keys between the sender and receiver [10].

The idea of public key algorithm is that two different keys should be used to transfer the message: a public key which is known to everyone, and a private key which is to be kept in tight security by each user. Each user creates both a public key and an associated private key. The sender encrypts the message using the intended recipient's public key; to decrypt the message, the recipient uses the private key [2, 10].

There are several public key algorithms, and they can be used for a variety of proposes. For example, the Diffie-Hellman key exchange is used to generate shared keys to use with symmetric algorithms, and the RSA public-key algorithm is used to encrypt data [14].

In 1976, Diffie and Hellman invented what is now known as the Diffie-Hellman algorithm (DH). DH is a method for securely exchanging a shared secret between two parties. The DH key exchange method allows two parties that have no prior knowledge of

each other to jointly establish a shared secret key over an insecure communications channel. A shared key is used by two users to independently generate keys for symmetric encryption algorithms that will be used to encrypt data between them [10, 11, 12]. The Diffie-Hellman algorithm is still playing an active role in important Internet security protocols such as Secure Sockets Layer (SSL), Secure Shell (SSH), and Internet Protocol Security (IPsec) [11].

In 1977, the RSA public algorithm was developed by Ron Rivest, Adi Shamir, and Len Adleman and published in 1978 [RIVE78]. The RSA is considered the most popular algorithm. In addition, RSA has resisted many years of extensive attacks. The manner of keeping RSA secure is by increasing the key length as opposed to DES, where the key length is fixed and short [2, 10, 13].

2.2 Data Authentication and Hash Function

Data Authentication is a procedure that allows communicating parties to verify that received messages are authentic. Generally, data authentication has two important aspects: to confirm the origin of the data and to convince the user that the data has not been modified or fabricated. Moreover, both data integrity and non-repudiation are maintained by using data authentication. Data authentication can be achieved either by using symmetric and public encryption algorithms or without using encryption algorithms [2]. Authentication using symmetric key is easily explained as in symmetric key only the sender and receiver share a key, so it is possible to perform authentication because only the legitimate sender can successfully encrypt a message for the other participant [2, 10].

Authentication is an interesting property of the public key algorithms. The use of private and public keys also allows protection of the authenticity of a message by creating a digital signature of a message using the private key, which can be verified using the public key. In addition to the previous two methods, there are several approaches of data authentication that do not rely on the message encryption. In those approaches, authentication is achieved by an authentication tag which is generated and added to transmitted message. An example of those approaches is Message Authentication Code (MAC). In the MAC, a secret key is used as input to generate a small block of data called tag that is appended to the message [2, 10].

Another approach used to authenticate data is called a Hash function. A Hash function is a procedure that takes variable length input data and produces fixed length data digest as output. Unlike the MAC, a hash function does not take a secret key as input. Moreover, a hash function assures both integrity and authentication as following. The hash function technique assumes that both sender and receiver share a common secret value. The secret value and the message are used by the hash function to calculate the hash and append it to the message. Once the recipient receives this message, he/she recalculates the hash of this message and then compares the result with the transmitted hash. If the hashes match, the message was not changed. It is not possible to the attacker to modify intercepted message as long as the secret value remains secret [2].

One of the Hash functions requirements is the one–way property. This property is important when the secret value is used in authentication technique. As mentioned previously that secret value itself is not sent, but if the hash function is not one way, attackers can easily discover the secret value. As a result, they can obtain the message

and the hash function. By applying this property and other properties to the hash function, this makes the hash function secure, and it is called secure hash function [2].

2.3 Secure Socket Layer

Secure Socket Layer (SSL) is a protocol developed by Netscape for transmitting private documents via the Internet. It was designed to make the use of TCP and to provide a reliable end to end secure service. SSL consists of two layers of protocols: the lower layer is SSL Record Protocol which provides the basic security services to various higher layer protocols, and the upper layer consists of three other protocols which are defined as part of SSL: the Hand Shake Protocol, the Change Cipher Spec Protocol, and the Alert Protocol. Also, the Hypertext Transfer protocol (HTTP) can operate on the top of SSL protocol [2].

Basically, the two important concepts of secure socket layer are SSL session and SSL connection. SSL connections are peer to peer relationships, and every connection is associated with one session. SSL sessions are created by the hand shake protocol, and each session defines the required parameters for an SSL connection such as protocol version, cryptographic security parameters, which can be shared among multiple connections [2].

2.4 Internet Protocol (IPv4 and IPV6 security).

When IPV4 was designed, security was not a very important feature because engineers were more concerned about the end to end model (how to deliver the data from source to destination). Therefore, the design of the Internet had no level of security and assumed that the security was the responsibility of the end nodes.

In 1994, the Internet Architecture Board issued a report entitled Security in the Internet Architecture (RFC1636). In response to the need to secure network architecture, a series of RFCs were issued to define IPSec, a protocol suite, which is used to define the policies and to ensure the various protections in IP networks [2, 10].

In 1998, the base architecture for IPSec was defined in (RFC2401). IPSec offers many types of protection services such as encryption of user data, authentication of the integrity of a message, and the protection against certain types of the other behavior. Those security capabilities were designed to be usable by both IPV4 and IPV6. In addition, IPSec is built into IPv6 to ensure the mandatory security.

Primarily, the fundamental concept of IPSec is a Security Association (SA). The (SA) is the establishment of one way logical connection between sender and receiver to support a secure communication between them, and it appears in both authentication and confidentiality [2]. In addition, three main facilities are provided by IPSec: an authentication only function referred to as an Authentication Hearer (AH) which is used to provide integrity and authentication to IP datagram, a combined authentication / encryption function called Encapsulating Security Payload (ESP) which is used to provide confidentiality services such as confidentiality of message connect and limited traffic flow confidentiality, and the third function is the key exchange [2, 13].

2.5 Firewalls.

One of the most well-known solutions in the network security is the firewall, and it is one of most effective forms of protection developed against hackers operating on the Internet. A firewall is a computer security device that is situated between an internal

network and the Internet. Also, it is placed at the junction point or gateway between two or more networks. A firewall can work at either hardware or software level to prevent unwanted outside access. The main task of firewalls is to monitor and to examine all incoming and outgoing traffic which allows controlling the traffic. If the traffic meets certain criteria, it is routed among the networks, otherwise; it is stopped. Furthermore, firewalls can protect both individual computers and corporate networks from hostile intrusion from the Internet [13].

Mainly, firewalls operate at different layers and use different criteria to restrict the network traffic. Based on the criteria, there are fundamentally four types of the firewalls: packet filtering firewalls, circuit level gateway firewalls, application level gateway firewalls, and stateful multilayer inspection firewalls [14].

Figure (II-1) depicts the main time lines of the industry of the firewall technology. Around 1985, Cisco's IOS (Internetwork Operating System) software division introduced the first generation of the firewall known as the Packet Filter firewall [16].

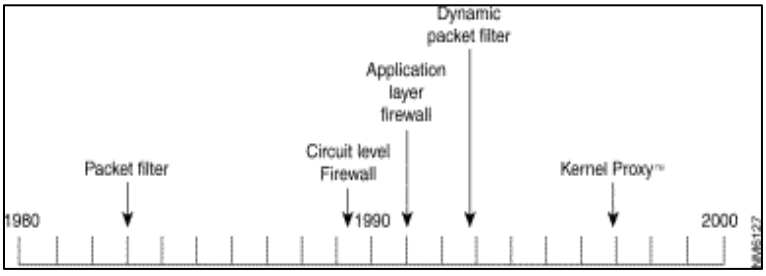


Figure (II-1) Time Line of Firewall Architectures [16]

Packet filtering firewalls work at the network level of the OSI model or the Internet protocol layer of TCP/IP model. They filter the packet based on their source and destination addresses, port numbers, and the used protocol. Generally, most routers

support packet filtering firewalls even if there are other firewalls used in the network [14].

The advantage of packet filtering firewalls is their low cost and low impact on network performance. Packet filtering is generally faster and easier to implement. On the other hand, packet filtering firewalls are susceptible to attacks. Attackers can tunnel malicious traffic through allowed ports on the filter [14, 15].

Secondly, in the late 1980's and early 1990's, AT&T Bell Laboratories pioneered the second generation of firewall architectures which are known as the circuit level gateway firewalls. They also implemented the first working model of the third generation of firewall architectures known as the application layer firewalls [15].

The circuit level gateway firewalls work at the session layer of the OSI model or the TCP layer of TCP/IP model. The main function of this firewall is to determine whether a requested session is legitimate or illegitimate by monitoring TCP handshaking to every connection setup. Thereafter, individual packets that are part of an approved session are not filtered. Therefore, traffic is filtered based on specified session rules and may be restricted to recognized computers. An advantage of circuit level gateway firewalls is that they hide the network itself from the outside which is useful for denying access to intruders. Also, they are relatively inexpensive. Thus, a circuit level gateway firewall is referred to as a transparent gateway [13, 14, 15].

The third type is the application level gateway firewall known also as a proxy server. Those firewalls can filter packets at the application layer of the OSI model. All internal computers establish a connection with the proxy server so that incoming or

outgoing packets can access services only via this proxy. They use application specific proxies to process and filter the incoming and outgoing information packets before copying and forwarding information across the gateway at the level of the application layer. Furthermore, the application level gateway firewalls are used to evaluate the content of packets. Also, they can be used to log user activity and logins. Although they offer a high level of security, they have a significant impact on network performance because they use more memory and processor resources [14, 15, 16].

Finally, the combination of the aspects of the three previous types of firewalls is called a stateful multilayer inspection firewall; they perform the previous three types of filtering. First, they filter packets at the network layer, then determine whether session packets are legitimate and then evaluate contents of packets at the application layer. The new idea in the stateful multilayer inspection firewalls is that they rely on algorithms to recognize and process the data instead of running application specific proxy. By using the stateful multilayer inspection firewalls, a high level of security is achieved, and good performance and transparency to end users are assured. On the other hand, due to their complexity they are expensive and can be difficult to configure and administer [14, 15, 16].

2.6 Intrusion Detection Systems

Intrusion is defined by Heady et al “as any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource [17].” Intrusion could be in many forms such as unauthorized user trying to access the resources in the system or network, malicious programs that spoil the system resources, or an authorized

user trying to gain additional privileges or access to confidential information, thus compromising the system's security policy [17].

Because the firewall was designed as a first layer of defense, another defense layer, called the Intrusion Detection System (IDS), was deployed to monitor attacks within an internal network system. An IDS system gathers and analyzes information from various areas within a computer or a network to detect Instruction and to identify the possible security actions [18, 19,27].

Generally, the idea of an IDS system is to analyze the traffic that traverses the network and compare this traffic against a loaded set of signatures of different types of possible attack patterns. These patterns can be used to identify such things as worms, viruses, and other network based attacks. When the IDS believes that it has found an attack, IDS performs an action or actions in response to a detected attack. There are various approaches of intrusion detection: network based (NIDS), host based (HIDS), signature based, and anomaly based intrusion detection systems [19, 27].

2.6.1 Host Based Intrusion Detection.

Host Based Systems (HIDS) were the first type of IDS to be developed and implemented. HIDS are run on individual hosts or devices on the network. A HIDS monitors the inbound and outbound packets and alerts the user or administrator if suspicious activity is detected [27].

2.6.2 Network Based Intrusion Detection.

Network based intrusion system (NIDS) analyzes data packets that travel over the actual network. These packets are examined and compared with empirical data to verify their nature: malicious or benign. NIDS is placed at a strategic point or points within the network to monitor traffic to and from all devices on the network. NIDS are best at detecting the following activities: unauthorized outsider access and bandwidth theft/denial of service. However, NIDS might create a bottleneck that would impair the overall speed of the network [18, 27]. A NIDS is used to inspecting incoming and ongoing network traffic.

2.6.3 Signature Based Intrusion Detection.

A signature based IDS, known as Misuse IDS, monitors packets on the network and compares them against a database of signatures or attributes from known malicious threats. Signature based detection techniques such as SNORT [20], USTAT [21], NSTAT [22], and P-Best [23] are limited in that they cannot detect new attacks. In addition, once a new attack is identified, there is a significant time lapse before the signature database is updated. Moreover, this type of techniques is well known for its high number of false positive alerts [19].

