Homework 4 – TCP/IP Vulnerability Analysis

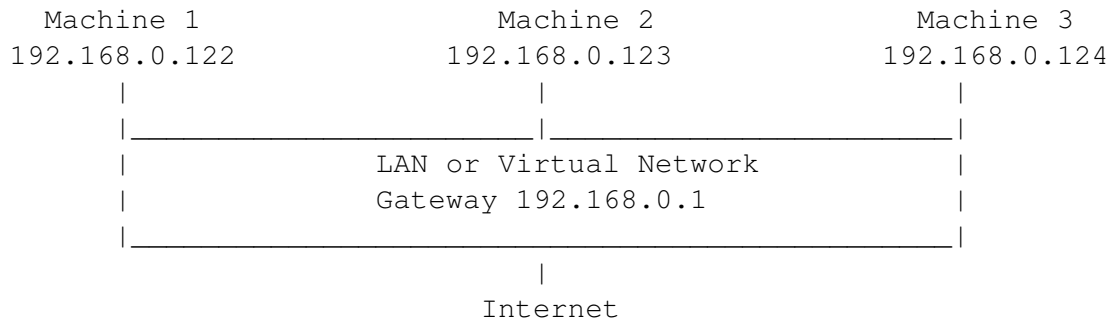
1 Overview

The learning objective of this homework is for students to gain the first-hand experience on the vulnerabilities of TCP/IP protocols, as well as on attacks against these vulnerabilities. The vulnerabilities in the TCP/IP protocols represent a special genre of vulnerabilities in protocol designs and implementations; they provide an invaluable lesson as to why security should be designed in from the beginning, rather than being added as an afterthought. Moreover, studying these vulnerabilities help students understand the challenges of network security and why many network security measures are needed. Vulnerabilities of the TCP/IP protocols occur at several layers.

2 Lab Environment

2.1 Environment Setup

Network Setup. To conduct this lab, students need to run at least 3 connected (virtual) machines. One computer is used for attacking, the second computer is used as the victim, and the third computer is used as the observer. Students can set up 3 virtual machines on the same host computer, or they can set up 2 virtual machines, and then use the host computer as the third computer. For this lab, we put all these three machines on the same LAN, the configuration is described in the following (it is not necessary to use the exact same IP addresses show below):



You can copy/clone some VMs from the given one and run these VMs with multiple VMware player instances. To make these VMs connected, you need to put them all in the same subnet (e.g., 192.168.0.X) and with the same subnet mask (e.g., 255.255.255.0). You can change the network adapter setting of each VM and the network configuration inside the VM to do so (and you can use ifconfig command to know the actual assigned IP addresses).

`Netwox` **Tools.** We need tools to send out network packets of different types and with different contents. You can modify the spoofing program you have written in homework 1. Or we can also use `Netwag` to do that. However, the GUI interface of `Netwag` makes it difficult for us to automate our process. Therefore, we strongly suggest that students use its command-line version, the `Netwox` command, which is the underlying command invoked by `Netwag`.

`Netwox` consists of a suite of tools, each having a specific number. You can run the command like the following (the parameters depend on which tool you are using). For some of the tool, you have to run it with the root privilege:

```
$ sudo netwox number [parameters ... ]
```

If you are not sure how to set the parameters, you can look at the manual by issuing "`netwox number --help`". You can also learn the parameter settings by running `Netwag`: for each command you execute from the graphic interface, `Netwag` actually invokes a corresponding `Netwox` command, and it displays the parameter settings. Therefore, you can simply copy and paste the displayed command.

`Scapy` **Tool.** Some of the tasks in this lab can also be conducted using `Scapy`, which is a powerful interactive packet manipulation program. `Scapy` is very well maintained and is widely used; while `Netwox` is not being maintained any more. There are many online tutorials on `Scapy`; we expect students to learn how to use Scapy from those tutorials

`Wireshark` **Tool.** You also need a good network-traffic sniffer tool for this lab. Although `Netwox` comes with a sniffer, you will find that another tool called `Wireshark` is a much better sniffer tool. Both `Netwox` and `Wireshark` can be downloaded. If you are using our pre-built virtual machine, both tools are already installed. To sniff all the network traffic, both tools need to be run by the `root`.

