AC 2012-3856: TEACHING NETWORK SECURITY THROUGH SIGNATURE ANALYSIS OF COMPUTER NETWORK ATTACKS

Dr. Te-Shun Chou, East Carolina University

Te-Shun Chou received his bachelor's degree in electronics engineering from Feng Chia University, Taiwan, R.O.C. in 1989, and the master's degree and doctoral degree both in electrical engineering from Florida International University, Miami, Fla., in 1992 and 2007, respectively. In 2008, he joined East Carolina University, Greenville, N.C., where he is currently an Assistant Professor with the Department of Technology Systems. His research interests include soft computing, wireless sensor network, and network security, especially intrusion detection and incident response.

Teaching Network Security Through Signature Analysis of Computer Network Attacks

Abstract

This paper presents an investigation of four categories of network attacks used in an intrusion detection and incident response graduate course; they are denial of service (*DoS*) attacks, *probe* attacks, user to root (*U2R*) attacks, and remote to local (*R2L*) attacks. In order to build an experimental network environment, virtualization technology is used. Two virtual machines are configured, one of which is used to launch attacks and the other acts as a victim host. A variety of network tools are installed for generation, collection and analysis of attack traffic traces. In each attack category, one real world attack is simulated; they are buffer overflow attack, TCP SYN scanning attack, backdoors attack, and guessing username and password attack. Finally, the attack traffic traces are analyzed and their attack signatures are extracted.

Keywords: Network security, intrusion detection system, virtualization technology

1. Introduction

Teaching network security is not an easy job. Due to the number of novel and sophisticated attacks that exploit system vulnerabilities within networks that are developed by hackers and crackers every day, it is impossible to teach the students every attack happens in the real world. However, it is possible to teach the students the ability of knowing how to analyze attackers' behavior and attack signatures. The best strategy to defend against attacks is to understand your enemy. Hence, in an intrusion detection and incident response graduate course we design a project that provides students with hands-on experience in terms of network configuration, real network attacks generation, collection and analysis, and intrusion detection system's implementation and evaluation. The complete procedure will not only provide a strong theoretical knowledge in the field of intrusion detection and incident response, but will also enhance each student's practical skills for advancement in the current and future network security job market. In this paper, we describe the attacks that we demonstrate in the designed project.

The concept of detecting abnormal behavior of computer users was first introduced by Anderson in 1980[1]. He published a paper, Computer Security Threat Monitoring and Surveillance, and defined that an attack was a specific formulation or execution of a plan to carry out a threat. He classified a threat as a deliberate unauthorized attempt to access information, manipulate information, or render a system unreliable or unusable. Since then, a variety of taxonomy schemes on grouping attacks into categories have been proposed. For example, in 1987 Denning[2] classified abnormal patterns of system usage into eight categories: attempted break-in, masquerading or successful break-in, penetration by legitimate user, leakage by legitimate user, inference by legitimate user, trojan horse, virus, and denial-of-service. In 1988, Smaha[3] divided intrusions into six main types: attempted break-ins, masquerade attacks, penetration of the security control system, leakage, denial of service, and malicious use. Howard[4] summarized the variations of taxonomy of attacks on the Internet from 1989 to 1995 in one of the chapters in his

PhD dissertation. In 1996, Sundaram[5] classified the intrusions into the categories of: attempted break-ins, masquerade, penetration of the security control system, leakage, denial of service, malicious use. Dekker[6] defined network security incident as an activity threat violated an explicit or implicit security policy and classified incidents into the probe, scan, account compromise, root compromise, packet sniffer, denial of service, exploitation of trust, malicious code, and Internet infrastructure attacks in 1997. In 1999, Lincoln Laboratory at MIT created the KDD99 data set, which is known as "DARPA Intrusion Detection Evaluation Data Set"[7]. The data set includes thirty-nine types of attacks that are classified into four main categories: *denial of service* (*DoS*) attacks, *probe* attacks, *user to root* (*U2R*) attacks, and *remote to local* (*R2L*) attacks.

The goal of this paper is to provide a detailed analysis of those four categories of attacks. The experiments simulate attacks that are conducted by attackers in the real world. To run the experiments, virtualization technique is used in building a network in a single physical host machine. Multiple virtual machines are created for attacks generation and collection. In each virtual machine, a variety of network tools and services are implemented. The virtual machines execute the applications just as a normal physical machine would. All of the experimental attacks are confined inside the virtual network. For each attack category, one attack is demonstrated in detailed steps in the project. Furthermore, each student is asked to simulate one attack for each category. The attack traffic traces are analyzed and their attack signatures are extracted. All of the analysis results are then used in the design of Snort rule of Snort intrusion detection system[8]. It further helps the students expanding their capabilities in building intrusion detection system as well as in evaluating the effectiveness of intrusion detection system design.

This paper is organized as follows: Section 2 presents four computer attack categories we investigate. Section 3 demonstrates the experimental methodology and followed by a discussion of the experiment results. Section 5 discusses the online survey statistics result. Finally, we present the conclusions and future work in the last section.

2. Four Categories of Computer Attacks

DoS Attacks

In the type of *DoS* attacks, attackers attempt to disrupt a host or network resource in order to make legitimate users not be able to access to the computer service. The victim machines can be web server, domain name system server, mail server, and so on. Known websites, such as Yahoo, eBay, Buy.com, CNN.com, E*TRADE and ZDNet have become victims of *DoS* attacks in the past[9].

DoS attacks come in a variety of forms and aim at a variety of services. Generally, they are categorized into three basic types: consumption of scarce, limited, or non-renewable resources, destruction or alteration of configuration information, and physical destruction or alteration of network components[10]. Among them, flooding is the most common way in which the attackers crumble the victim system with the use of an overwhelming number of packets, and therefore the services of legitimate users are blocked. For example, smurf attack can cause a target system crash by using the vulnerability of ICMP. The attacker sends a large number of ICMP "echo request" packets to the broadcast address and every packet has a spoofed source address of the intended target system. Any machine in the subnets will respond back to the target by sending ICMP “echo reply” packets. If the number of the packets is more than the target system can handle, the result is the spoofed system can no longer be able to service to the real ICMP requests. Another common way to fail a system is neptune attacks. It is a SYN flood attack that exists in TCP/IP implementation of a network. The attacker just simply rapidly sends out a large number of connection requests but never responds to any replies from the system. While the attacker continues to request new connections faster than the system can handle, the legitimate connection requests can never be accommodated. In the mean time, the system may run out of memory and even crash.