2.6.4 Anomaly Based Intrusion Detection.

Anomaly IDS is built by monitoring the behavior of the system over a period of time and by comparing its behavior against an established baseline. The baseline will identify what is “normal behavior” for that network such as the used bandwidth, the used protocols, and what ports and devices generally connect to each other. Anomaly IDS automatically classify statistically significant deviations from the normal behavior as

being abnormal. The advantage of this approach is that it is possible to detect unknown attacks. However, there is a potential for having a high rate of false positive alarms generated when the knowledge collected about normal behaviors is inaccurate [19].

Considerable previous work has addressed traffic, but while there are many secure approaches that focus on inbound traffic monitoring, there are fewer methods related to outbound traffic monitoring. Two methods mentioned earlier in this chapter were firewalls and IDS systems. However these techniques do no attempt to track the physical location of the traffic's destination. Few articles discuss the possibility of tracing the ultimate destination of outbound traffic. One such article is a report published by the ALEX e Solution Company in 2009, which discusses curtailing data theft via outbound traffic monitoring. The author discussed how this type of monitoring may help to reduce the risk of malware programs. The authors suggest monitoring all outbound connections to the Internet and analyzing outbound traffic. The approach is known as defense-in-depth. By applying this approach the theft of data might be traced, and destinations where malware sends the data might be known [25].

Another area of research related to our research is explained in [24]. In this article the researchers have investigated the use of available online resources to show the behavior of use of Internet. In [24] "The key hypothesis of their work is that most of the information needed to profile the Internet endpoints is already available around us on the web [24]." Based on the same hypothesis, we start this research. Our objective is different but we are also using the Internet as source of the information to find the location and the owner of the IP address. Eventually, additional research performed by a

follow-on student will attempt to use the Internet to identify the purpose of the outbound traffic to each IP address.

CHAPTER III

METHODLOGY

3.1 Overview of the Work.

This chapter explains the work that has been accomplished in developing the first stage of a tool to sample and classify a network's outbound traffic. Using the data provided by the OSU IT department in the form of TCP Dump files, programs were written in JAVA to find the information that is related to each destination IP address.

The first part of the code was to parse the TCP Dump file to extract the destination IP addresses from each line of the TCP Dump file. Because there were many users connected to the same IP website's addresses such as eBay and yahoo, many IP destination addresses were repeated. Therefore, duplicated IP addresses were removed prior to applying the code to query Internet search engines. As a result, the traffic that was sent to the Internet was reduced, and the program runtime speeds up.

The second part of the code was the use of Internet based search engines to find the probable physical location of outbound traffic IP addresses and to define the owner of each IP address. Two websites, http://www.dnsstuff.com and http://ip-address-lookup-v4.com, were employed in the code. Moreover, to make this program more general and to use it as an automatic query to look up IP address information, additional code was used to determine the detailed information that is

related to each IP address as it is fetched from the website's server. Because the IP addresses are distributed in different parts of the world, the format of the information is different and depends on the database sources that the websites use. For example, the DNS Stuff website uses more than five database formats such as The American Registry for Internet Numbers (ARIN), Réseaux IP Européens Network Coordination Centre (RIPE), and African Network Information Center (AFRINIC). The developed JAVA code parses the returned information and extracts the required information however it is provided by these websites.

The third part of code is the storage of retrieved information in a clear form. Access database tables were used to store the owner and location of each IP address. Additionally, text files were employed to save the detailed information from each TCP Dump file.

Generally, the main program is divided into three classes (Remove duplication class, sort class, and query website and the storage class); the main program and the three classes are explained by flow charts in the next section. In addition, the program code is attached in the appendix.

3.2 The Flow Charts of the Code.

3.2.1 The Flow Chart of the Main Program.

Figure (III-1) is drawn to give a general idea of the flow of the program, and it has three classes. Classes are the fundamental building blocks of a JAVA program. Each class has some fields (variables), which can be primitives or references to objects, and methods which define actions that a class's objects (or instances) can do. The method

defines the same as the function in other programming languages which has variables and code to perform some actions. Moreover, the class name is used to access each method in a class from the other classes.

Initially, the TCP Dump files are loaded into the main program, which will parse each line of the file and extract the IP destination addresses and store them in a string array. After that, the Remove Duplication class is called to remove the repeated IP addresses from the extracted IP addresses and store them in the string array. In order to save the IP addresses numerically by byte, the sort IP class is applied to the IP addresses. Finally, the program queries one of the two Internet search engines to find the estimated location and the owner of those IP addresses. This is done by the query website class. The storage of this information in Access Database is also completed in this class.

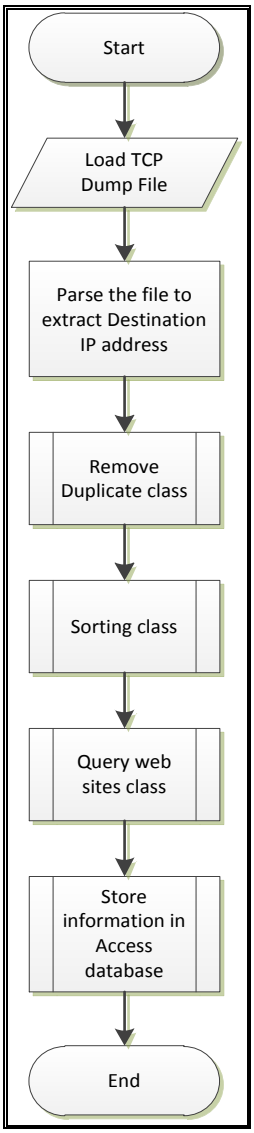


Figure (III-1) the main program's flow chart

3.2.2 The Flow chart of Remove Duplicated Class.

As was mentioned previously, the extracted IP addresses from the TCP Dump file were saved as the string element in a one dimensional array. By using a special implemented class in JAVA called List, the string array was converted to array list. The array list is resizable-array implementation of the List interface. The main point of changing array to array list is that array list can grow and shrink dynamically, at runtime.

Arrays cannot. Consequently, a set of methods in this class can be applied that facilitates the programming to remove repeated IP addresses. Another method in this class was applied to convert the list to a set of objects. For more explanation, the set is a collection of objects that cannot have duplicate values so that by applying the two last methods of Array List class to the IP addresses array the repeated IP addresses were removed. Furthermore, the resize method is supported in the Array List class to manipulate the size of the set, so the set will have the correct number of elements including null element. Finally, before returning the IP addresses to the main program to apply the sorting algorithm, the set of objects was changed back to an array of string elements. Figure (III-2) shows the main steps in the remove duplicated class

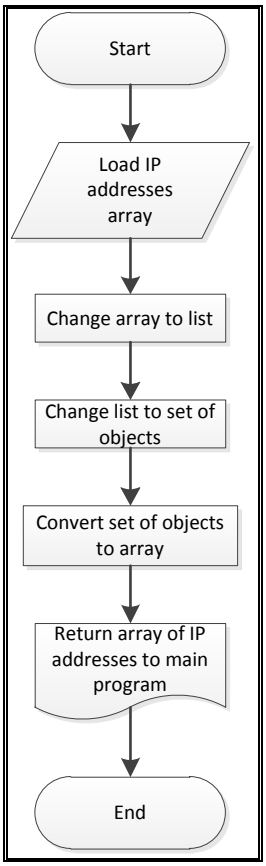


Figure (III-2) the Remove duplicate class's flow chart.

3.2.3 The Flow Chart of Sorting Class.

In order to rapidly find a specific IP address by hand, should a user so require, the IP addresses need to be sorted by bytes. Entries should initially be sorted based on the first byte of each IP address. Entries with identical first bytes should then be sorted numerically based on the second byte, etc. The standard numerical IP address is a combination of a characters (periods) and numbers, so it is difficult to use one of the common string sorting algorithms directly for a sort of this nature because of the format of IP address. In order to get a proper byte based numerical sort, the first step of this class was to parse each IP address and to save it into two dimensional arrays of integer elements. Each row of the array represents the four bytes of an IPv4 address. After saving the IP addresses as integer elements, the Bubble Sort algorithm was applied to sort the IP address in ascending order. The basic JAVA class that was used in sorting class is called StringTokenizer which was used to parse the IP addresses. The following flow chart shows the steps of the sorting class. Each IP address is compared with all IP addresses. First step is to compare the first byte if byte is out of the order; the swapping operation is done between IP addresses. The same comparison does to the other three bytes of each IP address to put the IP addresses in the correct order. The sort class's flow chart is drawn in figure (III-3).

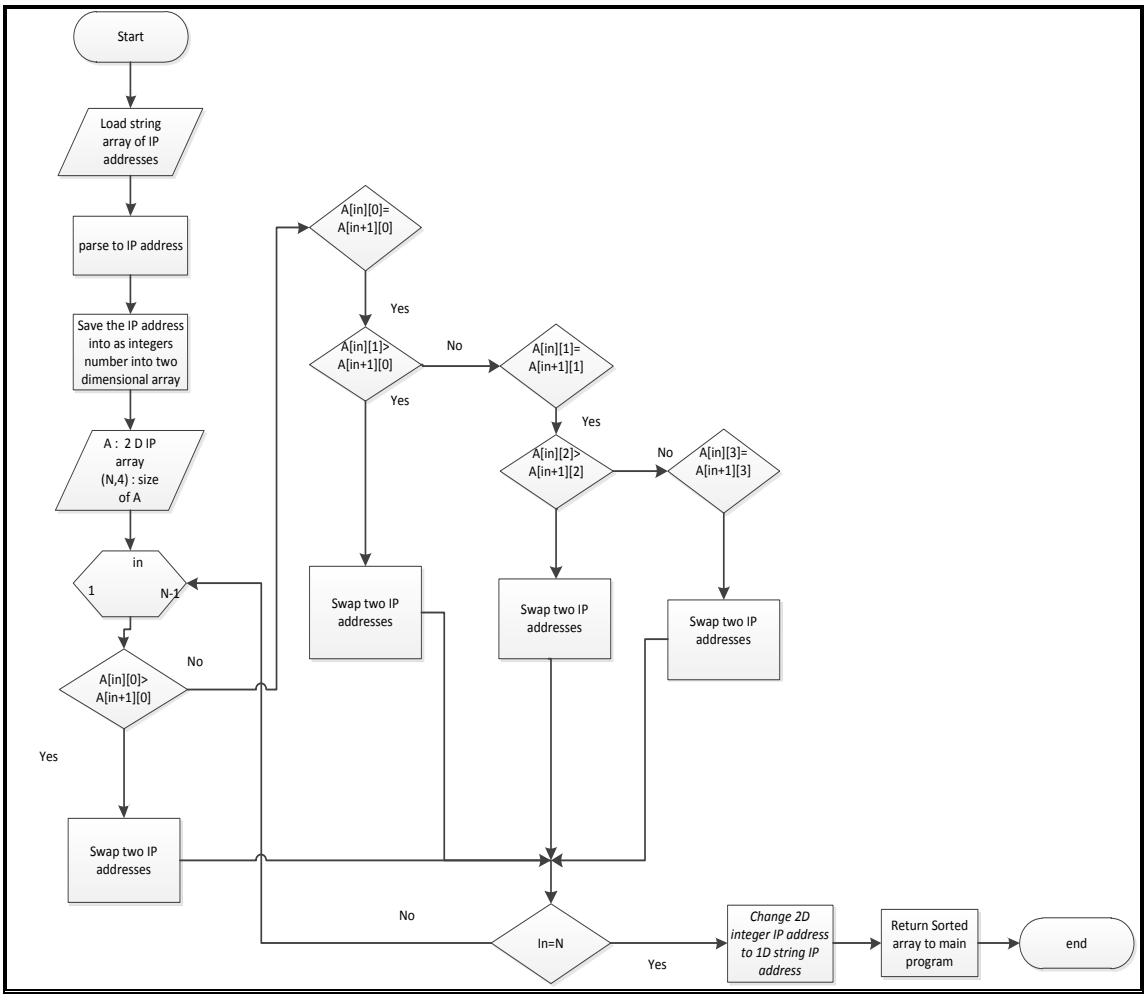


Figure (III-3) the sort class's flow chart

3.2.4 The Flow Chart of Query Website Class.

This class primarily accomplishes three main tasks; the first task is the search into the Access database to see if the information of this destination IP address is already known. Generally, this task is considered as an extra step in this stage of the research because each TCP Dump IP address was stored in a separate table to study the pattern of the traffic over time, so there were no repeated IP addresses. But as explained in the first chapter, the final objective of this project is to develop a tool to sample a network's outbound traffic; therefore, the result of all TCP Dump files will be stored in the same

table, so some IP destination addresses will be repeated among TCP Dump files. For more explanation, the remove Duplication class removes the repeated IP address within the same TCP Dump file, and this function is to ensure that, for this thesis, there are no IP addresses sent more than one time to the geolocation website among all the TCP Dump files, so the heavy traffic can be avoided and the run time will be reduced.