3 Lab Tasks

In this lab, you need to conduct attacks on the TCP/IP protocols. You can use the `Netwox` tools and/or other tools (e.g., the one you build in your homework 1) in the attacks.

To simplify the "guess" of TCP sequence numbers and source port numbers, we assume that attacks are on the same physical network as the victims. Therefore, you can use sniffer tools to get that information. The following is the list of attacks that need to be implemented.

3.1 Task (1) : Surveillance Techniques

Before launching attacks, attackers often use surveillance techniques to identify their targets. They often choose those that are potentially vulnerable. In this lab, students will get familiar with two of the surveillance techniques:

- *Port Scanning:* The objective of port scanning is to find out which ports on the target machine are open. *Nmap* is a popular software for port scanning, and it uses a variety of scanning methods, such as TCP connect scan, SYN stealth scan, FIN scan, Ping scan, UDP scan, IP Protocol scan, etc. Please try at least 5 techniques on a target machine, and report your observations.
- Fingerprinting Operating Systems: The objective of OS fingerprinting is to find out what operating systems the target machine is running. OS fingerprinting is normally done by probing the differences in the implementation of the TCP/IP stack among operating systems. The differences include the handling of FIN, DON'T fragment bit, fragmentation, TCP ISN sampling, TCP options, etc. Please use *Nmap* to fingerprint `Linux` and `Windows` (if you also use `Windows`); report your observations.

3.2 Task (2) : ARP cache poisoning

The ARP cache is an important part of the ARP protocol. Once a mapping between a MAC address and an IP address is resolved as the result of executing the ARP protocol, the mapping will be cached. Therefore, there is no need to repeat the ARP protocol if the mapping is already in the cache. However, because the