Probe Attacks

Probe attacks are conducted by the attackers use programs to automatically scan a large amount network IP addresses in order to find vulnerabilities that can be exploited. Once any vulnerability is found, the attackers can thus gain the access to the system and start to gather information without authorization. One of the most common *probe* attacks is called port scanning, which allows attackers to scan all ports on network hosts and discover which ones are available for connections. The popular scanning methods include TCP scanning, UDP scanning, SYN scanning, ACK scanning, FIN scanning, ICMP scanning, protocol scanning, and idle scanning. For example, portsweep attack discovers exploitable communication channels on remote hosts by systematically requesting connections to multiple TCP ports.

U2R Attacks

U2R attacks are the attacker pretends as a legitimate user of the system without authorization and then exploits the system’s vulnerabilities to get root access of that system. For example, the attacker may exploit a system’s vulnerabilities to gain root privileges and install a backdoor program onto a system for future access. The result may cause the system crash or make the system execute the attacker's program as if it is part of the system’s original programs. Another example is phf attack that exploits a security flaw of CGI script on a web server. Once the vulnerability is identified, the attacker can execute local commands on the attacked remote web server.

R2L Attacks

R2L attacks are unauthorized attackers through networks gain local access as users of local machines. The attacks can be launched from anywhere on the internet. Once the attacker has access to the information systems, they can then exploit the machine’s vulnerabilities and cause serious damages such as stealing important data or crashing the information systems. For example, ftp_write attack is that the attacker creates rhost file to make anonymous FTP directory writable and finally obtains local login to the system. Guess_passwd attack is that the attacker tries to gain access to a user’s account by repeatedly guessing the possible passwords. Any service that needs password to access possibly becomes an attacked target.

3. Experimental Methodology

The experimental environment is implemented in virtualization software VMware[11]. It allows us to run the experiments on multiple virtual machines within a single host system. Within VMware workstation, two virtual machines, Windows XP[12] and Linux CentOS[13], are preconfigured with IP addresses 192.168.17.146 and 192.168.17.144, respectively. The Linux CentOS system is used to launch attacks. The Windows XP system acts as a victim and records all the traffic generating from the attack host. In order to generate attacks and collect their traffic for analysis, a variety of network tools are installed and configured in both virtual machines. It includes Metasploit framework[14], Wireshark[15], Nmap[16], Netcat[17], Mozilla Firefox[18], Information Internet Services[19], and FTP server[20].

DoS Attacks

For *DoS* attack experiment, the Metasploit framework is used to issue an attack from the Linux CentOS host to Windows XP system. The Metasploit framework is an open source software for people to perform penetration testing, IDS signature development, and exploit research. Of its 320 exploits and 217 payloads, windows/vnc/ultravnc_client equipped with payload windows/shell_bind_tcp is chosen to exploit ultravnc_client buffer overflow vulnerability of Windows XP machine.

This is a client buffer overflow attack. The attacker exploits the vulnerability of a system that does not correctly perform a boundary check of user's input data before copying it to a fixed length memory buffer. Once the vulnerability is found, the attacker can supply excess data into the insufficiently sized memory buffers and therefore possibly corrupt the data and thus make the service crash. Furthermore, the attacker can add executable data into the stream and remotely activate it to gain unauthorized access when the buffer overflows. Example can be seen such as installing a backdoor program on the vulnerable system for future use.

Probe Attacks

Probe attacks are attacks to explore open vulnerabilities or weaknesses of a network. They aim to gather information on systems within a network in order to lead to access to targeted computers in the future. Among various types of *probe* attacks, network port scanning is a common way to find out what resources are available on your network. In this experiment, a free security scanner Nmap is used in Linux CentOS host for network exploration of target Windows XP. It divides ports into six states: open, closed, filtered, unfiltered, open|filtered, or closed|filtered. These states give attackers an idea of services' statuses in the target computer system.

A variety of scans are provided by Nmap, which include TCP connect, SYN stealth, FIN, NULL, Xmas Tree, Ping, UDP, IP Protocol, Idle, Ack, Window, RPC, List, Version Detection, Timing and Hiding Scans[16]. In this experiment, the most common used port scan, TCP SYN scanning, is applied. If the connection to a port is successful, the port is listed as open, otherwise it is said to be closed. The scan result provides the basic port information of a system and the attacker can then look to open ports and vulnerabilities for further exploration.

U2R Attacks

In a *U2R* attack, the attacker normally starts with a remote attack to gain access to a vulnerable system. Once the attacker has access at some level as a legitimate user, he/she will gain a higher level privilege such as administrator or root. This is often done through installing a backdoor program on the compromised system. By using this technique, the attacker can bypass the normal authentication process and easily return to the system for desired activities. Basically backdoors are classified into three basic categories: active, passive and attack-based backdoors[21]. Active backdoors are actively monitored by hackers and can be used anytime whenever they wish to access to the compromised system from the remote systems. Passive backdoors can be triggered by time or events and therefore the attackers have to wait for them to happen. They are similar to active ones that they can establish access into the compromised network for sending data out and receiving acknowledgements and/or commands from the remote systems. Attack-based backdoors could be classified as the “unknown backdoors”. They are generally caused from the attackers use buffer overflow technique to exploit vulnerabilities of poorly-written programs and therefore gain administrator or root level access to the compromised system.

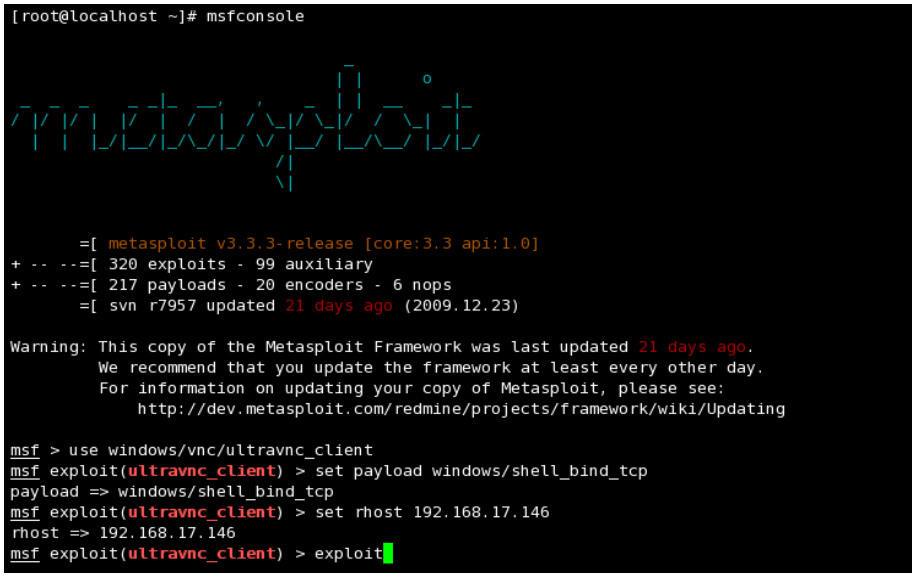
In this experiment a *U2R* attack is conducted by installing an active backdoor on target Windows XP system and connecting the attack Linux CentOS host to the victim’s http port. Internet Information Services (IIS) is installed in the victim’s machine and the default port is 80. After the backdoor is open on port 80 of the target system, the attacker in the remote host can gain the access to the command shell and execute commands such as cd, dir, and mkdir on the victim machine. The entire process is done by creating a Netcat backdoor listener in Windows XP and running Netcat as client mode in Linux CentOS.