After duplicate IP addresses are removed, the second task of this class is applied. Each IP address was inserted into a properly formatted packet and the DSN STUFF website was queried in order to determine the location and the owner of the IP address. The server of the website replied to the query in the form of Hypertext Markup Language (HTML) page. This reply contained all the information that is related to the IP address according to the used database as well as some external links to the other websites. The HTML page was read and saved as the text file. Then specific key words that have relation to the location and the owner were searched for. Because the website uses different database sources to up load their information, each database format was manipulated in a different way to get the requested information.

The search operation for the specific key words in text file was applied. The words such as owner, descr, and Org Name were looked for to find the owner of the IP address. In addition to those key words, in some IP addresses the owner was written between two symbols in some database format, so those symbols were parsed and the owner was extracted.

To find the location, the “country” and “location” words were the keys. The country field in some database formats did not contain the country name but it had the

abbreviation of the country's name, therefore, extra code was written to map this abbreviation to the country's name.

By using the DSN STUFF website as the primary source for this part of the code, many locations of IP addresses were unknown, so IP addresses were sent to a second website (IP Address Lookup) to find their location, and the mapping code also was applied to find the country's name.

The storage task is the last part of the code. The first step of this third task was to create the connection between the Access Database and NetBeans IDE. Note that NetBeans IDE is both a platform framework for JAVA desktop applications and an integrated development environment (IDE) for developing with JAVA [27]. After establishing the connection, each IP address with its location and owner was sent to the Access database table in ascending order. All TCP Dump IP addresses were separately saved in the table, so the results of each file can be analyzed, and the pattern of OSU traffic was shown for different time periods. Figure (III-4) parts A, B explain query website class's flow chart.

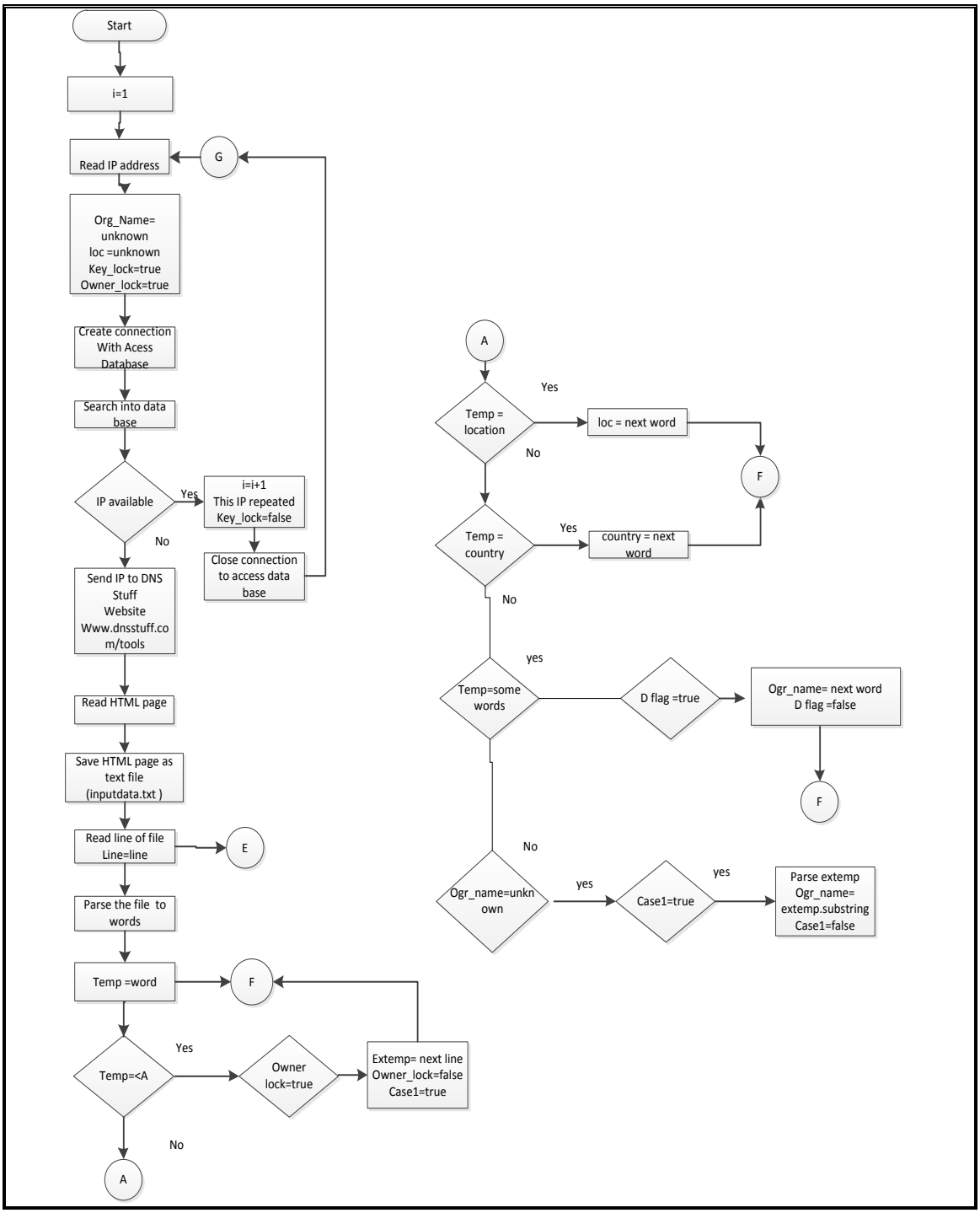


Figure (III-4) the query website class's flow chart (part A)

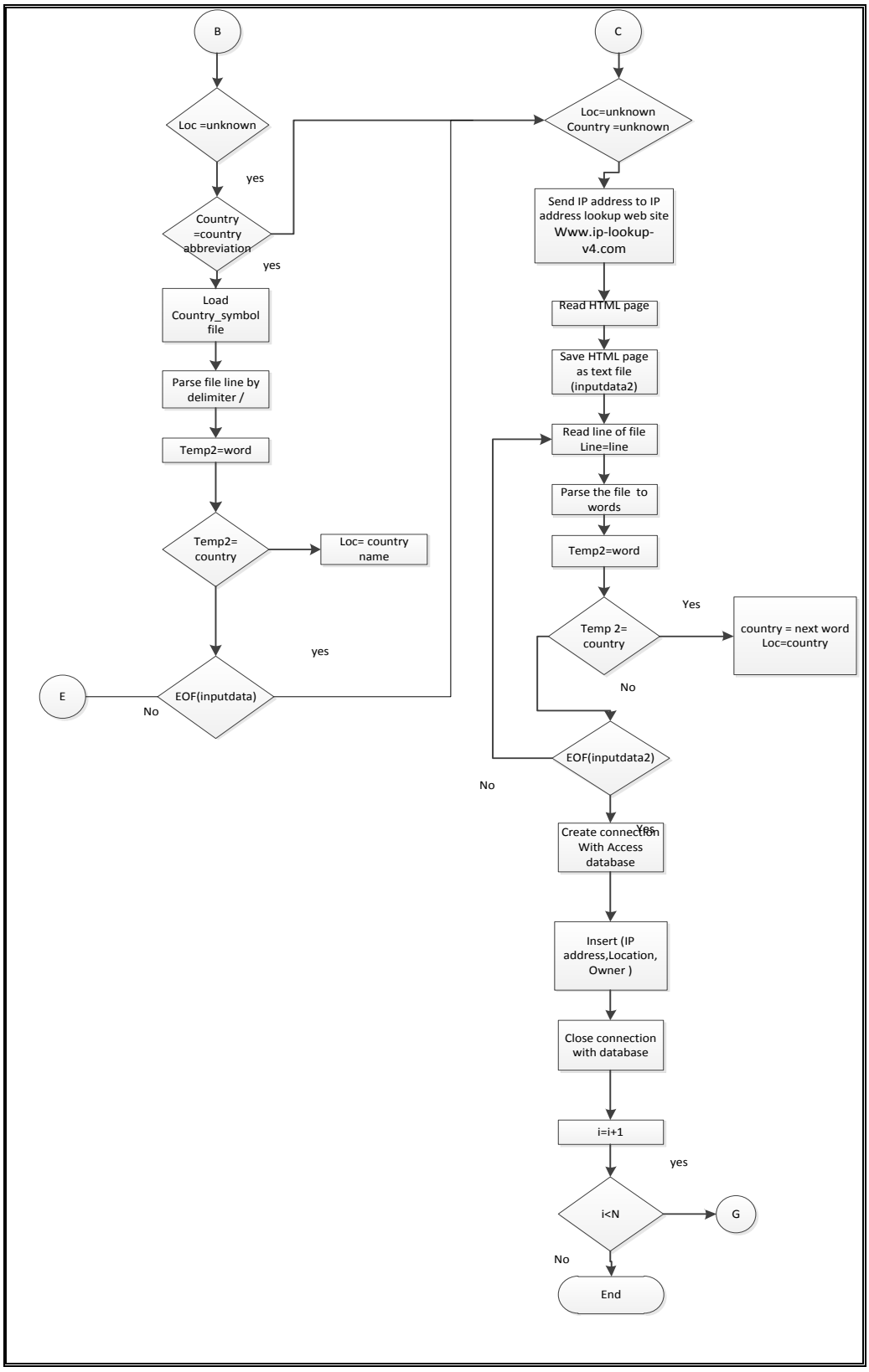


Figure (III-4) the query website class's flow chart (part B)

3.3 The Program Graphic Interface.

The graphic interface is an application to make the use of this tool easier to the user. It is not necessary for the user to know the code steps of the program to run this code. Basically, the interface consists of two sections: IP Analysis and IP search as shown in figure (III-5).

IP Analysis has three buttons, the first button is used to browse the files and up load the TCP Dump file. After that, there are two options to run and save the retrieved information from websites by choosing one of two Run and save buttons. Based on the user's choice, the results will be saved either in the Access database or text file for the chosen TCP Dump file.

The second part of the graphic interface is titled as IP search. The search operation is done within the available database; the search is done by either the IP address or the country name. By writing the IP address in text field and clicking the search button, the search operation returns the information that belongs to this IP address. If the IP address is not found in the database, a message is displayed which explained that this IP is not located in the database. Another option is the searching by country name by selecting the country name from country's list and presses the search button, all IP addresses that are located in this country are displayed on screen, or if there are no IP addresses located in this country, a message is displayed to show that database does not have IP addresses in this country.

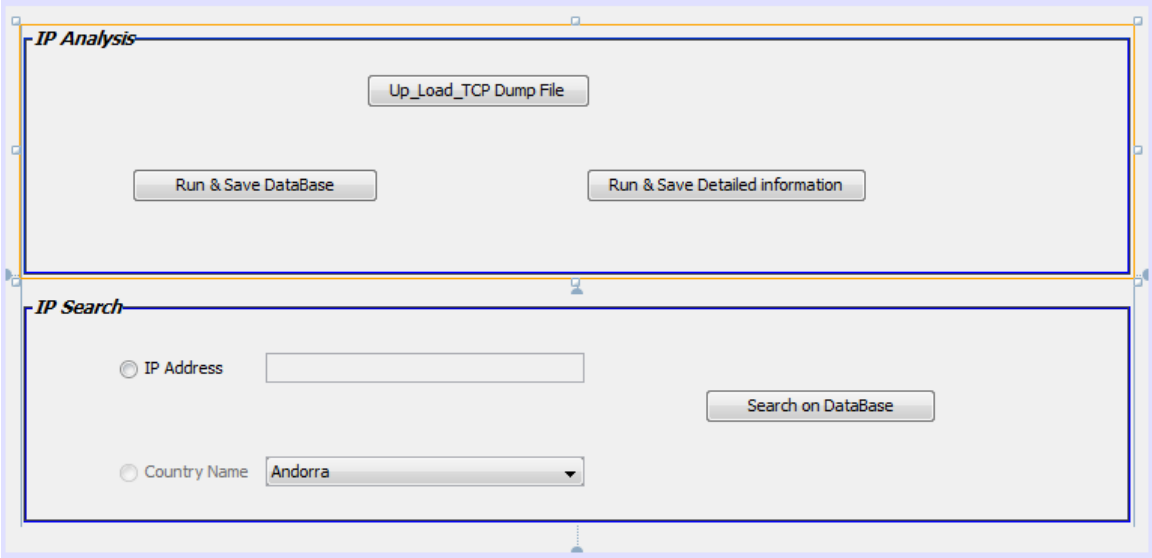


Figure (III-5) Graphic user interface.