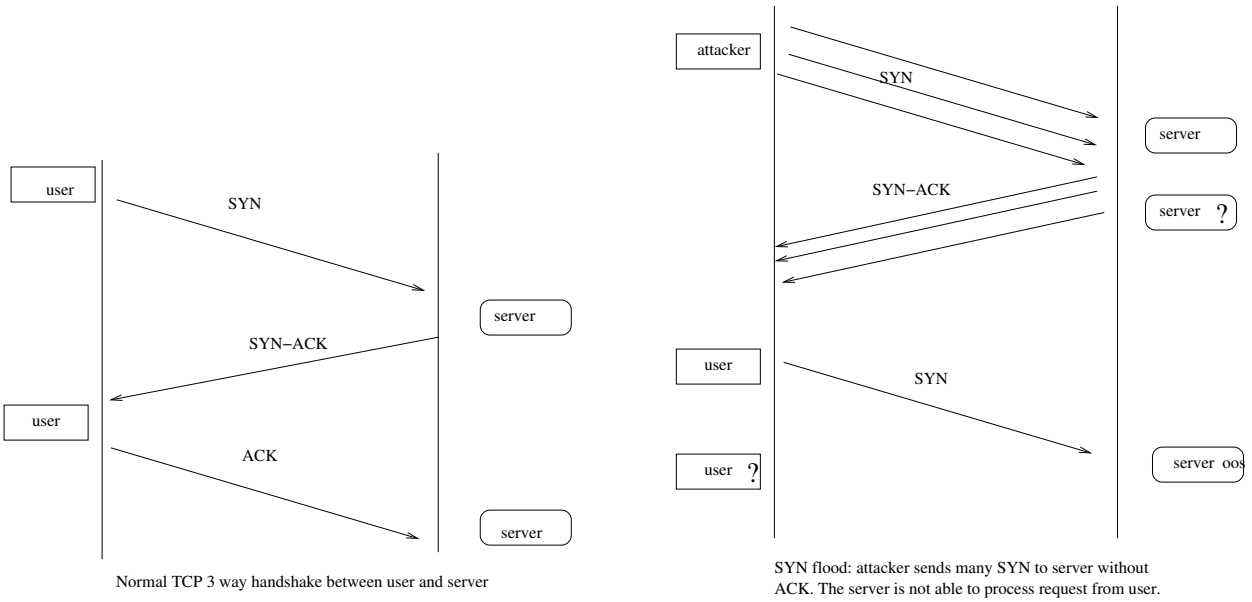


Figure 1: SYN Flood

ARP protocol is stateless, the cache can be easily poisoned by maliciously crafted ARP messages. Such an attack is called the ARP cache poisoning attack.

In such an attack, attackers use spoofed ARP messages to trick the victim to accept an invalid MAC-to-IP mapping, and store the mapping in its cache. There can be various types of consequences depending on the motives of the attackers. For example, attackers can launch a DoS attack against a victim by associating a nonexistent MAC address to the IP address of the victim's default gateway; attackers can also redirect the traffic to and from the victim to another machine, etc.

In this task, you need to demonstrate how the ARP cache poisoning attack work. Several commands can be useful in this task. In `Linux` we can use command `arp` to check the current mapping between IP address and MAC.

3.3 Task (3) : SYN Flooding Attack

SYN flood is a form of DoS attack in which attackers send many SYN requests to a victim's TCP port, but the attackers have no intention to finish the 3-way handshake procedure. Attackers either use spoofed IP address or do not continue the procedure. Through this attack, attackers can flood the victim's queue that is used for half-opened connections, i.e. the connections that has finished SYN, SYN-ACK, but has not yet got a final ACK back. When this queue is full, the victim cannot take any more connection. Figure 1 illustrates the attack.

The size of the queue has a system-wide setting. In `Linux`, we can check the system queue size setting using the following command:

```
$ sudo sysctl -q net.ipv4.tcp_max_syn_backlog
```

We can use command "`netstat -na`" to check the usage of the queue, i.e., the number of half-opened connection associated with a listening port. The state for such connections is `SYN-RECV`. If the 3-way handshake is finished, the state of the connections will be `ESTABLISHED`.

In this task, you need to demonstrate the SYN flooding attack. You can use the Netwox tool to conduct the attack, and then use a sniffer tool to capture the attacking packets. While the attack is ongoing, run the "`netstat -na`" command on the victim machine, and compare the result with that before the attack. Please also describe how you know whether the attack is successful or not.

For your convenience, the corresponding Netwox tool for this task is numbered 76.

SYN Cookie Countermeasure: If your attack seems unsuccessful, one thing that you can investigate is whether the SYN cookie mechanism is turned on. SYN cookie is a defense mechanism to counter the SYN flooding attack. The mechanism will kick in if the machine detects that it is under the SYN flooding attack. You can use the `sysctl` command to turn on/off the SYN cookie mechanism:

```
$ sudo sysctl -a | grep cookie     (Display the SYN cookie flag)
$ sudo sysctl -w net.ipv4.tcp_syncookies=0 (turn off SYN cookie)
$ sudo sysctl -w net.ipv4.tcp_syncookies=1 (turn on  SYN cookie)
```

Please run your attacks with the SYN cookie mechanism on and off, and compare the results. In your report, please describe why the SYN cookie can effectively protect the machine against the SYN flooding attack. If your instructor does not cover the mechanism in the lecture, you can find how the SYN cookie mechanism works from the Internet.

Note on Scapy : Although theoretically, we can use Scapy for this task, we have observed that the number of packets sent out by Scapy per second is much smaller than that by Netwox. This low rate makes it difficult for the attack to be successful. We were not able to succeed in SYN flooding attacks using Scapy.

3.4 Task (4) : TCP RST Attacks on `telnet` and `ssh` Connections

The TCP RST Attack can terminate an established TCP connection between two victims. For example, if there is an established `telnet` connection (TCP) between two users A and B, attackers can spoof a RST packet from A to B, breaking this existing connection. To succeed in this attack, attackers need to correctly construct the TCP RST packet.