R2L Attacks

For protecting network services, systems in network always use authentication technique to prove users’ identities by providing their usernames and passwords. In general, people do not create strong passwords so that the attackers have chances to apply brute force attack or dictionary attack technique to break those bad passwords. The objective of *R2L* attack experiment is to simulate guessing username/password attack. It starts with running FTP server on the victim Windows XP host, and then the server is connected to the attack Linux CentOS host using a web browser. Once the communication channel is established, the guessing username/password attacks are simulated by entering incorrect information on the client machine. The entire course of attacks is recorded on the victim machine with Wireshark and the packet capture file is saved for future analysis.

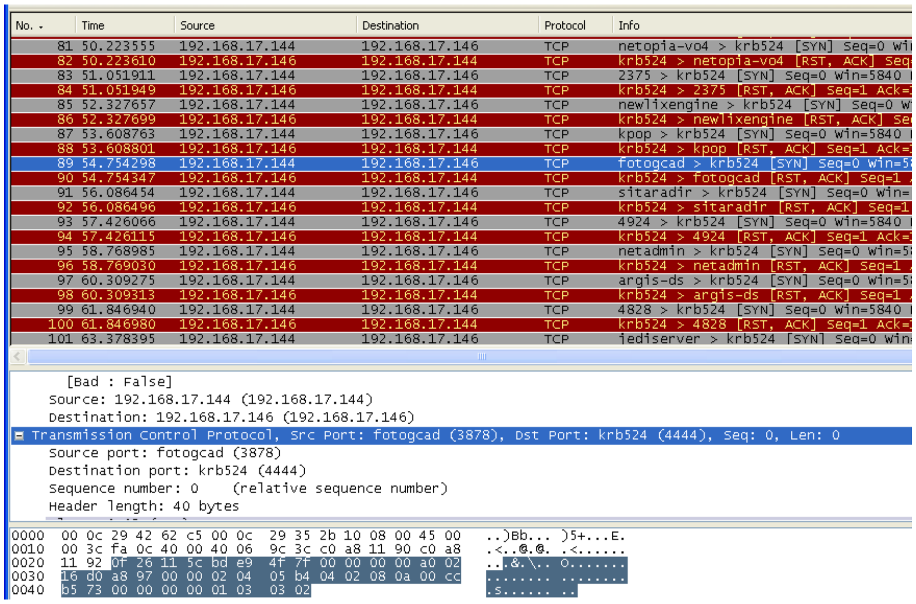
4. Experimental Results

Figure 1 shows the commands used in Metasploit Framework to start a *DoS* attack on Windows victim machine. Figure 2 shows part of the packets captured by Wireshark during the attack period of time. After examining the packets, it indicates that *DoS* attack uses TCP port 4444 (krb524 service) as its targeting channel. During the attack, the attacker keeps sending SYN packets with randomly port number to the same TCP port of target host. In a one minute period, the attacker sends nearly 120 packets to the same port of target host. Whenever the target machine receives a packet from the attack host, it replies with ACK flag as well as RST flag indicating the port is closed. However, the attack host just ignores those responses and keeps sending SYN packets out. These packets alternated back and forth and no actual TCP connection is established for further communications. If the service port is open, this SYN flood attack can keep the target busy to reply and therefore makes the target unable to respond to other legitimate requests until the attack ends.



```
[root@localhost ~]# msfconsole

                                  _
                                 | |      o
 _  _  _    _ _|_  __,   ,    _  | |  __    _|_
/ |/ |/ |  |/  |  /  |  / \_|/ \_|/  /  \_|  |
  |  |  |_/|__/|_/\_/|_/ \/ |__/ |__/\__/ |_/|_/
                           /|
                           \|


       =[ metasploit v3.3.3-release [core:3.3 api:1.0]
+ -- --=[ 320 exploits - 99 auxiliary
+ -- --=[ 217 payloads - 20 encoders - 6 nops
       =[ svn r7957 updated 21 days ago (2009.12.23)

Warning: This copy of the Metasploit Framework was last updated 21 days ago.
         We recommend that you update the framework at least every other day.
         For information on updating your copy of Metasploit, please see:
             http://dev.metasploit.com/redmine/projects/framework/wiki/Updating

msf > use windows/vnc/ultravnc_client
msf exploit(ultravnc_client) > set payload windows/shell_bind_tcp
payload => windows/shell_bind_tcp
msf exploit(ultravnc_client) > set rhost 192.168.17.146
rhost => 192.168.17.146
msf exploit(ultravnc_client) > exploit
```