CHAPTER IV

FINDINGS

Results Analysis

This chapter describes the statistical results of an analysis of the location and the owner of destination IP addresses to show the pattern of OSU outbound traffic. The charts are drawn by using Excel program based on the data that has been obtained from two Internet websites.

The pattern of the outbound traffic was investigated by analyzing the results of the TCP dump files. The outbound traffic was captured four times over a 24 hour period, at 9:00 am, 3:00 pm, 9:00 pm, and 3:00 am, on the 18th and 19th of May, 2010. Two websites were queried to get the information about the destinations by searching on the database of these websites for a given IP address. The location and owner were extracted from the retrieved information. Because the majority of IP destination addresses were located in the United States, the United States was excluded in the following investigation.

The first analysis was based on the location of the destination IP addresses. Two statistics were calculated by using the location of the IP address; the location analysis was done by considering the first ten countries that own the largest number of IP address locations at each time. Moreover, the percentage of the outbound traffic destinations to each continent was determined by taking the top ten countries from each continent.

The first TCP Dump file was captured on May, 18th at 9:00 am. The total unique IP destinations were 4898. 2068 destinations were detected in the USA, and the other destinations were located in 132 different countries. Table (IV-1) gives the names of the countries that possess the majority of these destination addresses. Five countries were located in Europe, and three countries were located in Asia, in addition to Canada and Brazil. Brazil held the largest number of the destination IP locations. In addition, the number of the destinations in Russian and the United Kingdom was almost the same while the time is different between them where Russian is 13 hours and United Kingdom 7 hours ahead of the United States. Because some countries span multiple time zones, the local time was calculated based on the capital city of each country.

Country	IP Address	Local Time
Brazil	185	noon
Russian Federation	143	10:00 Pm
United Kingdom	142	4:00 Pm
Germany	135	5:00 Pm
India	127	8:30 Pm
China	121	11:00 Pm
Bulgaria	115	5:00 Pm
Canada	108	10:00 Am
Italy	108	5:00 Pm
Poland	88	4:00 Pm

Table IV-1. The top ten countries, 9:00 am capture.

Figure (IV-1) illustrates the percentage of the traffic. There was not much difference in the percentage of the traffic that departed from the OSU to Europe and Asia at this time. The percentage of traffic was 40% to Europe and 34 % to Asia. Besides, South America held 14% of the traffic. Both Africa and North America (excluding the U.S.) had the same amount of the traffic at 5%.

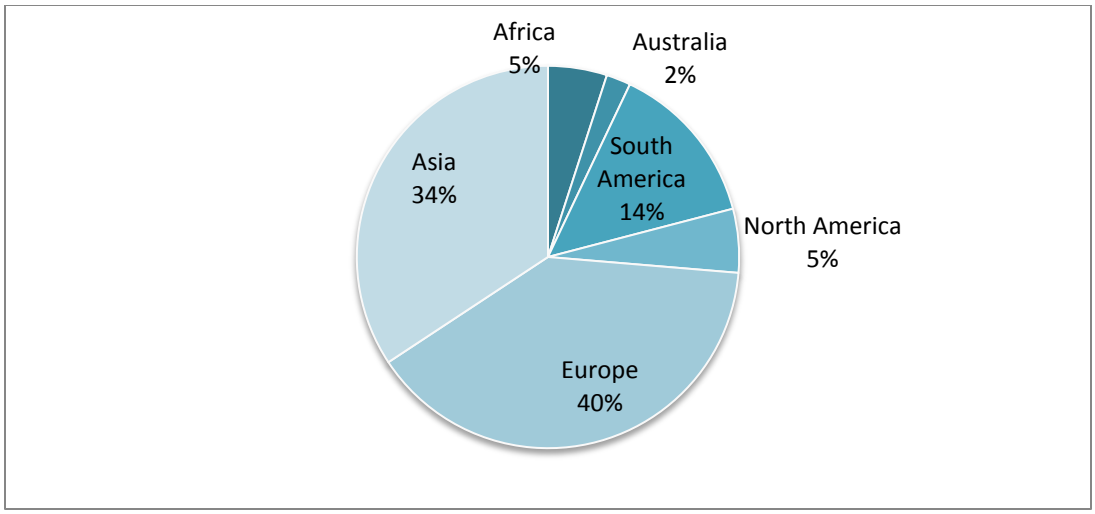


Figure (IV-1) the percentage of IP destinations by continent, 9:00 am capture.

The second TCP Dump file was collected on May, 18th at 3:00 pm. The total number of the destination IP addresses was 4239. The USA possessed 2196, and the remanding IP addresses were located in 115 different countries. Table (IV-2) below shows the location of the most IP destination address at 3:00 pm, again excluding the USA. The largest number of the IP address was located in the United Kingdom where the local time was 10:00 pm. Brazil had the second largest number of destination addresses at the same time although the time zones differ between the Brazil and United Kingdom as is noted in the next table.

Country Name	IP Address	Local Time
United Kingdom	145	10:00 Pm
Brazil	134	5:00 Pm
Canada	112	4:00 Pm
Bulgaria	108	12:00 (Noon)
Russian Federation	94	4:00 Am
Germany	90	11:00 Pm
China	89	4:00 Am
France	83	11:00 Pm
Poland	75	11:00 Pm
Italy	74	11:00 Pm

Table IV-2. The top ten countries, 3 pm capture.

Figure (IV-2) shows the percentage of the traffic to each continent, 46 % of the outbound traffic departed from the OSU to Europe, and 24 % of the traffic was directed to Asia. Moreover, South America had almost the same amount of the traffic as at 9:00 am, 15% of the total traffic. The smallest percentage of the outbound traffic left to Australia, and it was 3%.

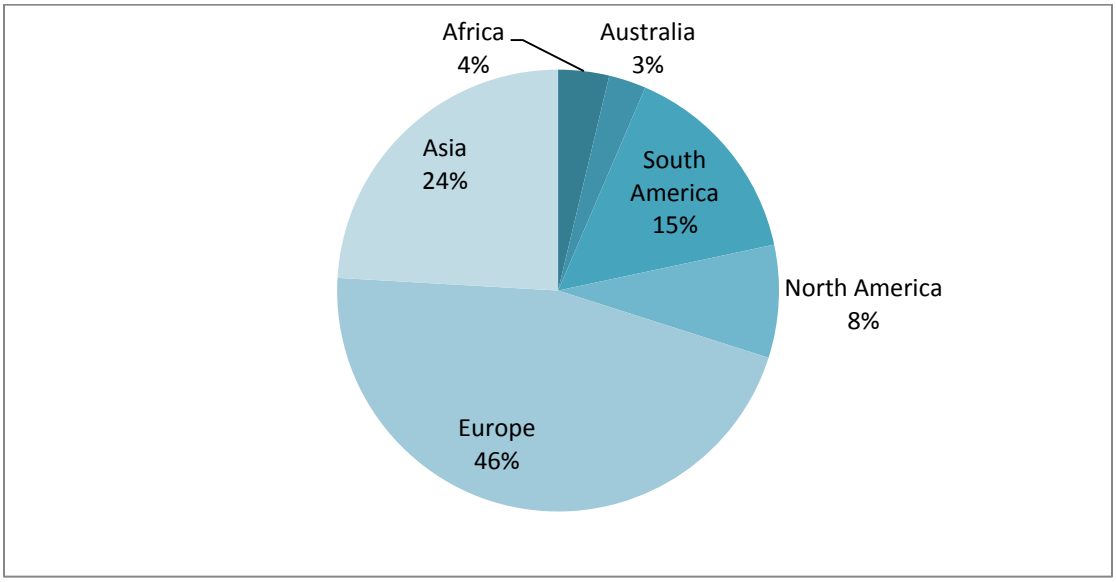


Figure (IV-2) the percentage of IP destinations by continent, 3:00 pm capture.

The third TCP Dump file was captured on May, 18th at 9:00 pm. The behavior of outbound traffic at this time was different from the two previous times, and the reason for this behavior is the time difference between the continents. The total number of the IP destination addresses at this time was 4015. 1985 users were connected to destination addresses located in the USA while the other IP addresses were distributed among 96 countries. Table (IV-3) below provides the name of the countries that have the most IP addresses at 9:00 pm. It is clear that six countries are located to Asia, and that the majority of destinations were located in China and Malaysia where both countries are 14

hours ahead of the United States, and their local time is 12:00 AM. Moreover, the number of the connected users to China at this time increased, and the country occupied the first position comparing by the previous time where it occupied the seventh position. The number of the users that connected to Brazil declined at this time from 134 users at 3:00 pm to 75 users at 9:00 pm where the local time is 11:00 pm. In addition, Canada had the third number of IP locations in both 3:00 pm and 9:00 pm where it is one hour ahead of the United States.

Country name	IP Address	Local Time
China	291	12:00 Am
Malaysia	138	12:00 Am
Canada	136	10:00 Pm
Taiwan	128	10 :00 Am
Japan	94	11:00 Am
United Kingdom	89	3:00 Am
Bulgaria	76	5:00 Am
Russian Federation	76	11:00 Am
Brazil	75	11:00 Pm
Korea	68	noon

Table IV-3. The top tenth countries, 9 pm capture.

The percentage of the outbound traffic is shown in figure (IV-3). Most of the OSU traffic left to Asia, and its percentage was 53 % of the total traffic. The second continent was Europe, and it had 23 % of total traffic. The pattern at this time was the opposite behavior from the previous times. Moreover, the smallest percentage of traffic was directed to Africa, and it represented 1% of the total traffic.

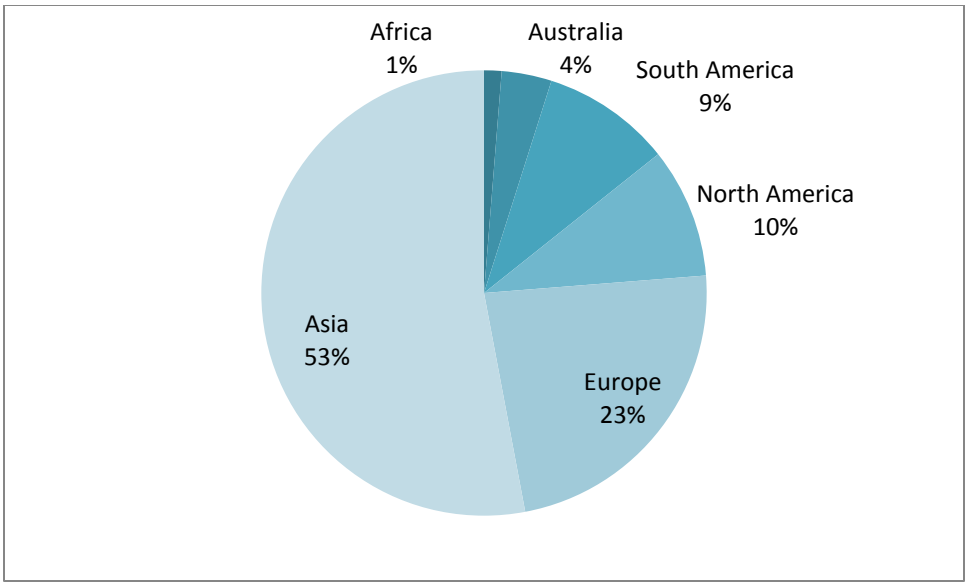


Figure (IV-3) the percentage of IP destinations by continent, 9:00 pm capture.

The last TCP Dump file was collected on May, 19th at 3:00 am. A large amount of outbound traffic was captured at this time, and the total IP destinations were 7508. 1960 IP addresses were located in the USA, and the other IP locations were distributed in 138 diverse countries. Table (IV-4) shows the top ten countries at this time; it appears from the table that India had the largest number of the IP location at 3:00 am because India is 11:30 hours ahead from the United States so the local time in India was 2:30 pm. China also had large number of the destination IP addresses where the time is 5:00 pm in China. From the table, it is obvious that Canada and Brazil were not listed on the table due to their local time at this time thus the top ten counties were distributed in Asia and Europe.