In this task, you need to launch an TCP RST attack to break an existing `telnet` connection between A and B. After that, try the same attack on an `ssh` connection. Please describe your observations. To simply the lab, we assume that the attackers and the victims are on the same LAN, i.e., attackers can observe the TCP traffic between A and B.

Using Netwox. The corresponding Netwox tool for this task is numbered 78. You can type ”netwox 78 –help” to get the help information.

Using Scapy. Please also use Scapy to conduct the TCP RST attack. A skeleton code is provided in the following (you need to replace each @@@@ with an actual value):

```
#!/usr/bin/python
from scapy.all import *

ip  = IP(src="@@@@", dst="@@@@")
tcp = TCP(sport=@@@@, dport=@@@@, flags="@@@@", seq=@@@@, ack=@@@@)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)
```

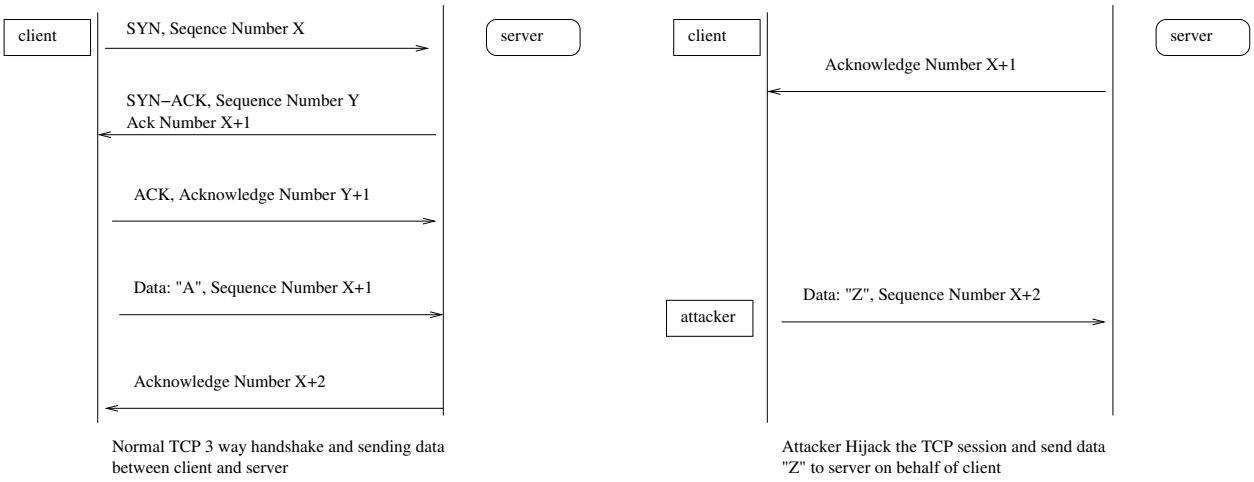


Figure 2: TCP Session Hijacking

3.5 Task (5) : TCP RST Attacks on Video Streaming Applications

Let us make the TCP RST attack more interesting by experimenting it on the applications that are widely used in nowadays. We choose the video streaming application in this task. For this task, you can choose a video streaming web site that you are familiar with (we will not name any specific web site here). Most of video sharing websites establish a TCP connection with the client for streaming the video content. The attacker's goal is to disrupt the TCP session established between the victim and video streaming machine. To simplify the lab, we assume that the attacker and the victim are on the same LAN. In the following, we describe the common interaction between a user (the victim) and some video-streaming web site:

- The victim browses for a video content in the video-streaming web site, and selects one of the videos for streaming.
- Normally video contents are hosted by a different machine, where all the video contents are located. After the victim selects a video, a TCP session will be established between the victim machine and the content server for the video streaming. The victim can then view the video he/she has selected.

Your task is to disrupt the video streaming by breaking the TCP connection between the victim and the content server. You can let the victim user browse the video-streaming site from another (virtual) machine or from the same (virtual) machine as the attacker. Please be noted that, to avoid liability issues, any attacking packets should be targeted at the vitim machine (which is the machine run by yourself), not the content server machine (which does not belong to you). You only need to use Netwox for this task.