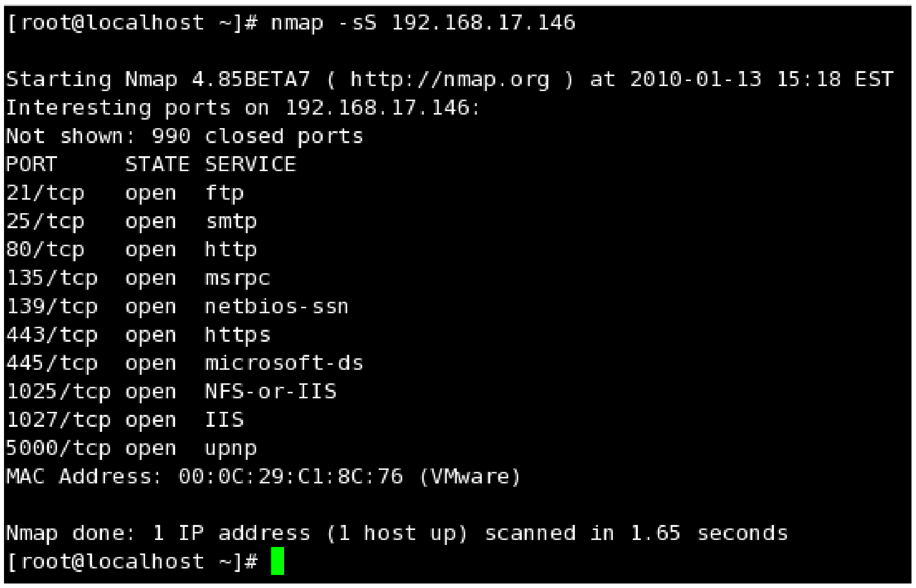
Figure 1. Metasploit Framework



```
No. -   Time       Source           Destination      Protocol  Info
     81 50.223555  192.168.17.144   192.168.17.146   TCP       netopia-vo4 > krb524 [SYN] Seq=0 Wi
     82 50.223610  192.168.17.146   192.168.17.144   TCP       krb524 > netopia-vo4 [RST, ACK] Seq
     83 51.051911  192.168.17.144   192.168.17.146   TCP       2375 > krb524 [SYN] Seq=0 Win=5840
     84 51.051949  192.168.17.146   192.168.17.144   TCP       krb524 > 2375 [RST, ACK] Seq=1 Ack=
     85 52.327657  192.168.17.144   192.168.17.146   TCP       newlixengine > krb524 [SYN] Seq=0 W
     86 52.327699  192.168.17.146   192.168.17.144   TCP       krb524 > newlixengine [RST, ACK] Se
     87 53.608763  192.168.17.144   192.168.17.146   TCP       kpop > krb524 [SYN] Seq=0 Win=5840
     88 53.608801  192.168.17.146   192.168.17.144   TCP       krb524 > kpop [RST, ACK] Seq=1 Ack=
     89 54.754298  192.168.17.144   192.168.17.146   TCP       fotogcad > krb524 [SYN] Seq=0 Win=5
     90 54.754347  192.168.17.146   192.168.17.144   TCP       krb524 > fotogcad [RST, ACK] Seq=1
     91 56.086454  192.168.17.144   192.168.17.146   TCP       sitaradir > krb524 [SYN] Seq=0 Win=
     92 56.086496  192.168.17.146   192.168.17.144   TCP       krb524 > sitaradir [RST, ACK] Seq=1
     93 57.426066  192.168.17.144   192.168.17.146   TCP       4924 > krb524 [SYN] Seq=0 Win=5840
     94 57.426115  192.168.17.146   192.168.17.144   TCP       krb524 > 4924 [RST, ACK] Seq=1 Ack=
     95 58.768985  192.168.17.144   192.168.17.146   TCP       netadmin > krb524 [SYN] Seq=0 Win=5
     96 58.769030  192.168.17.146   192.168.17.144   TCP       krb524 > netadmin [RST, ACK] Seq=1
     97 60.309275  192.168.17.144   192.168.17.146   TCP       argis-ds > krb524 [SYN] Seq=0 Win=5
     98 60.309313  192.168.17.146   192.168.17.144   TCP       krb524 > argis-ds [RST, ACK] Seq=1
     99 61.846940  192.168.17.144   192.168.17.146   TCP       4828 > krb524 [SYN] Seq=0 Win=5840
    100 61.846980  192.168.17.146   192.168.17.144   TCP       krb524 > 4828 [RST, ACK] Seq=1 Ack=
    101 63.378395  192.168.17.144   192.168.17.146   TCP       jediserver > krb524 [SYN] Seq=0 Win

      [Bad : False]
    Source: 192.168.17.144 (192.168.17.144)
    Destination: 192.168.17.146 (192.168.17.146)
⊟ Transmission Control Protocol, Src Port: fotogcad (3878), Dst Port: krb524 (4444), Seq: 0, Len: 0
    Source port: fotogcad (3878)
    Destination port: krb524 (4444)
    Sequence number: 0    (relative sequence number)
    Header length: 40 bytes

0000  00 0c 29 42 62 c5 00 0c  29 35 2b 10 08 00 45 00   ..)Bb... )5+...E.
0010  00 3c fa 0c 40 00 40 06  9c 3c c0 a8 11 90 c0 a8   .<..@.@. .<......
0020  11 92 0f 26 11 5c bd e9  4f 7f 00 00 00 00 a0 02   ...&.\.. O.......
0030  16 d0 a8 97 00 00 02 04  05 b4 04 02 08 0a 00 cc   ........ ........
0040  b5 73 00 00 00 00 01 03  03 02                     .s...... ..
```

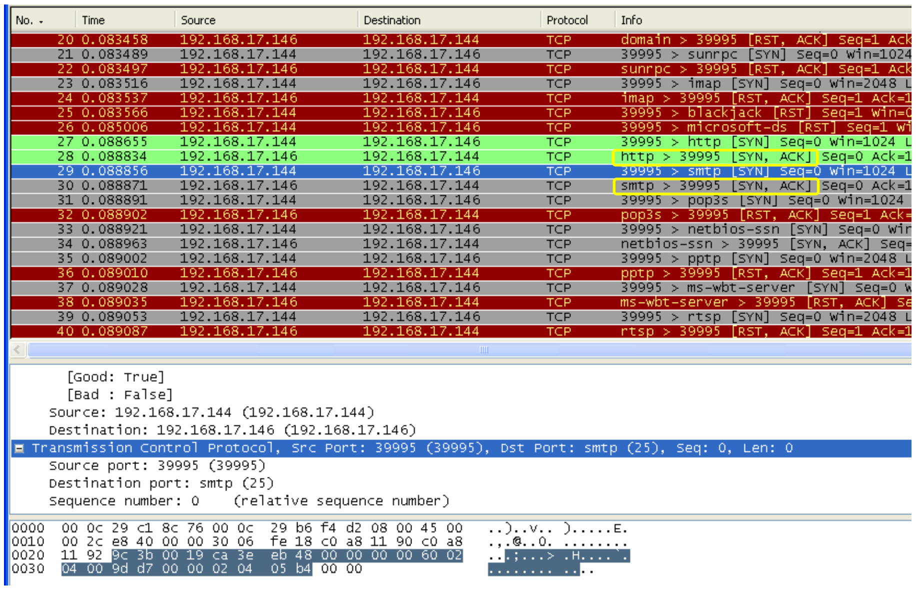
Figure 2. *DoS* Attack

Next we analyze *probe* attack which uses TCP SYN scan to check the port status of the target Windows system. Figure 3 shows the scanning result of open ports on the target machine. The attacker uses half-open scan technique to determine which ports are open and which are closed on its target. SYN packets are sent to the target's port one after another, but a full TCP connection is never established. If a SYN/ACK packet is replied by the target, it represents the port is open and listening. On the other hand the port is closed if a RST/ACK packet is replied. Figure 4 indicates part of the packets during the scanning process that are recorded by Wireshark. It shows most ports of the target system are closed that reply with RST/ACK packets. The two yellow boxes show http and smtp ports are open and SYN/ACK packets are responded. Also, it is noticeable that the attacker uses a static source port, 39995, to send all SYN packets to the target system.



```
[root@localhost ~]# nmap -sS 192.168.17.146

Starting Nmap 4.85BETA7 ( http://nmap.org ) at 2010-01-13 15:18 EST
Interesting ports on 192.168.17.146:
Not shown: 990 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
25/tcp    open  smtp
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
1025/tcp open  NFS-or-IIS
1027/tcp open  IIS
5000/tcp open  upnp
MAC Address: 00:0C:29:C1:8C:76 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.65 seconds
[root@localhost ~]#
```