Country	IP Address	Local Time
India	506	2:30 Pm
China	313	5:00 Pm
Russian Federation	292	5:00 Pm
United Kingdom	263	10:00 Am
Italy	255	11:00 Am
Viet Nam	246	4:00 Am
Germany	229	11:00 Am
Poland	222	11:00 Am
Taiwan	222	5:00 Pm
Japan	187	6:00 Pm
France	182	11:00 Am

Table IV-4. The top ten countries, 3:00 am capture.

The percentage of the traffic is drawn in figure (IV-4), 88 % of the traffic was sent to Asia and Europe where Asia received 48% and Europe 40% of the outbound traffic. In addition, both Africa and North America had the same percentage of the traffic at this time, and the percentage was 4%.Simliarly, Australia and South America received 2% of the traffic at this time.

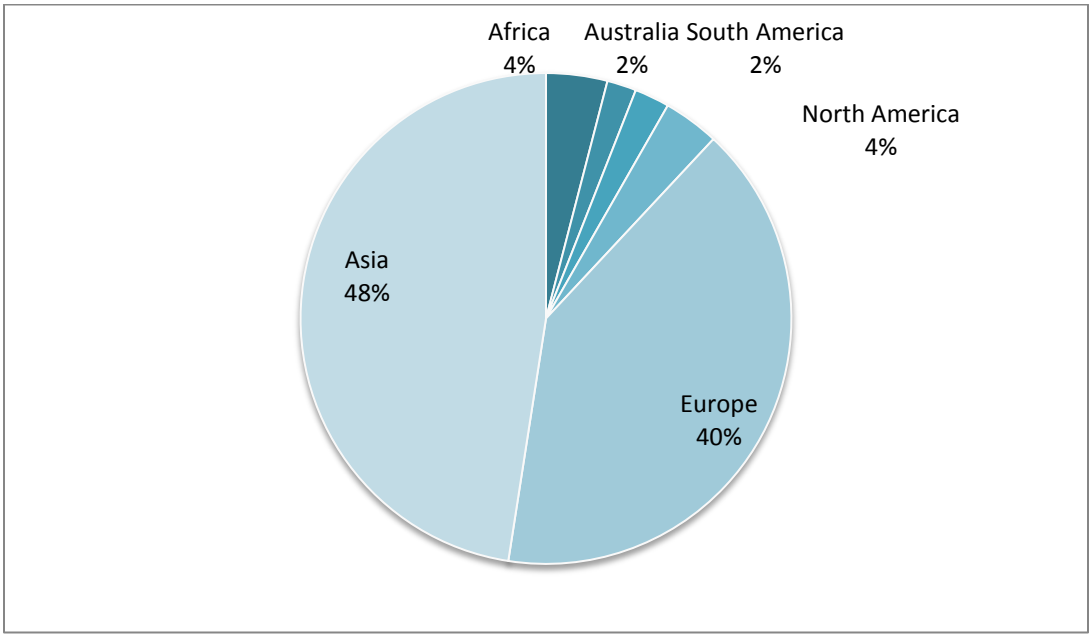


Figure (IV-4) the percentage of IP destinations by continent, 3:00 am capture.

To summarize the most countries that owned the IP addresses in four times, figure (IV-5) depicts comparison of the results of each county in four times. It was obvious that the most of the outbound traffic left the OSU to two world regions Asia and Europe. Moreover, South America had reasonable amount of the traffic in some times of the day. The change of pattern was based on the time difference between Asia and Europe. In general, the amount of traffic that was sent to Asia was more than the amount of the traffic to Europe because the number of the Asian students at OSU is more than the number of the European Students. Regarding the countries, the most destination IP addresses were distributed in 16 countries as explained in figure (IV-5).

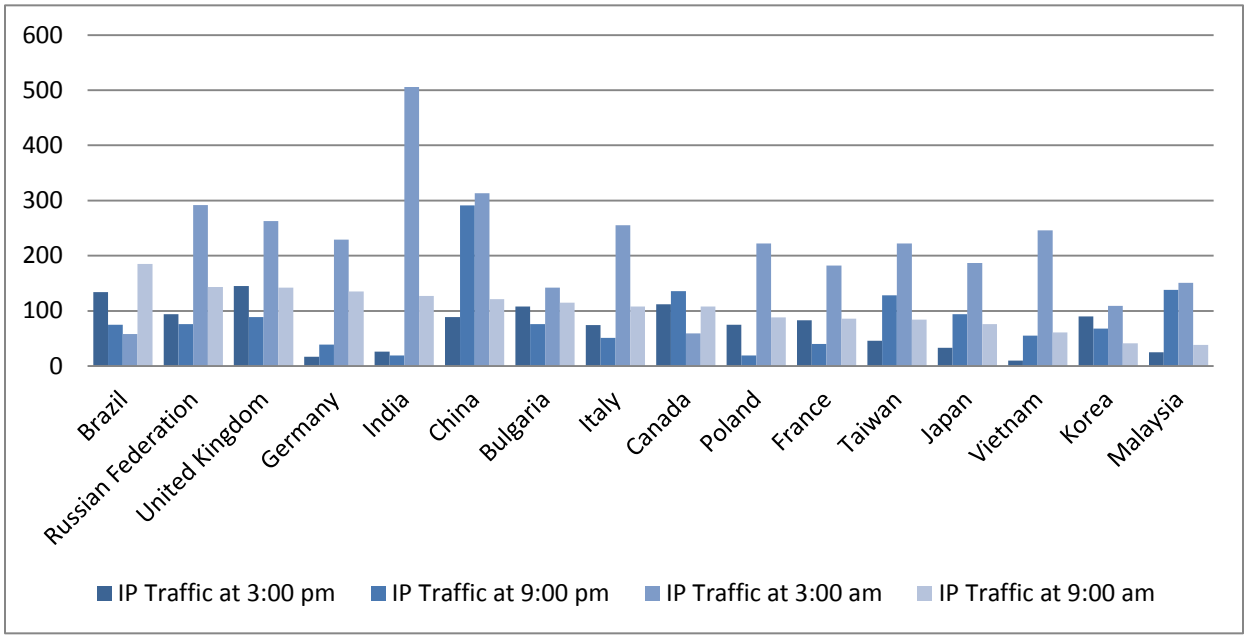


Figure (IV-5) the number of the location in each country during different times.

The second analysis of the results was based on the owner of the IP destination addresses to demonstrate how students use the internet and which sites they access. The percentage of IP destination addresses that belong to some common websites was included in the following statistics. It is clear from table (IV-5) that there was generally

no significant difference at the first three times while the percentage of IP addresses that owned by this websites declined in the last time ,which mean that the number of users who access those websites is declined at this time.

Website Name	The Access Percentage to Website			
	9:00 am	3:00 pm	9:00 pm	3:00 am
Educational & university websites	4.06	4.29	4.13	1.76
Google	2.14	2.35	1.86	0.63
Yahoo	1.14	1.17	0.44	0.22
Facebook	0.95	1.03	0.69	0.14
Skype	0.28	0.37	0.37	0.20
EBay	0.16	0.21	0.12	0.10

Table IV-5. The percentage of IP addresses of common websites.

CHAPTER V

CONCLUSION

The objective of monitoring outbound traffic research was to use of the Internet search engine resources to trace the location and owner of IP destination addresses. This objective was achieved by querying two websites (http://www.dnsstuff.com and http://ip-address-lookup-v4.com) and by using the available information in their databases. The querying of websites was by using JAVA codes, and to test the code, headers of outbound Oklahoma State University traffic was used. JAVA codes were also used to analyze the received data from the websites to monitor the destination IP addresses of the outbound traffic and to get the required information.

The location and owner of this IP addresses were found by searching on the data from two websites. One difficulty that was faced in this code was working with different Internet databases such as The American Registry for Internet Numbers (ARIN), Réseaux IP Européens Network Coordination Centre (RIPE), and African Network Information Center (AFRINIC). At all observed times, most of IP addresses were located in the USA, and the remaining IP addresses where distributed among different countries in the world. The percentage of unknown locations at different times did not exceed 0.3% in the four TCP Dump files. The results are provided in graphical and tabular manner to show where the majority of destination IP addresses are located. In addition, the owners

of IP addresses were defined. All JAVA codes were put in a form to be useful to the user who wants to use this code to sample network traffic captured by TCP Dump. This form is the Graphical Interface. One part of the Graphical Interface can be used to run this code and analyze the traffic, and another part can be used to search for the information belonging to each IP address, or it can be searched by country.

Future work

This was the first stage of a program to sample the network's outbound traffic by using Internet based search engines to find the probable physical location of outbound traffic IP addresses and to define the owner of that IP address. In the future work two objective should be done to complete this research. The first objective is to define the type of system (router, server, PC owned by a teenage hacker, etc.…) at each destination IP address. Finally, by using the previous information and if necessary the probable purpose of the communication between the two parties should be estimated.

REFERENCES

1. G.C. Kessler, "An Overview of TCP/IP Protocols and the Internet," Nov 9, 2010. [Online]. Available: http://www.garykessler.net/library/tcpip.html. [Accessed Dec. 13, 2011].
2. W. Stallings, Data and Computer Communications. 8th ed., Pearson Prentice Hall, 2007, pp. 701–740.
3. D. Adkins, K. Lakshminarayanan, A. Perrig, and I. Stoica, *"Taming IP Packet Flooding Attacks"*, In *ACM SIGCOMM HotNets II* (2003).
4. "CA-1997-28 IP Denial-of-Service Attacks," May 26, 1998. [Online]. Available: http://www.cert.org/advisories/CA-1997-28.html. [Accessed Dec. 13, 2011].
5. M. Hayman, "Windows 95 and NT Internet-related Exploits," [Online]. Available: http://cws.internet.com/article/1469-.htm. [Accessed Nov. 29, 2011].
6. R. Naraine, "Windows 7, Vista exposed to 'teardrop attack'," Sep 8, 2009. [Online]. Available: http://www.zdnet.com/blog/security/windows-7-vista-exposed-to-teardrop-attack/4222. [Accessed Dec. 13, 2011].
7. "CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks," Mar13, 2000. [Online]. Available: US-CERT Vulnerability Notes, http://www.cert.org/advisories/CA-1998-01.html. [Accessed Dec. 13, 2011].
8. " CERT® Advisory CA-2000-21 Denial-of-Service Vulnerabilities in TCP/IP Stacks," Dec 4, 2000. [Online]. Available: US-CERT Vulnerability Notes, http://www.cert.org/advisories/CA-2000-21.html. [Accessed Dec. 13, 2011].

9. W. Eddy, "TCP SYN Flooding Attacks and Common Mitigations," Aug 2007. [Online]. Available: http://tools.ietf.org/html/rfc4987. [Accessed Dec. 13, 2011].
10. J. Wang, *Computer Network Security Theory and Practice.*, Higher Education Press, Beijing and Springer-Verlag GmbH Berlin Heidelberg, 2009,
11. D.A. Carts, "A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols," Nov 5, 2001. [Online]. Available: http://www.sans.org/reading_room/whitepapers/vpns/review-diffie-hellman-algorithm-secure-internet-protocols_751. [Accessed Dec. 13, 2011].
12. W. Diffie & M. Hellman, "New Directions in Cryptography," *Ieee Trans. Inform. Theory*, vol. 22, no. 6, pp. 644–654, Nov 1976.
13. M.W. Murhammer, O. Atakan, L.Pugh, K.Suzuki, D.Wood & S. Bretz, *TCP /IP Tutorial and Technical Overview*. 6th ed., Prentic -Hall,Inc, 1989.
14. "Firewalls," 2002. [Online]. Available: http://www.vicomsoft.com/learning-center/firewalls. . [Accessed Dec. 13, 2011].
15. T. Bradley, "Introduction to Firewalls," [Online]. Available: http://netsecurity.about.com/od/hackertools/a/aa072004.htm. [Accessed Dec. 13, 2011].
16. "Evolution of the Firewall Industry," 1992-2002. [Online]. Available: http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ch3.htm. [Accessed Dec. 13, 2011].
17. R. Heady, G. Luger , A.Maccabe& M.Servilla, "The architecture of a network level Intrusion detection system," Technical Report, , Department of Computer Science, University of New Mexico, Aug 1990.

