Lab on Firewall, Ethereal, ICMP and ARP

in SANDBOX lab (PL112)

CSE398: Network Systems Design, Lehigh University
Instructor: Dr. Liang Cheng, Assistant Professor, Computer Science and Engineering

Lab Graduate Assistant: Yaoyao Zhu

March 16th, 2005

Introduction

This lab session includes two parts:

1. Configuring firewalls using `iptables` in Linux;
2. Using `ethereal` to capture network packets and observe the packets in various layers;

Each machine has one network interface cards (NIC), `eth0`. Their IP-address configurations are DHCP enabled.

The `eth0` interface belongs to a 192.168.1.0/24 network and is connected with a switch, which enables an Internet connection. A machine with an IP address 192.168.1.200 is configured and activated in this 192.168.1.0/24 network to be used for `ping` checking functionality.

Procedure

Firewall

1. Make a temporary directory called "`temp`" and perform the rest of the steps under "temp" directory.
2. Open a terminal window. Try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. They all should be ping-able, otherwise please ask the lab graduate assistant for help.
3. Open a web browser, and visit http://www.cse.lehigh.edu/~cheng/Teaching/CSE398-05/firewall-031605.tar to download `firewall-031605.tar` to the `temp` directory that you have just created.
4. Untar the `firewall-031605.tar` using "tar –xvf `firewall-031605.tar`". It will automatically create a `firewall-031605` sub-directory.
5. Change directory to `firewall-031605`. And modify all the script files to be executable by using "`chmod 700 *`".
6. Study the firewall script `accept-all` based on the Linux command **`iptables`**, which has been briefly described in the appendix of this document.
7. Run the script `accept-all` using "./`accept-all`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. They all should be ping-able, otherwise please ask the lab graduate assistant for help.

8. Study the firewall script `deny-all` based on the Linux command `iptables`, which has been briefly described in the appendix of this document.
9. Run the script `deny-all` using "./`deny-all`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. They all should NOT be ping-able, otherwise please ask the lab graduate assistant for help.
10. Modify the `deny-all` script to enable the Loopback interface.
11. Run the script `deny-all` using "./`deny-all`". Then try to ping `localhost`, and it should be ping-able.
    Then try to ping 192.168.1.200, and www.lehigh.edu. They all should NOT be ping-able, otherwise please ask the lab graduate assistant for help.
12. Modify the `deny-all` script to enable the `LOCAL_INTERFACE` interface.
13. Run the script `deny-all` using "./`deny-all`". Then try to ping `localhost`, and it should be ping-able.
    Then try to ping 192.168.1.200 and www.lehigh.edu, and they should be ping-able.
14. Run the script `accept-all` using "./`accept-all`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. Then they all should be ping-able, otherwise please ask the lab graduate assistant for help.
15. Study the firewall script `block-ip` based on the Linux command `iptables`, which has been briefly described in the appendix of this document.
16. Run the script `block-ip` using "./`block-ip 192.168.1.200`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. Only "ping 192.168.1.200" should be blocked.
17. Study the firewall script `unblock-ip` based on the Linux command `iptables`, which has been briefly described in the appendix of this document.
18. Run the script `unblock-ip` using "./`unblock-ip 192.168.1.200`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. The blocked 192.168.1.200 should be enabled to be ping again.
19. Run the script `accept-all` using "./`accept-all`". Then try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. They all should be ping-able, otherwise please ask the lab graduate assistant for help.

ethereal

20. Click the main menu and from the "Internet" menu to the "more Internet application" menu, and run the `ethereal` application.
21. From the "Capture" menu click "Start", choose `eth0` from the popup dialog, then click "OK" button to start capturing packets across `eth0`. Note that `eth0` is connected to the Internet.
22. Go to the course website: http://www.cse.lehigh.edu/~cheng/
23. After the webpage has been correctly displayed, stop packets capturing.
24. Observe TCP three-way handshaking process and the sequence number mechanism.
25. Observe HTTP protocol such as the GET command.
26. Choose the captured TCP packet with SYN and ACK flag, and observe packet details in various layers.
    Ethernet layer: destination and source MAC addresses

IP layer: version, header length, TTL, flag, protocol, source and destination IP addresses
Transmission layer: source port and destination port, SN, ACK number, header length, Flag, window size, checksum