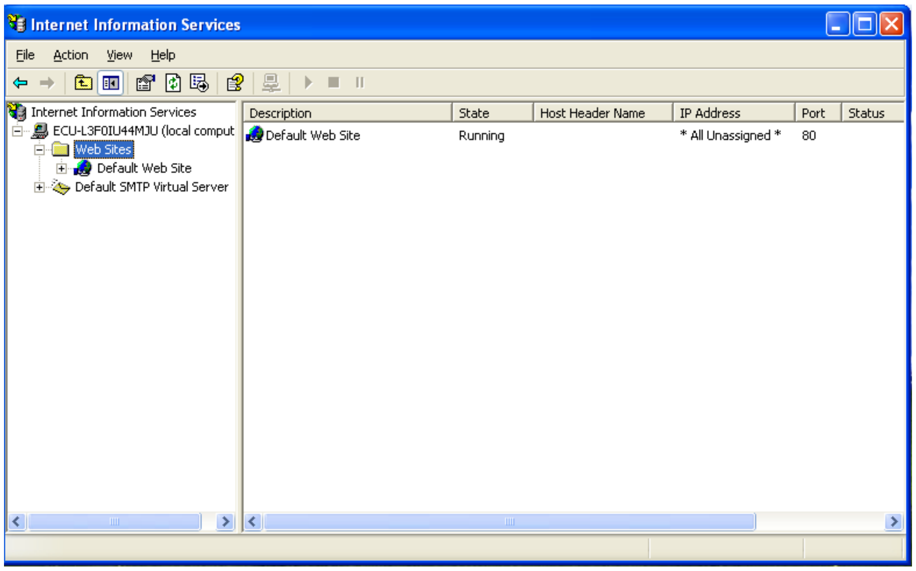
Figure 3. TCP SYN scanning



```
No. -  Time      Source          Destination     Protocol  Info
   20 0.083458   192.168.17.146  192.168.17.144  TCP       domain > 39995 [RST, ACK] Seq=1 Ack
   21 0.083489   192.168.17.144  192.168.17.146  TCP       39995 > sunrpc [SYN] Seq=0 Win=1024
   22 0.083497   192.168.17.146  192.168.17.144  TCP       sunrpc > 39995 [RST, ACK] Seq=1 Ack
   23 0.083516   192.168.17.144  192.168.17.146  TCP       39995 > imap [SYN] Seq=0 Win=2048 L
   24 0.083537   192.168.17.146  192.168.17.144  TCP       imap > 39995 [RST, ACK] Seq=1 Ack=1
   25 0.083566   192.168.17.144  192.168.17.146  TCP       39995 > blackjack [RST] Seq=1 Win=0
   26 0.085006   192.168.17.144  192.168.17.146  TCP       39995 > microsoft-ds [RST] Seq=1 Wi
   27 0.088655   192.168.17.144  192.168.17.146  TCP       39995 > http [SYN] Seq=0 Win=1024 L
   28 0.088834   192.168.17.146  192.168.17.144  TCP       http > 39995 [SYN, ACK] Seq=0 Ack=1
   29 0.088856   192.168.17.144  192.168.17.146  TCP       39995 > smtp [SYN] Seq=0 Win=1024 L
   30 0.088871   192.168.17.146  192.168.17.144  TCP       smtp > 39995 [SYN, ACK] Seq=0 Ack=1
   31 0.088891   192.168.17.144  192.168.17.146  TCP       39995 > pop3s [SYN] Seq=0 Win=1024
   32 0.088902   192.168.17.146  192.168.17.144  TCP       pop3s > 39995 [RST, ACK] Seq=1 Ack=
   33 0.088921   192.168.17.144  192.168.17.146  TCP       39995 > netbios-ssn [SYN] Seq=0 Win
   34 0.088963   192.168.17.146  192.168.17.144  TCP       netbios-ssn > 39995 [SYN, ACK] Seq=
   35 0.089002   192.168.17.144  192.168.17.146  TCP       39995 > pptp [SYN] Seq=0 Win=2048 L
   36 0.089010   192.168.17.146  192.168.17.144  TCP       pptp > 39995 [RST, ACK] Seq=1 Ack=1
   37 0.089028   192.168.17.144  192.168.17.146  TCP       39995 > ms-wbt-server [SYN] Seq=0 W
   38 0.089035   192.168.17.146  192.168.17.144  TCP       ms-wbt-server > 39995 [RST, ACK] Se
   39 0.089053   192.168.17.144  192.168.17.146  TCP       39995 > rtsp [SYN] Seq=0 Win=2048 L
   40 0.089087   192.168.17.146  192.168.17.144  TCP       rtsp > 39995 [RST, ACK] Seq=1 Ack=1

      [Good: True]
      [Bad : False]
    Source: 192.168.17.144 (192.168.17.144)
    Destination: 192.168.17.146 (192.168.17.146)
⊟ Transmission Control Protocol, Src Port: 39995 (39995), Dst Port: smtp (25), Seq: 0, Len: 0
    Source port: 39995 (39995)
    Destination port: smtp (25)
    Sequence number: 0    (relative sequence number)

0000  00 0c 29 c1 8c 76 00 0c  29 b6 f4 d2 08 00 45 00   ..)..v.. ).....E.
0010  00 2c e8 40 00 00 30 06  fe 18 c0 a8 11 90 c0 a8   .,.@..0. ........
0020  11 92 9c 3b 00 19 ca 3e  eb 48 00 00 00 00 60 02   ...;...> .H....`.
0030  04 00 9d d7 00 00 02 04  05 b4 00 00               ........ ....
```