18. M. Weiss, "Network Defense: The Attacks of Today and How Can We Improve," Master dissertation, Dep. Telecommunications Engineering Technology, Rochester Institute of Technology, Rochester, NY, 2009.
19. Y. Al-Nashif, "Multi-Level Anomaly Based Autonomic Intrusion Detection System," Ph.D. dissertation, Dep. Electrical and Computer Engineering, University of Arizona, 2008.
20. M. Roesch, "*Snort - Lightweight Intrusion Detection for Networks*," *Proceedings of LISA '99:13th Systems Administration Conference,* Seattle, Washington, USA, USENIX Association, 1999, pp. 230-238.
21. K.Ilgun , "USTAT: a real-time intrusion detection system for UNIX," *Research in Security and Privacy, 1993 IEEE Computer Society Symposium on* , pp.16-28, May 1993.
22. G.Vigna, R. Kemmerer , "NetSTAT: a network-based intrusion detection approach," *Computer Security Applications Conference, 1998, Proceedings., 14th Annual* , pp.25-34, Dec 1998.
23. U.Lindqvist, P.A. Porras,, "Detecting computer and network misuse through the production-based expert system toolset (P-BEST)," *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on* , pp.146-161, 1999.
24. I.Trestian, S. Ranjan ,A. Kuzmanovic, A. Nucci , "Googling the Internet: Profiling Internet Endpoints via the World Wide Web," *Networking, IEEE/ACM Transactions on* , vol.18, no.2, pp.666-679, Apr 2010
25. R. *Spanier,* " Curtailing Data Theft via Outbound Traffic Monitoring, ",ALEX eSolutions, vol.7,no.12,Dec 2009

26. “meet-in-the-middle attack, ",February 2010, [Online]. Available: http://searchsecurity.techtarget.com/definition/meet-in-the-middle-attack. [Accessed Dec. 13, 2011].

27. P. Kazienko, P. Dorosz **“**Intrusion Detection Systems (IDS) Part 2 - Classification; Methods; Techniques, “Jul, 2004, [Online]. Available: http://www.windowsecurity.com/articles/ids-part2-classification-methods-techniques.html. [Accessed Feb. 13, 2012].

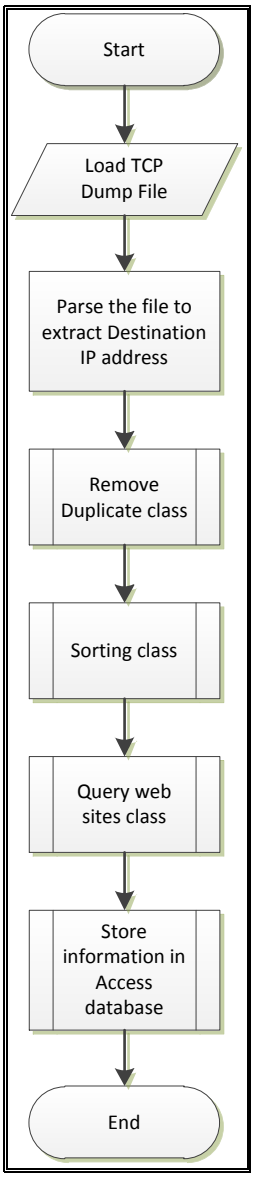
APPENDIX A

APPENDIX A1

A1-JAVA Code of the Main Class.

```
import java.io.*;
import java.lang.String.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.regex.Pattern.*;
import java.util.StringTokenizer;

public class MainProgram {

    public static void main(String args[]) throws InterruptedException {
        String p;
        String TEMPDES;
        String TEMPSRC;
        String DESIP;
        String temp;
        String DES[];

        String[] Temp_DES = new String[65000];
        int i = 0;
        // define object to call RemoveDuplicate class
        RemoveDuplicate var = new RemoveDuplicate();
        // define object to call sorting class
        IP_Sorting sort = new IP_Sorting();
        // define object to call detail result class
        detail_Result var1 = new detail_Result();
        // define variable to call the class that access the website and do mapping
country symbol to location of IP
            SendIP_StoreDB var3 =new  SendIP_StoreDB ();

// Read TCP Dump file and extract IP destination addresses and store them in array
try {
            FileInputStream fis = new
FileInputStream("C:\\Users\\AMAL\\Documents\\thesis_programs\\2100051810.txt");

            BufferedInputStream bis = new BufferedInputStream(fis);
            DataInputStream dis = new DataInputStream(bis);
            while (dis.available() != 0) // check for the input string
```

```
            {
               p = dis.readLine();
               int IN1 = p.indexOf("IP");
               int IN2 = p.indexOf("> ");
               int IN3 = p.indexOf(": ");

               TEMPSRC = p.substring(IN1 + 2, IN2);
               TEMPDES = p.substring(IN2 + 2, IN3);

// Separate destination IP from port No using StringTokenizer JAVA class
               StringTokenizer st = new StringTokenizer(TEMPDES, ".");
               int len = st.countTokens();
               String IP1 = st.nextToken();

               String IP2 = st.nextToken();
               String IP3 = st.nextToken();
               String IP4 = st.nextToken();
               DESIP = IP1 + "." + IP2 + "." + IP3 + "." + IP4;
               Temp_DES[i] = DESIP;
               i++;
            }
            fis.close();
            bis.close();
            dis.close();

         } catch (Exception e) {
            System.out.println("ERROR");
         }

// calling remove function from Remove Duplicate class
         DES = var.Remove(Temp_DES, Temp_DES.length);
         DES[0]="0.0.0.0";
 // call sort class to sort IP function from Sort class
        DES=sort.Sort_IP(DES);
 // call the Get_Search_StroreDB function from SendIP_StoreDB to send IP address
to website and retrieve the information that is related to this IP
          try {
        var3.Get_Search_StoreDB(DES);

         } catch (IOException ex) {
            Logger.getLogger(MainProgram.class.getName()).log(Level.SEVERE, null, ex);
         }

 // call the class detail_ result for getting the detail information
            //  var1.DSN_Data(DES);
            }*          // end the class
```

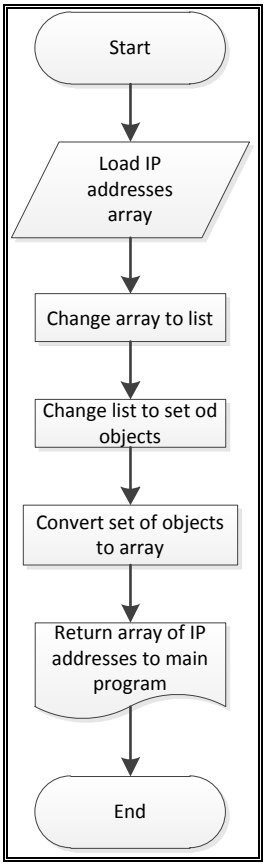


The flow chart of the main class.

APPENDIX A

A2-JAVA Code of Remove Duplication IP Class.

```
import java.util.Arrays;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
public class RemoveDuplicate {
      // A string array with duplicate values
      public  String [] Remove( String[] data,int len){
       // Convert it to list as we need the list object to create a set object.
      // A set is a collection object that cannot have a duplicate values, so
      // by converting the array to a set the duplicate value will be removed.
      List<String> list = Arrays.asList(data);
      Set<String> set = new HashSet<String>(list);
            // Create an array to convert the Set back to array. The Set.toArray()
      // method copy the value in the set to the defined array.
            String[] result = new String[set.size()];
      set.toArray(result) ;
               return (result);
   }}
```



The flow chart of remove duplication IP class

A3-JAVA Code of Sorting IP Class.

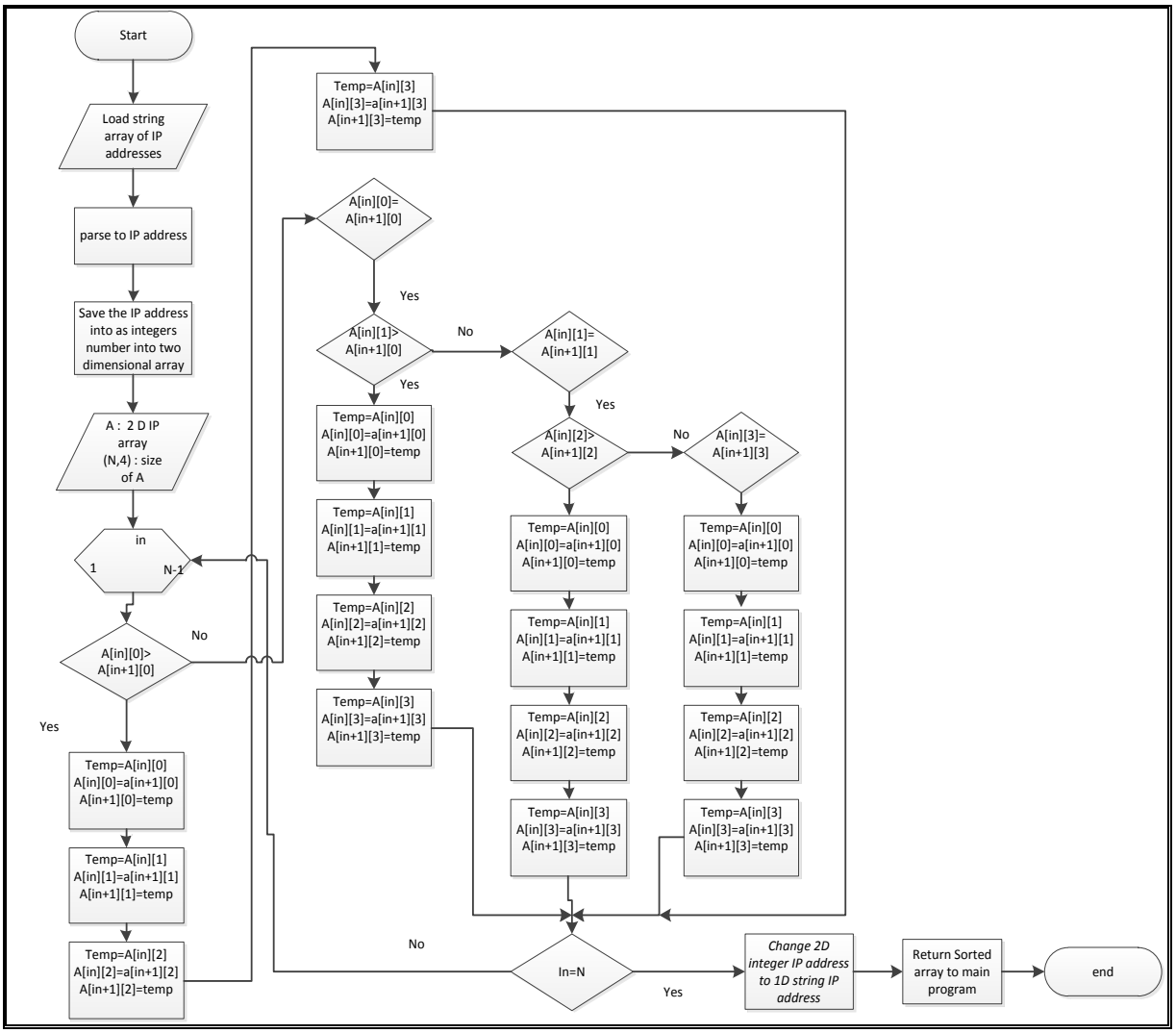
```
import java.util.StringTokenizer;

public class IP_Sorting {

   public String [] Sort_IP (String[]ipstring) {

        string [] SortedIp = new String[ipstring.length];

          int [][] a = new int[ipstring.length][4];

      // remove (.) from IP address and changed IP parts to  four parts integer numbers and store them in two dimensional array

          for(int j =0;j<ipstring.length;j++){

                    StringTokenizer st1 = new StringTokenizer(ipstring[j], ".");

                    int i = 0;

                    while(st1.hasMoreTokens()){

                    a[j][i] = Integer.parseInt(st1.nextToken());

                    i++;

                    }

          }

      // Apply Bubble Sort algorithm to integer two dimensional array

          int pass, in, temp;

for (pass=1; pass < ipstring.length; pass++) { // count how many times

// this next loop becomes shorter and shorter

        for (in=0; in < (ipstring.length) - pass; in++) {

          if(a[in][0] > a[in+1][0])    // out of order?

                                        {

                                        temp = a[in][0];

                                        a[in][0] = a[in+1][0];
```

```
                    a[in+1][0] = temp;

                    //swap(in, in+1);     // swap them

                    temp = a[in][1];

                    a[in][1] = a[in+1][1];

                    a[in+1][1] = temp;

                    //swap(in, in+1);     // swap them

                    temp = a[in][2];

                    a[in][2] = a[in+1][2];

                    a[in+1][2] = temp;

                    //swap(in, in+1);     // swap them
                    //swap(in, in+1);     // swap them
                    temp = a[in][3];

                    a[in][3] = a[in+1][3];

                    a[in+1][3] = temp;

                    //swap(in, in+1);     // swap them

                    }
if (a[in][0] == a[in+1][0])
                        {
                        if(a[in][1] > a[in+1][1])    // out of order?
                        {
                        temp = a[in][0];
                        a[in][0] = a[in+1][0];
                        a[in+1][0] = temp;
                        //swap(in, in+1);     // swap them
                        temp = a[in][1];
                        a[in][1] = a[in+1][1];
```

```
                    a[in+1][1] = temp;

                    //swap(in, in+1);     // swap them

                    temp = a[in][2];

                    a[in][2] = a[in+1][2];

                    a[in+1][2] = temp;

                    //swap(in, in+1);     // swap them
                    //swap(in, in+1);     // swap them

                    temp = a[in][3];

                    a[in][3] = a[in+1][3];

                    a[in+1][3] = temp;

                    //swap(in, in+1);     // swap them

                    }
if(a[in][1] == a[in+1][1])    // out of order?
                      {
                      if(a[in][2] > a[in+1][2])
                    {
                    temp = a[in][0];

                    a[in][0] = a[in+1][0];

                    a[in+1][0] = temp;

                    //swap(in, in+1);     // swap them

                    temp = a[in][1];

                    a[in][1] = a[in+1][1];

                    a[in+1][1] = temp;

                    //swap(in, in+1);     // swap them

                    temp = a[in][2];
```

```
            a[in][2] = a[in+1][2];

            a[in+1][2] = temp;

        //swap(in, in+1);     // swap them

        //swap(in, in+1);     // swap them

            temp = a[in][3];

            a[in][3] = a[in+1][3];
            a[in+1][3] = temp;

            //swap(in, in+1);     // swap them

            }

if(a[in][2] == a[in+1][2])

            {

            if(a[in][3] > a[in+1][3])

                    {

                        temp = a[in][0];
                        a[in][0] = a[in+1][0];
                        a[in+1][0] = temp;

                        /swap(in, in+1);     // swap them
                        temp = a[in][1];

                        a[in][1] = a[in+1][1];

                        a[in+1][1] = temp;

                        //swap(in, in+1);     // swap them

                        temp = a[in][2];

                        a[in][2] = a[in+1][2];
                        a[in+1][2] = temp;

                        //swap(in, in+1);     // swap them

                        //swap(in, in+1);     // swap them

                        temp = a[in][3];
```

```
                a[in][3] = a[in+1][3];
                a[in+1][3] = temp;
                        //swap(in, in+1);     // swap them
                }       }               }               }               }       }

                for(int j= 0; j<ipstring.length;j++)
                {
                for(int i=0;i<4;i++)
                  {     if(i==0)
                {       SortedIp[j] = Integer.toString(a[j][i]);
                }
                else{
                   SortedIp[j] = SortedIp[j] +"." +Integer.toString(a[j][i]);
                }               }               }
            return (SortedIp);
}}
```