3.6 Task (6) : TCP Session Hijacking

The objective of the TCP Session Hijacking attack is to hijack an existing TCP connection (session) between two victims by injecting malicious contents into this session. If this connection is a `telnet` session, attackers can inject malicious commands into this session, causing the victims to execute the malicious commands. We will use `telnet` in this task. We also assume that the attackers and the victims are on the same LAN.

Note on using Netwox/Wireshark. The corresponding Netwox tool for this task is numbered 40. If you use Wireshark to observe the network traffic, you should be aware that when Wireshark displays the TCP sequence number, by default, it displays the relative sequence number, which equals to the actual sequence number minus the initial sequence number. If you want to see the actual sequence number in a packet, you need to right click the TCP section of the Wireshark output, and select "Protocol Preference". In the popup window, uncheck the "Relative Sequence Number and Window Scaling" option.

In the netwox command (number 40), the tcp-data part only takes hex data. If we want to inject a command string, which is typically represented as a human-readable ASCII string, we need to convert it into a hex string. There are many ways to do that, but we will just use a very simple command in Python. In the following, we convert an ASCII string "Hello World" to a hex string (the quotation marks are not included).

```
$ python
>>> "Hello World".encode("hex")
48656c6c6f20576f726c64
```

Using Scapy. Please also use Scapy to conduct the TCP Session Hijacking attack. A skeleton code is provided in the following (you need to replace each @@@@ with an actual value):

```
#!/usr/bin/python
from scapy.all import *

ip  = IP(src="@@@@", dst="@@@@")
tcp = TCP(sport=@@@@, dport=@@@@, flags="@@@@", seq=@@@@, ack=@@@@)
data = "@@@@"
pkt = ip/tcp/data
ls(pkt)
send(pkt,verbose=0)
```

3.7 Investigation

The level of difficulty in TCP attacks depends on a number of factors. Please investigate the following and write down your discoveries and observations in your report.

- Study the pattern of the Initial Sequence Numbers (ISN), and answer whether the patterns are predictable.
- Study the TCP window size, and describe your observations.
- Study the pattern of the source port numbers, and answer whether the patterns are predictable.

3.8 Note

It should be noted that because some vulnerabilities might have already been fixed in Linux, some of the above attacks might fail in Linux, but they could still be successful against other operating systems.

4 Homework Report

You should submit a report (remember to put your name and UIN in). The report should cover the following sections:

- **Design:** The design of your attacks, including the attacking strategies, the packets that you use in your attacks, the tools that you used, etc.
- **Observation:** Is your attack successful? How do you know whether it has succeeded or not? What do you expect to see? What have you observed? Is the observation a surprise to you?
- **Explanation:** Some of the attacks might fail. If so, you need to find out what makes them fail. You can find the explanations from your own experiments (preferred) or from the Internet. If you get the explanation from the Internet, you still need to find ways to verify those explanations through your own experiments. You need to convince us that the explanations you get from the Internet can indeed explain your observations.
- **Defense thought:** I encourage you to think hard on how to defend against this type of attacks after you understand the vulnerability.

References

[1] Netwox/Netwag Guides, by Sridhar Iyer. http://courses.cse.tamu.edu/guofei/csce465/netwox.pdf

[2] ICMP attacks against TCP, by Gont, F. http://www.gont.com.ar/drafts/icmp-attacks-against-tcp.html

[3] Slipping in the Window: TCP Reset attacks, by Paul A. Watson. http://osvdb.org/ref/04/04030-SlippingInTheWindow_v1.0.doc

[4] Strange Attractors and TCP/IP Sequence Number Analysis by Zalewski. http://lcamtuf.coredump.cx/newtcp/

[5] Remote OS detection via TCP/IP Stack FingerPrinting by Fyodor, 1998. http://www.insecure.org/nmap/nmap-fingerprinting-article.html