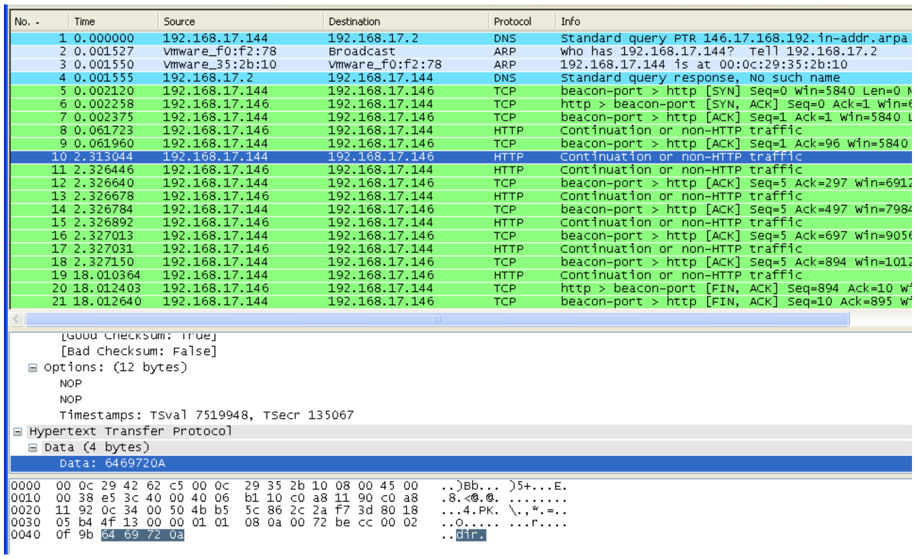
Figure 4. *Probe* Attack

Figure 5 shows the status and port information of IIS running on the target machine. Figure 6 shows part of the packets of *U2R* attack. In the beginning of the communication, a three-way handshake, a SYN, a SYN/ACK, and followed by an ACK, establishes a connection on http port between the destination and the source. Then, packet 10 shows that the attacker issues a "dir" command in remote machine. Packets 11, 13, 15, and 17 show the target sends the queries back to the attacker. The result indicates the attacker successfully bypasses the normal authentication process and obtains access to the target machine undetected.


Internet Information Services
File
Action
View
Help
Internet Information Services
ECU-L3F0IU44MJU (local comput
Web Sites
Default Web Site
Default SMTP Virtual Server

Description	State	Host Header Name	IP Address	Port	Status
Default Web Site	Running		* All Unassigned *	80	

Figure 5. Internet Information Services

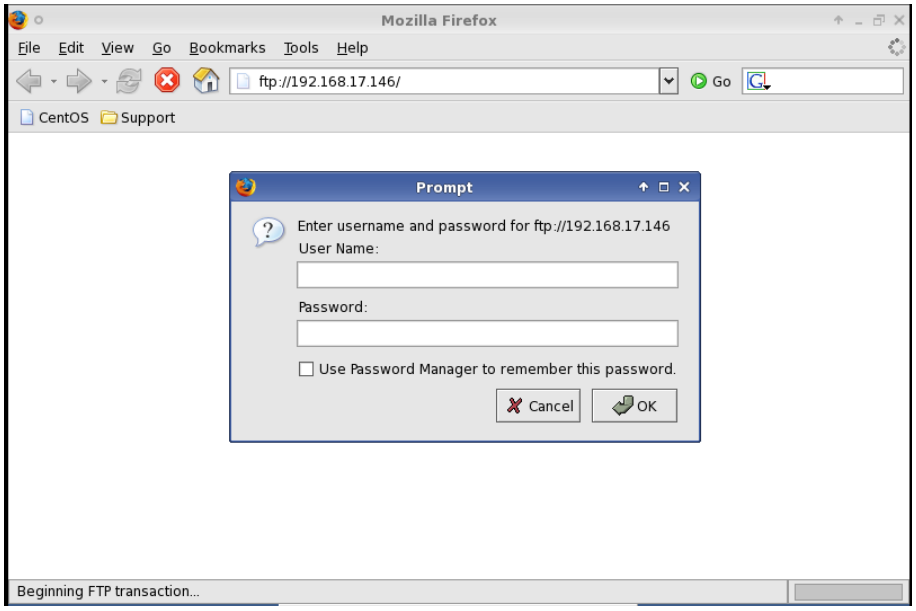


```
No. -  Time       Source             Destination        Protocol  Info
     1 0.000000   192.168.17.144     192.168.17.2       DNS       Standard query PTR 146.17.168.192.in-addr.arpa
     2 0.001527   Vmware_f0:f2:78    Broadcast          ARP       Who has 192.168.17.144?  Tell 192.168.17.2
     3 0.001550   Vmware_35:2b:10    Vmware_f0:f2:78    ARP       192.168.17.144 is at 00:0c:29:35:2b:10
     4 0.001555   192.168.17.2       192.168.17.144     DNS       Standard query response, No such name
     5 0.002120   192.168.17.144     192.168.17.146     TCP       beacon-port > http [SYN] Seq=0 Win=5840 Len=0 M
     6 0.002258   192.168.17.146     192.168.17.144     TCP       http > beacon-port [SYN, ACK] Seq=0 Ack=1 Win=
     7 0.002375   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=1 Ack=1 Win=5840 L
     8 0.061723   192.168.17.146     192.168.17.144     HTTP      Continuation or non-HTTP traffic
     9 0.061960   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=1 Ack=96 Win=5840
    10 2.313044   192.168.17.144     192.168.17.146     HTTP      Continuation or non-HTTP traffic
    11 2.326446   192.168.17.146     192.168.17.144     HTTP      Continuation or non-HTTP traffic
    12 2.326640   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=5 Ack=297 Win=691
    13 2.326678   192.168.17.146     192.168.17.144     HTTP      Continuation or non-HTTP traffic
    14 2.326784   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=5 Ack=497 Win=798
    15 2.326892   192.168.17.146     192.168.17.144     HTTP      Continuation or non-HTTP traffic
    16 2.327013   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=5 Ack=697 Win=905
    17 2.327031   192.168.17.146     192.168.17.144     HTTP      Continuation or non-HTTP traffic
    18 2.327150   192.168.17.144     192.168.17.146     TCP       beacon-port > http [ACK] Seq=5 Ack=894 Win=101
    19 18.010364  192.168.17.144     192.168.17.146     HTTP      Continuation or non-HTTP traffic
    20 18.012403  192.168.17.146     192.168.17.144     TCP       http > beacon-port [FIN, ACK] Seq=894 Ack=10 W
    21 18.012640  192.168.17.144     192.168.17.146     TCP       beacon-port > http [FIN, ACK] Seq=10 Ack=895 W
```

```
        [Good Checksum: True]
        [Bad Checksum: False]
  ⊟ Options: (12 bytes)
        NOP
        NOP
        Timestamps: TSval 7519948, TSecr 135067
⊟ Hypertext Transfer Protocol
  ⊟ Data (4 bytes)
        Data: 6469720A

0000  00 0c 29 42 62 c5 00 0c  29 35 2b 10 08 00 45 00   ..)Bb... )5+...E.
0010  00 38 e5 3c 40 00 40 06  b1 10 c0 a8 11 90 c0 a8   .8.<@.@. ........
0020  11 92 0c 34 00 50 4b b5  5c 86 2c 2a f7 3d 80 18   ...4.PK. \.,*.=..
0030  05 b4 4f 13 00 00 01 01  08 0a 00 72 be cc 00 02   ..O..... ...r....
0040  0f 9b 64 69 72 0a                                  ..dir.
```