the flow chart of sort IP class

A4-JAVA Code Query Website Class.

// this program to extract the owner and location information to store On Access Data base table to each IP.

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
public class SendIP_StoreDB {

   public static void Get_Search_StoreDB(String[] IP) throws IOException,
InterruptedException {
// variable definition
              boolean Dflag=true;
              boolean case1=false;
              boolean owner_lock = true;
              String loc="Unknown";
              String Ogr_name="Unknown";
              String City="Unknown";
              String country="Unknown";
              String extemp = null;
              String x="~}";
             String Url = "jdbc:odbc:firstdb";
             String ip=null;
             boolean key_lock = false;
            String username="amel";
            String password="amel";

  // Search into table database for each IP before send it to website

 for (int i=1;i<IP.length;i++ ){

loc="unknown";Ogr_name="Unknown";City="Unknown";country="Unknown";

  ip = IP[i];
        key_lock=true;

         try {
// create connection between Java and Access database

      Connection conn = DriverManager.getConnection(Url,username,password);
            Statement st = conn.createStatement();

       String query="SELECT *FROM STable_2100051810" + " IP_Address ";
              ResultSet rs = st.executeQuery(query);
              while(rs.next()){
                 String Key=rs.getString("IP_Address");
                 if (Key.equalsIgnoreCase(ip))
```

```
{System.out.println("Iam repeated IP "+Key);key_lock=false;}

    }
// close the data base
          conn.close();
          st.close();

       } catch (Exception e) {
          System.err.println("Got an exception! ");
          System.err.println(e.getMessage());
       }// end search

 // if the Enter IP does not exist in the DataBase , I will send it to website to find its information and store them in Database
            if (key_lock==true){
// query first website www.dnsstuff.com
  try {
    URL url = new
URL("http://www.dnsstuff.com/tools/whois/?tool_id=66&token=&toolhandler_redirect=0&ip="+IP[i]);
                    BufferedReader reader = new BufferedReader(new
                    InputStreamReader(url.openStream()));
                    BufferedWriter writer = new BufferedWriter(new
                    FileWriter("inputdata.txt"));
                    int count_line=0;
                    int limit=0;
                    String line;
                    while ((line = reader.readLine()) != null) {
                    count_line++;
                    if((count_line>130)&(limit<=60)){
                    writer.write(line);
                    writer.newLine();
                    limit++;
                 }
                //end while
                }

          reader.close();
          writer.close();
          //end try statement
       } catch (MalformedURLException e) { e.printStackTrace();}
       catch (IOException e) {e.printStackTrace(); }

//   Get information for each IP from the website and store them into file named (inputdata) to extract the owner and location.
```

```
File file = new File("inputdata.txt");

try {
      Dflag=true;
      Scanner scanner = new Scanner(file);
      while (scanner.hasNextLine()) {
       String line = scanner.nextLine();
       Scanner sc1 =new Scanner (line);
       sc1.useDelimiter(" ");
       while (sc1.hasNext()){
       String temp= sc1.next();
       if(temp.equalsIgnoreCase("(<A")){if(owner_lock==true){ extemp=line;
owner_lock=false; case1=true;}}
        if (temp.equalsIgnoreCase("Location:"))
       {loc=sc1.nextLine().trim(); }
        if (temp.equalsIgnoreCase("City:"))
       {City=sc1.nextLine().trim();}
        if (temp.equalsIgnoreCase("Country:"))
       {country=sc1.nextLine().trim(); }
       if (temp.equalsIgnoreCase("descr:" ) ||
(temp.equalsIgnoreCase("OrgName:"))||(temp.equalsIgnoreCase("owner:"))||(temp.equals
IgnoreCase("descr:                descr:" ))){
            if(Dflag==true){
                if(sc1.hasNext()){
                Ogr_name=sc1.nextLine();
                 if (Ogr_name.contains(x))                    {
                  System.out.println("ok");
              }
              else {
              Dflag=false; System.out.println("not ok");
              }
                }            }
       }
       }
    }
} catch (FileNotFoundException e) { e.printStackTrace();}
```

// end of get the owner and location and country of the IP,

Next there are some special format // Special data format to find the owner of IP case 1

```
 if (Ogr_name.equalsIgnoreCase("Unknown"))
{
if (case1==true)
{
 int prfex= extemp.indexOf("(");
```

```
        Ogr_name=extemp.substring(0, prfex);
    }
    case1=false;
    }
```
//Mapping country symbol to country name if location is unknown some IP has the country symbols so here I changed this country symbol to country name.
// symbol to location based on the file " country_symbols.txt"

```
        if (loc.equalsIgnoreCase("Unknown")){
            File symbol = new File("country_symbols.txt");
           try {
          Scanner scanner1 = new Scanner(symbol);
            while (scanner1.hasNextLine()) {
            String line = scanner1.nextLine();
            Scanner scanner2 =new Scanner (line);
            scanner2.useDelimiter("/");
             while (scanner2.hasNext()){
            String intemp= scanner2.next().trim();
                        if (intemp.equalsIgnoreCase(country)){
                      loc=scanner2.next().trim();
          }
        }
        }

