

Attack Lab: Attacks on TCP/IP Protocols

Copyright © 2006 - 2009 Wenliang Du, Syracuse University.

The development of this document is funded by the National Science Foundation's Course, Curriculum, and Laboratory Improvement (CCLI) program under Award No. 0618680 and 0231122. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation. A copy of the license can be found at <http://www.gnu.org/licenses/fdl.html>.

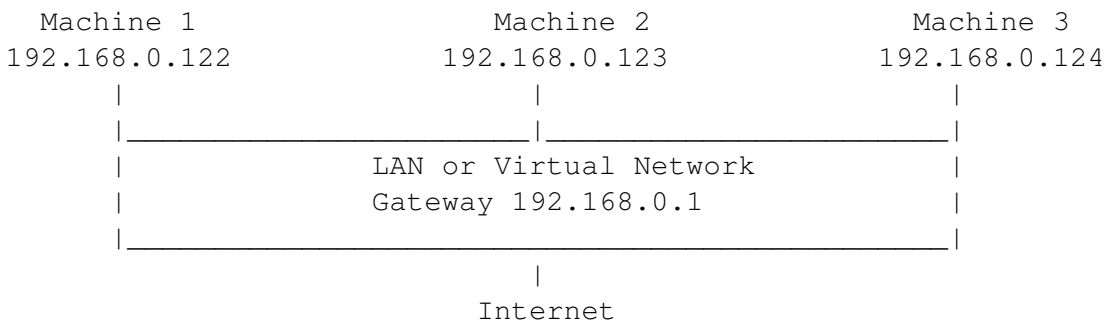
1 Lab Overview

The learning objective of this lab 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 have at least 3 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:



Operating System. This lab can be carried out using a variety of operating systems. Our pre-built virtual machine is based on Ubuntu Linux, and all the tools needed for this lab are already installed. If you prefer to use other Unix operating systems, such as Fedora, you should feel free to use them; however, some of the commands used in this lab description might not work or exist in other operating systems.

Netwox Tools. We need tools to send out network packets of different types and with different contents. We can use Netwag to do that. However, the GUI interface of Netwag makes it difficult for us to auto-

mate 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:

```
# 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.

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`.

Enabling the ftp and telnet Servers. For this lab, you may need to enable the `ftp` and `telnet` servers. For the sake of security, these services are usually disabled by default. To enable them in our pre-built Ubuntu virtual machine, you need to run the following commands as the `root` user:

```
Start the ftp server
# service vsftpd start

Start the telnet server
# service openbsd-inetd start
```

2.2 Note for Instructors

For this lab, a lab session is desirable, especially if students are not familiar with the tools and the environments. If an instructor plans to hold a lab session (by himself/herself or by a TA), it is suggested the following be covered in the lab session. We assume that the instructor has already covered the concepts of the attacks in the lecture, so we do not include them in the lab session.

- The use of virtual machine software.
- The use of `Wireshark`, `Netwag`, and `Netwox` tools.
- Using the `Netwox` command-line tool to create arbitrary TCP, UDP, IP packets, etc.

3 Lab Tasks

In this lab, students need to conduct attacks on the TCP/IP protocols. They can use the `Netwox` tools and/or other tools in the attacks. All the attacks are performed on Linux operating systems. However, instructors can require students to also conduct the same attacks on other operating systems and compare the observations.

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 sniffers to get those information. The following is the list of attacks that need to be implemented.

3.1 ARP cache poisoning

In ARP cache poisoning attack, attackers use spoofed ARP message in LAN to associate MAC address and IP address in a malicious way. 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, or sniffer the victim's IP traffic in a switch gateway by poisoning both victim and gateway.

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.

3.2 ICMP Redirect Attack

The ICMP `redirect` message is used by routers to provide the up-to-date routing information to new hosts, which initially have minimal routing information. In an ICMP redirect attack, the attacker can send a spoofed ICMP `redirect` message to a victim; this message can cause the victim's routing information to be modified. In Linux, the routing information can be displayed using the command `route`.

3.3 SYN Flooding Attacks

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.

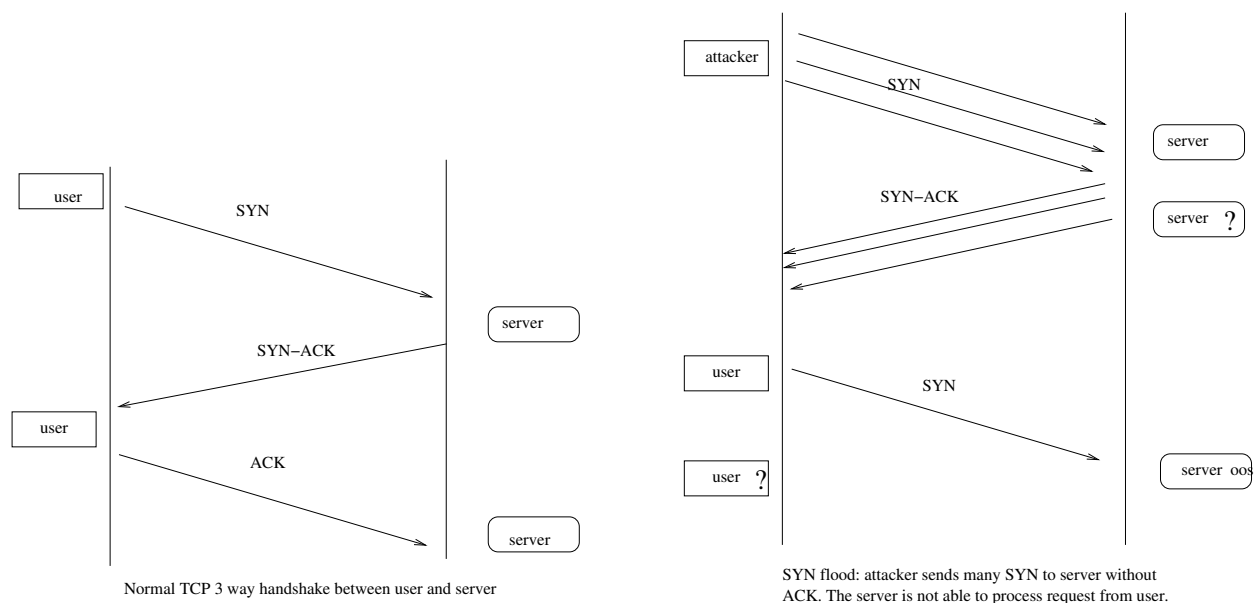


Figure 1: SYN Flood

The size of the queue has a system-wide setting. The application that uses this queue, for example ftp, can also specify its size. In Linux, we can check the system queue size setting using the following command:

```
# sysctl -q net.ipv4.tcp_max_syn_backlog
```

We can use command "`netstat -na`" to check the usage of the queue, which is the number of half opened connection associated with a listening port. The state for such connections is `SYN-RECV`.

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. You can use the `sysctl` command to turn on/off the SYN cookie mechanism:

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

3.4 TCP RST Attacks on telnet Connections

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, our goal is to launch an TCP RST attack to break an existing `telnet` connection between A and B. To simplify 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.

3.5 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 the hosts the streaming of the content from a different machine, where all the video contents are located. Now 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. Any attacking packets should be targeted at the victim machine, not the content server machine.

3.6 ICMP Blind Connection-Reset Attacks

ICMP messages can also be used achieve the connection-resetting attack. To do this, attackers send an ICMP error message that indicates a "hard error" to either of the two endpoints of a TCP connection. The connection can be immediately torn down as RFC 1122 states that *a host should abort the corresponding connection when receiving such an ICMP error message*. RFC 1122 defines "hard errors" as ICMP error

messages of type 3 (Destination Unreachable) codes 2 (protocol unreachable), 3 (port unreachable), and 4 (fragmentation needed and DF bit set).

When launching this attack, students should be noted that some systems may reasonably ignore this type of ICMP errors in certain TCP state.

3.7 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.

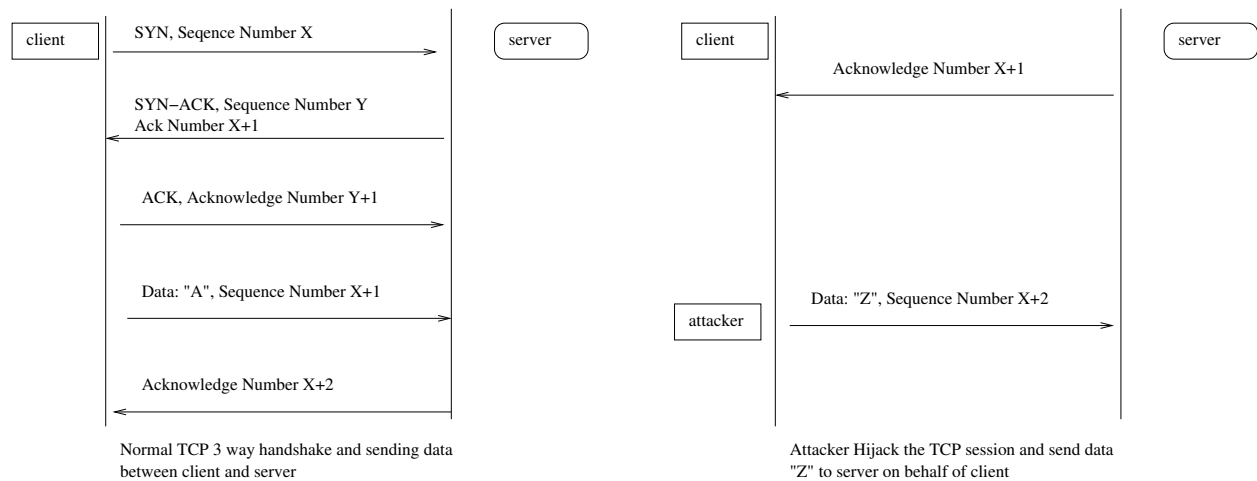


Figure 2: TCP Session Hijacking

Note: 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.

3.8 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 lab reports.

- 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.

3.9 Note

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

4 Lab Report

You should submit a lab report. 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.