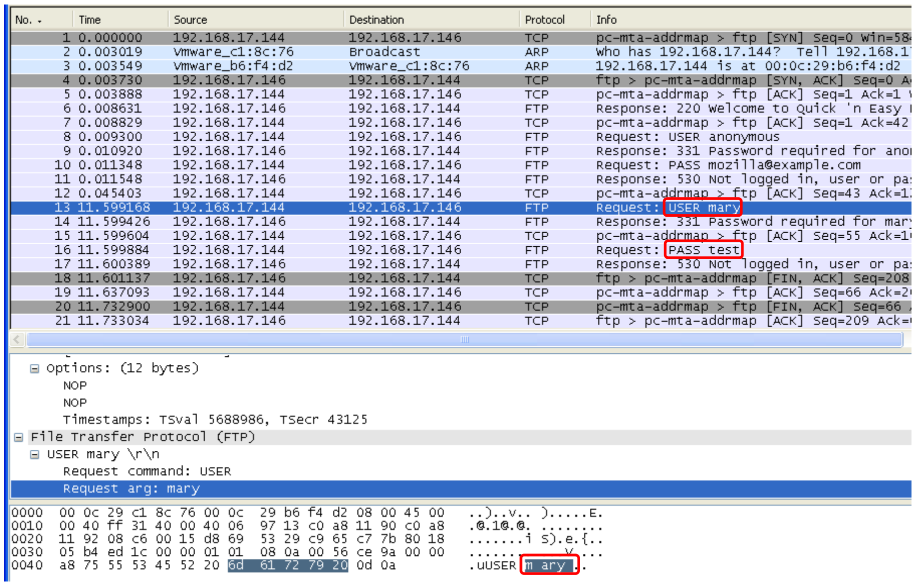
Figure 6. *U2R* Attack

Finally, a *R2L* attack is simulated by performing guessing username/password attack in a ftp server that shown on Figure 7. The attack is analyzed in Figure 8 and the following characteristics in the captured packets are found. First of all, a three-way handshake is communicated between attack host and target machine's port 21. A welcome message is then sent from ftp server shown on packet 6 and followed by a request of user's login username and password in packet 13. Whenever the user inputs an incorrect username or password, ftp server stops its service and the user needs to reconnect to the server again. In the simulation, username "mary" and password "test" are used. During the authentication process between client and server, the result indicates the username and password information is visible with plaintext in the data payload which is shown on the red box of the figure.



Enter username and password for ftp://192.168.17.146
User Name:
Password:
Use Password Manager to remember this password.
Beginning FTP transaction...

Figure 7. Ftp Login Window



```
No. . Time       Source             Destination        Protocol  Info
   1 0.000000   192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [SYN] Seq=0 Win=58
   2 0.003019   Vmware_c1:8c:76    Broadcast          ARP       Who has 192.168.17.144?  Tell 192.168.1
   3 0.003549   Vmware_b6:f4:d2    Vmware_c1:8c:76    ARP       192.168.17.144 is at 00:0c:29:b6:f4:d2
   4 0.003730   192.168.17.146     192.168.17.144     TCP       ftp > pc-mta-addrmap [SYN, ACK] Seq=0 A
   5 0.003888   192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [ACK] Seq=1 Ack=1
   6 0.008631   192.168.17.146     192.168.17.144     FTP       Response: 220 Welcome to Quick 'n Easy
   7 0.008829   192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [ACK] Seq=1 Ack=42
   8 0.009300   192.168.17.144     192.168.17.146     FTP       Request: USER anonymous
   9 0.010920   192.168.17.146     192.168.17.144     FTP       Response: 331 Password required for anon
  10 0.011348   192.168.17.144     192.168.17.146     FTP       Request: PASS mozilla@example.com
  11 0.011548   192.168.17.146     192.168.17.144     FTP       Response: 530 Not logged in, user or pa
  12 0.045403   192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [ACK] Seq=43 Ack=1
  13 11.599168  192.168.17.144     192.168.17.146     FTP       Request: USER mary
  14 11.599426  192.168.17.146     192.168.17.144     FTP       Response: 331 Password required for mar
  15 11.599604  192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [ACK] Seq=55 Ack=1
  16 11.599884  192.168.17.144     192.168.17.146     FTP       Request: PASS test
  17 11.600389  192.168.17.146     192.168.17.144     FTP       Response: 530 Not logged in, user or pa
  18 11.601137  192.168.17.146     192.168.17.144     TCP       ftp > pc-mta-addrmap [FIN, ACK] Seq=208
  19 11.637093  192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [ACK] Seq=66 Ack=2
  20 11.732900  192.168.17.144     192.168.17.146     TCP       pc-mta-addrmap > ftp [FIN, ACK] Seq=66
  21 11.733034  192.168.17.146     192.168.17.144     TCP       ftp > pc-mta-addrmap [ACK] Seq=209 Ack=

  ⊟ Options: (12 bytes)
      NOP
      NOP
      Timestamps: TSval 5688986, TSecr 43125
⊟ File Transfer Protocol (FTP)
  ⊟ USER mary \r\n
      Request command: USER
      Request arg: mary

0000  00 0c 29 c1 8c 76 00 0c  29 b6 f4 d2 08 00 45 00   ..)..v.. ).....E.
0010  00 40 ff 31 40 00 40 06  97 13 c0 a8 11 90 c0 a8   .@.1@.@. ........
0020  11 92 08 c6 00 15 d8 69  53 29 c9 65 c7 7b 80 18   .......i S).e.{..
0030  05 b4 ed 1c 00 00 01 01  08 0a 00 56 ce 9a 00 00   ........ ...V....
0040  a8 75 55 53 45 52 20 6d  61 72 79 20 0d 0a         .uUSER m ary ..
```