     } catch (FileNotFoundException e) {
        e.printStackTrace();
     }
  }
```

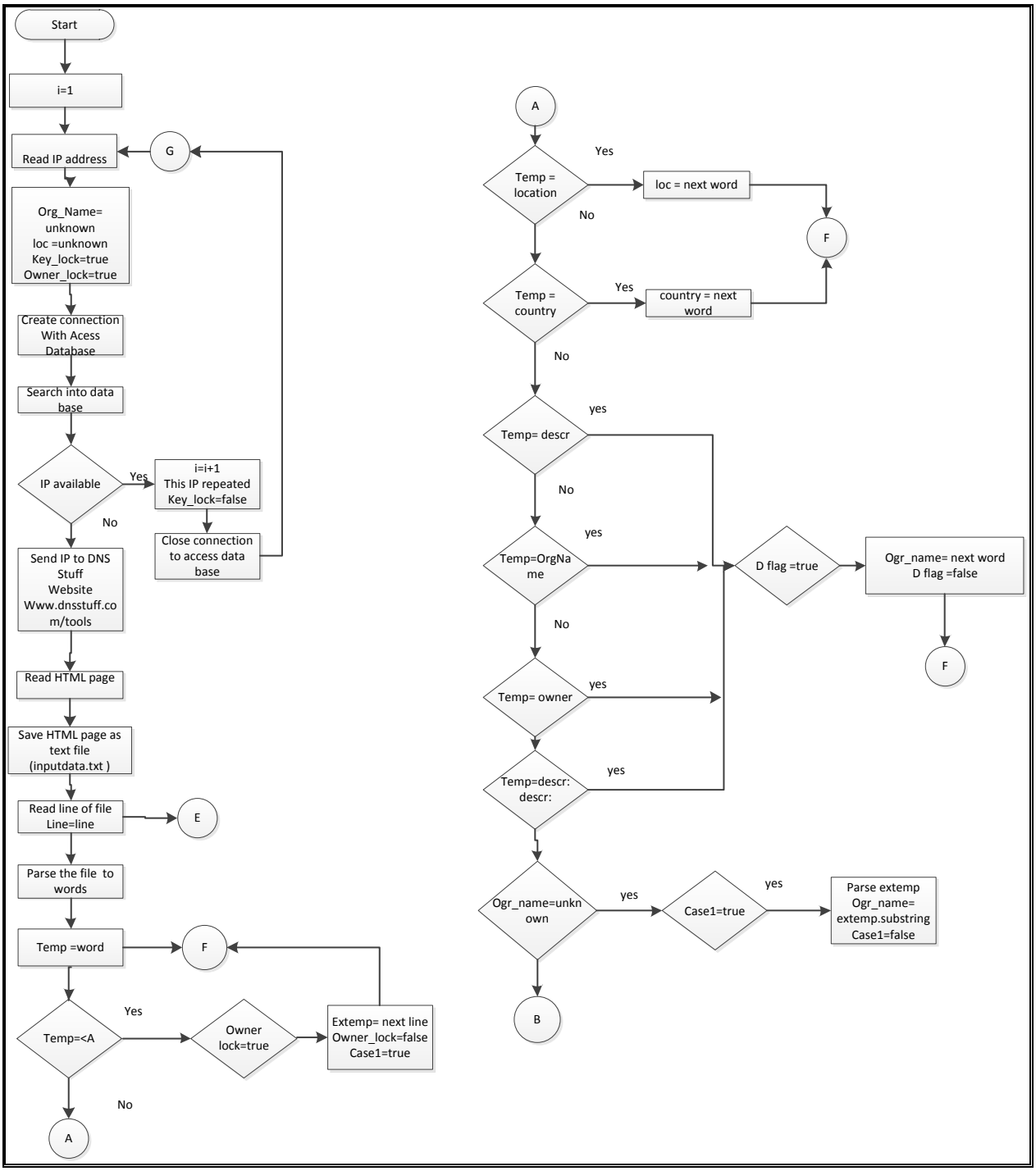
// if the location is still is unknown query the second website ip-address-lookup-v4.com & send IP address

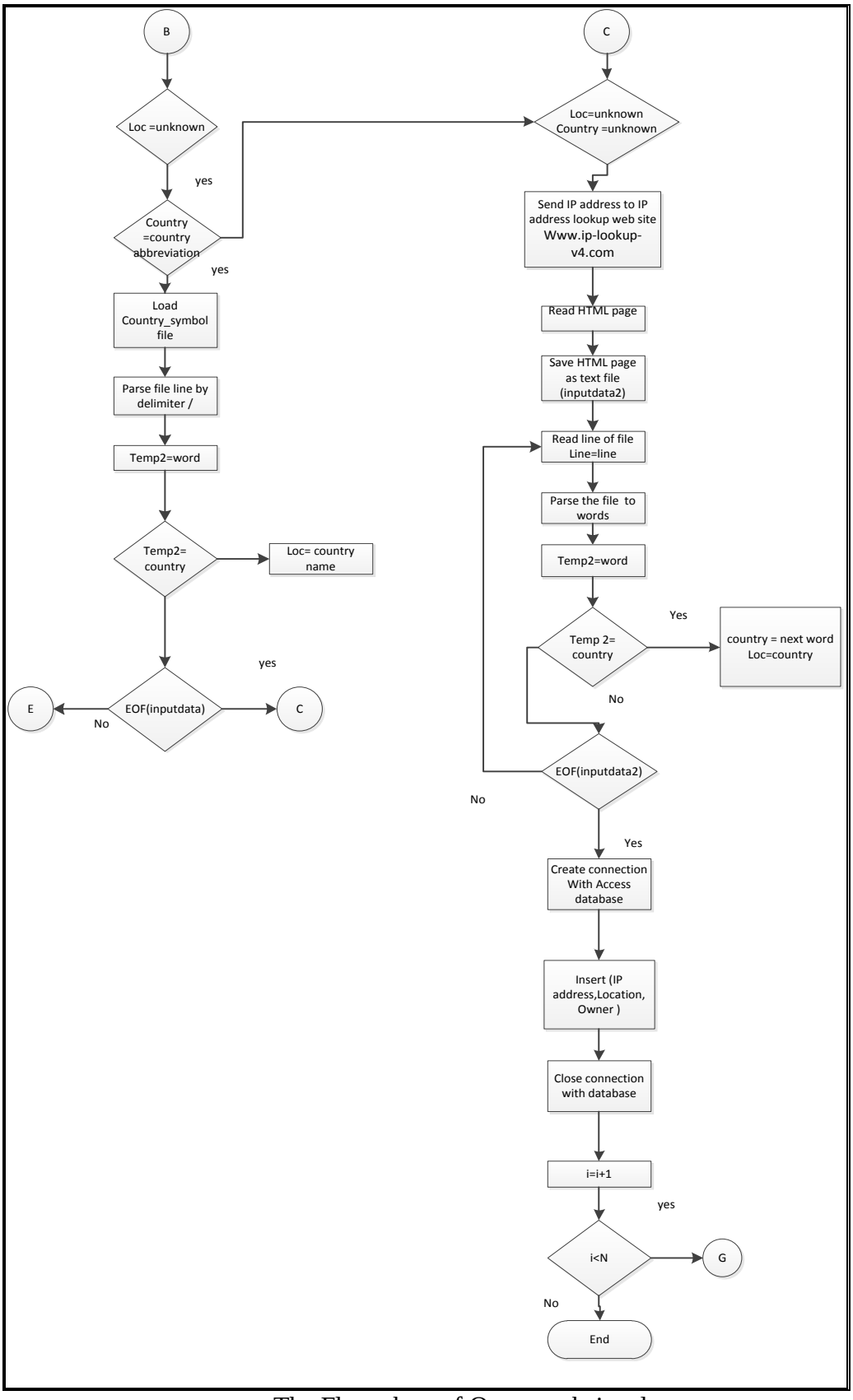
```
 if (( loc.equalsIgnoreCase("Unknown"))&& (country.equalsIgnoreCase("Unknown") ))
{
          try {
        URL url = new URL("http://ip-address-lookup-v4.com/lookup.php?ip="+IP[i]);
         Thread.sleep(15);
           BufferedReader reader = new BufferedReader(new
         InputStreamReader(url.openStream()));
         BufferedWriter writer = new BufferedWriter(new FileWriter("inputdata2.txt"));
          int count_line2=0;
          int limit2=0;
           String line1;
          while ((line1 = reader.readLine()) != null) {
            count_line2++;
           if((count_line2>70)&(limit2<=100)){
```

```
            //   System.out.println(line);
               writer.write(line1);
               writer.newLine();
               limit2++;
                }
               //end while
               }
            reader.close();
            writer.close();
            //end try statment
         } catch (MalformedURLException e) { e.printStackTrace();}
         catch (IOException e) {e.printStackTrace(); }
```
//   Get information for each IP from the second website and store them into file named (inputdata2)
```
         File file2 = new File("inputdata2.txt");
         try {
            Scanner scanner = new Scanner(file2);
            while (scanner.hasNextLine()) {
               String line2 = scanner.nextLine();
               Scanner sc2=new Scanner (line2);
               sc2.useDelimiter(" ");
              while (sc2.hasNext()){
               String temp= sc2.next();
               if (temp.equalsIgnoreCase("<tr><td><b>Country")){country=sc2.next();
            }
            }
         } catch (FileNotFoundException e) { e.printStackTrace();}
        country=country.replaceAll("Code</b></td><td>", "  ");
        country=country.replaceAll("</td></tr>", "  ");
        country=country.trim();
```
//Mapping country symbol to country name if location is unknown some IP has the country symbols so here I changed this

Symbol to location based on the file " country_symbols file"
```
              if (loc.equalsIgnoreCase("Unknown")){
                File symbol = new File("country_symbols.txt");
                try {
                Scanner scanner1 = new Scanner(symbol);
                while (scanner1.hasNextLine()) {
                String line = scanner1.nextLine();
                Scanner scanner2 =new Scanner (line);
                scanner2.useDelimiter("/");
                while (scanner2.hasNext()){
                String intemp= scanner2.next().trim();
                if (intemp.equalsIgnoreCase(country)){
                loc=scanner2.next().trim();              }
```

```
          }          }

        } catch (FileNotFoundException e) {
           e.printStackTrace();
        }  }      }

            // end of mapping function of country name

        if (loc.contains("'")){loc=loc.replace("'", " ");}
        if(Ogr_name.contains("'")){Ogr_name=Ogr_name.replace("'", " ");}


//   store the location and the owner for each IP in Access database each TCP file is
stored in separate table.

  try {

     Connection conn = DriverManager.getConnection(Url,username,password);
               Statement st = conn.createStatement();
                 int k=st.executeUpdate("insert into
STable_2100051810(IP_Address,Location,Country_Symbol,Organization_name)
values('"+ip+"','"+loc+"','"+country+"','"+Ogr_name+"')");

                 conn.close();
                 st.close();
                   }
               catch (SQLException ex) {
                Logger.getLogger(SendIP_StoreDB.class.getName()).log(Level.SEVERE,
null, ex);
               }                }
// end for loop
     owner_lock =true;
          }
// end main program
    }
   // end the class
}
```


Start
i=1
Read IP address
G
Org_Name= unknown loc =unknown Key_lock=true Owner_lock=true
Create connection With Acess Database
Search into data base
IP available
Yes
i=i+1 This IP repeated Key_lock=false
Close connection to access data base
No
Send IP to DNS Stuff Website Www.dnsstuff.com/tools
Read HTML page
Save HTML page as text file (inputdata.txt)
Read line of file Line=line
E
Parse the file to words
Temp =word
F
Temp=<A
Yes
Owner lock=true
Extemp= next line Owner_lock=false Case1=true
No
A
A
Temp = location
Yes
loc = next word
No
F
Temp = country
Yes
country = next word
No
Temp= descr
yes
No
Temp=OrgName
yes
D flag =true
Ogr_name= next word D flag =false
F
No
Temp= owner
yes
Temp=descr: descr:
yes
Ogr_name=unknown
yes
Case1=true
yes
Parse extemp Ogr_name= extemp.substring Case1=false
B



The Flow chart of Query website class

A5- Java Code to store the detailed information which retrieved from website into text file format.

```
import java.io.*;
import java.net.MalformedURLException;
import java.net.URL;

Public class detail_Result{

   public static void DSN_Data(String[] ADDIP){
  boolean lock_out;
   boolean first=true;
   File file= new File("outputdetail.txt");
         try {
              for (int i=1;i<ADDIP.length;i++ )
               {
             System.out.println("THE IP NUMBER " +i);lock_out=true;
              if (file.exists()& (first==true)){
// query website and read URL page for each IP address and write the IP
information.
    URL url = new
URL("http://www.dnsstuff.com/tools/whois/?tool_id=66&token=&toolhandler_redirect=
0&ip="+ADDIP[i]);
              BufferedReader reader = new BufferedReader(new
              InputStreamReader(url.openStream()));
               BufferedWriter writer = new BufferedWriter(new FileWriter(file, true));
               int count_line=0;
              String line;
              writer.write(str);
              writer.newLine();
              writer.newLine();
              writer.write(str1);writer.write("THE IP NUMBER :: ");
              writer.write(ADDIP[i]);
               while ((line = reader.readLine()) != null) {
               count_line++;
                if ( line.equalsIgnoreCase("</pre></TD></TR></TABLE>")){ lock_out
          =false;}
            if((count_line>142)&(lock_out==true)){  writer.write(line);writer.newLine();}
                 //end while
                }
                first=false;
          reader.close();
          writer.close();
             }
             else
              {
```

```
            URL url = new
URL("http://www.dnsstuff.com/tools/whois/?tool_id=66&token=&toolhandler_redirect=
0&ip="+ADDIP[i]);
            BufferedReader reader = new BufferedReader(new
InputStreamReader(url.openStream()));
            BufferedWriter writer = new BufferedWriter(new FileWriter(file, true));
            int count_line=0;
              String line;
              writer.newLine();
              writer.write(str1);writer.write("THE IP NUMBER:: ");
               writer.write(ADDIP[i]);
               writer.newLine();
              while ((line = reader.readLine()) != null) {
                count_line++;
               if ( line.equalsIgnoreCase("</pre></TD></TR></TABLE>")){ lock_out
=false;}
                if((count_line>142)&(lock_out==true)){  writer.write(line);writer.newLine();}
             //end while
           }
           reader.close();
           writer.close();

              }

           }      // end for loop

         //end try statement
       }
       catch (MalformedURLException e) {e.printStackTrace();}
       catch (IOException e) { e.printStackTrace();
       }
    }
}
```

VITA

AMAL ALI AM ALGEDIR

Candidate for the Degree of

Master of Science

Thesis: OUTBOUND NETWORK TRAFFIC MONITORING

Major Field: Electrical and Computer Engineering

Biographical:

Personal Data: Born in Valletta, Malta, On January 12, 1979.

Education: Completed the requirements for the Master of Science in Electrical and Computer Engineering at Oklahoma State University, Stillwater, Oklahoma in May, 2012.

Completed the requirements for the Bachelor of Science in Computer Engineering at Alfateh University, Tripoli, Libya in 2002.

Experience: worked as teaching assistance in at Alfateh university in Electrical engineering department from 2004-2008.

Professional Memberships:

Name: AMAL ALI .AM. ALGEDIR Date of Degree: May, 2012

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: OUTBOUND NETWORK TRAFFIC MONITORING

Pages in Study: 70 Candidate for the Degree of Master of Science

Major Field: Electrical and Computer Engineering.

Scope and Method of Study:

This study examined outbound computer network traffic by tracing the location of IP destination addresses and finding the owner of that IP address. In addition, changes in the pattern of outbound Oklahoma state university traffic over a 24 hour period were examined. The method used queries search engines that are available in Internet and used their databases to get the required information.

Findings and Conclusions:

The objective of this research is to begin the task of identifying the purpose of outbound traffic of a computer network. In this study, resources available on the Internet were used to find the probable location and the owner of observed destination IP addresses as the first step of this long term research goal. JAVA code was written which uses Internet search engines to get the required owner and location information. To test the code, headers of outbound Oklahoma State University traffic were collected using TCP Dump during four time intervals over a 24 hour period. By using the available information in Internet, the percentage of known IP Locations was approximately 99.7 % at all different times. The majority of IP destination address locations were in the United States. Traffic patterns were observed to change over time with most non-U.S. traffic headed for Asia and Europe.

ADVISER'S APPROVAL: George Scheets