Figure 8. *R2L* Attack

5. Evaluation

The project designed for the intrusion detection and incident response graduate course elaborates a process of intrusion detection and prevention system (IDPS) development. The project is divided into seven phases, they are:

Phase 1. Creation of an intrusion detection experimental environment
- To help students recognize the procedure of virtual network installation and configuration

Phase 2. Attacks recording
- To help students understand real world network attacks and computer systems' vulnerabilities

Phase 3. Analysis of attack signatures
- To help students investigate attack behavior from network traffic

Phase 4. Generation of intrusion detection rules
- To help students construct effective intrusion detection rules

Phase 5. Collection of normal traffic
- To help students assemble an intrusion detection experimental dataset

Phase 6. IDPS performance evaluation
- To help students perform proper evaluation of IDPS

Phase 7. The final integration
- To combine everything done in previous phases

In the end of the semester, a survey with eight individual questions was posted online for students' access. The objective of the survey is to evaluate the project's effectiveness in order to improve the project manual for future use. A five-level Likert scale was used. Available responses were: strongly disagree, disagree, neutral, agree, and strongly agree. In order to investigate attitudes of the respondents toward each question, we coded the responses accordingly: strongly disagree = 1, disagree = 2, neutral = 3, agree = 4, and strongly agree = 5.

In the survey, three questions regarding phases 2 and 3 were designed. Table 1 shows the questions and Table 2 shows the descriptive statistics result. Totally thirteen questionnaires are successfully collected at the end of the course.

Table 1: Three questions regarding attack recording and analysis of attack signatures

No.	Question
1	I know how to generate computer attacks to attack vulnerable victims.
2	By inspecting network traffic, I can find possible attack activities with the use of the packet analyzer.
3	After completing the report, I have a better understanding of the signatures of difference attacks.

Table 2: Survey statistics result

Question	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)	Mean	Standard Deviation
1			2 (15.38%)	7 (53.85%)	4 (30.77%)	4.15	0.69
2			1 (7.69%)	9 (69.23%)	3 (23.08%)	4.15	0.55
3				5 (38.46%)	8 (61.54%)	4.62	0.51

With Likert scale data, the most frequent response is the best way to illustrate the analysis result. Over 80% of students expressed that they knew how to apply security exploitation tools to exploit computer system vulnerabilities and analyzed the attack signatures using a packet analyzer. Students showed that the process helped them understanding the signatures of difference attacks. "*After completing this project, I do now have quite a bit better understanding of how to do this task.*", "*I can certainly generate attacks using the tools provided and the tools found. I must say it has peaked my interest and I will maintain the virtual environment for testing of new tools and attacks in the future.*", and "*It was very nice to understand the various types of attacks (DoS, Probe, U2R, & R2L), so then creating and understanding those signatures was very helpful in my learning process.*"

Overall the average of the three questions is over 4 points, which shows the students had very positive attitude toward the questions. In addition, we asked students to provide one example where they have added to their knowledge from this project. Some of responses related to the three questions shown on Table 1 were: "*I learned a great deal about how to use a packet analyzer to better understand network traffic*", "*I learned a lot during the attack analysis phase.*", "*I really liked how we were shown how to use Metasploit. Overall, I think I have a much better hands-on mentality of intrusion detection.*", "*Metasploit, Metasploit, Metasploit. I was intrigued by this program from it's introduction in the course in DoS Attack 1.*", and "*I tried out Metasploit to test my own system's vulnerabilities but I was never able to completely gain access over a machine before this class – seeing is believing. I was able to see this happen first-hand during this class.*"

6. Conclusions and Future Work

This paper presents research of four categories of network attacks used in a graduate course project in intrusion detection and incident response. In each category, one real world attack is simulated in a virtualization network. The attack traffic is then collected and analyzed in an attempt to detect any attack signatures. The project helps students develop skills in generating, collecting and analyzing malicious network activities in real scenarios. According to the discovery, a set of rules can then be designed for the use of misuse intrusion detection systems. Also, all of the research outcomes can be verified within the virtualization network. The project will also help students expand their capabilities in building real intrusion detection systems and in evaluating the effectiveness of systems' design. In the future, more attacks will be included and analyzed, therefore enabling the students to have a broader understanding of the different kinds of network attacks' behavior.

Bibliography

1. J. P. Anderson, *Computer Security Threat Monitoring and Surveillance*, Technical Report, James P. Anderson Co., Fort Washington, PA, April 1980.
2. E. Denning, “An Intrusion-Detection Model,” IEEE Transactions on Software Engineering, Volume 13, Number 2, pp. 222-232, 1987.
3. S. E. Smaha, “Haystack: An Intrusion Detection System,” Fourth Aerospace Computer Security Applications Conference, pp. 37-44, Austin, Texas, 1988.
4. J. D. Howard, An Analysis of Security Incidents on the Internet 1989 – 1995, Dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1997.
5. A. Sundaram, “An Introduction to Intrusion Detection,” Crossroads: The ACM Student Magazine, 2, 4, 1996.
6. M. Dekker, “Security of the Internet,” The Froehlich/Kent Encyclopedia of Telecommunications, Volume 15, pp. 231-255, New York, 1997.
7. DARPA Intrusion Detection Evaluation, MIT Lincoln Laboratory. http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/docs/attackDB.html (Last browsed in February 2012)
8. Snort. http://www.snort.org/ (Last browsed in February 2012)
9. Piskozub, “Denial of service and distributed denial of service attacks,” TCSET, pp. 303-304, February 2002.
10. CERT Coordination Center, Denial of Service. http://www.packetstormsecurity.org/distributed/denial_of_service.htm (Last browsed in February 2012)
11. VMware. http://www.vmware.com/ (Last December in December 2011)
12. Windows XP. http://www.microsoft.com/windows/windows-xp/default.aspx (Last browsed in February 2012)
13. Linux CentOS. http://www.centos.org/ (Last browsed in February 2012)
14. Metasploit Framework. http://www.metasploit.com (Last browsed in February 2012)
15. Wireshark. http://www.wireshark.org (Last browsed in February 2012)
16. Nmap. http://www.nmap.org (Last browsed in February 2012)
17. Netcat. http://netcat.sourceforge.net (Last browsed in February 2012)
18. Mozilla Firefox web browser. http://www.mozilla.com/firefox (Last browsed in February 2012)
19. Information Internet Services. http://www.iis.net/ (Last browsed in February 2012)
20. Quick ‘n Easy FTP Server. http://www.pablosoftwaresolutions.com/html/quick__n_easy_ftp_server.html
21. B. Rudis and P. Kostenbader, “The Enemy Within: Firewalls and Backdoors,” June 2003. http://www.securityfocus.com/infocus/1701