Study the ICMP and ARP protocol

27. Try to ping `localhost`, 192.168.1.200, and www.lehigh.edu. They all should be ping-able, otherwise please ask the lab graduate assistant for help.
28. Click "Close" from the "File" menu of the ethereal application, which clears the screen of the captured packets.
29. Start capturing packets across `eth0`, which is connected to a hub.
30. Ask your neighbor group to ping 192.168.1.200.
31. Observe whether you have captured any ICMP packet because of "ping 192.168.1.200" by your neighbor group. Confirm it from the source IP address of the ICMP packets. You should be able to capture the ICMP packets from your neighbor group. If not, please ask the lab graduate assistant for help.
32. Choose any captured ICMP packet, and observe packet details in various layers:
    Ethernet layer: destination and source MAC addresses
    IP layer: version, header length, TTL, flag, protocol, source and destination IP addresses
    ICMP layer: type, checksum, ID, SN
33. After your observation, click "Close" from the "File" menu of the ethereal application, which clears the screen of the captured packets.
34. Observe whether you have captured any ARP packet because of "ping www.ees.lehigh.edu" by your neighbor group. If yes, observe packet details in various layers.
35. Discuss your observation and confirm it with the lab graduate assistant.
36. What you have observed in terms of the following information?
    Ethernet layer: destination and source MAC addresses
    IP layer: version, header length, TTL, flag, protocol, source and destination IP addresses

Last question:

37. Ask the ip address of one of your neighbor. Create your own script to make your neighbor not being able to ping your machine but you can still ping your neighbor's machine. Study the manual of "`iptables`" by keying in "man `iptables`".

Appendix. (from "`man iptables`")

NAME

iptables - administration tool for IPv4 packet filtering and NAT

SYNOPSIS

iptables [-t table] -[ADC] chain rule-specification [options]
iptables [-t table] -I chain [rulenum] rule-specification [options]
iptables [-t table] -R chain rulenum rule-specification [options]
iptables [-t table] -D chain rulenum [options]
iptables [-t table] -[LFZ] [chain] [options]
iptables [-t table] -N chain
iptables [-t table] -X [chain]
iptables [-t table] -P chain target [options]
iptables [-t table] -E old-chain-name new-chain-name

DESCRIPTION

Iptables is used to set up, maintain, and inspect the tables of IP packet filter rules in the Linux kernel. Several different tables may be defined. Each table contains a number of built-in chains and may also contain user-defined chains.

Each chain is a list of rules which can match a set of packets. Each rule specifies what to do with a packet that matches. This is called a 'target', which may be a jump to a user-defined chain in the same table.

TARGETS

A firewall rule specifies criteria for a packet, and a target. If the packet does not match, the next rule in the chain is the examined; if it does match, then the next rule is specified by the value of the target, which can be the name of a user-defined chain or one of the special values ACCEPT, DROP, QUEUE, or RETURN.

ACCEPT means to let the packet through. DROP means to drop the packet on the floor. QUEUE means to pass the packet to userspace (if supported by the kernel). RETURN means stop traversing this chain and resume at the next rule in the previous (calling) chain. If the end of a built-in chain is reached or a rule in a built-in chain with target RETURN is matched, the target specified by the chain policy determines the fate of the packet.

TABLES

There are currently three independent tables (which tables are present at any time depends on the kernel configuration options and which modules are present).

-t, --table table
This option specifies the packet matching table which the command should operate on. If the kernel is configured with automatic module loading, an attempt will be made to load the appropriate module for that table if it is not already there.

The tables are as follows:

filter This is the default table (if no -t option is passed). It contains the built-in chains INPUT (for packets coming into the box itself), FORWARD (for packets being routed through the box), and OUTPUT (for locally-generated packets).

nat This table is consulted when a packet that creates a new connection is encountered. It consists of three built-ins: PREROUTING (for altering packets as soon as they come in), OUTPUT (for altering locally-generated packets before routing), and POSTROUTING (for altering packets as they are about to go out).

mangle This table is used for specialized packet alteration. Until kernel 2.4.17 it had two built-in chains: PREROUTING (for altering incoming packets before routing) and OUTPUT (for altering locally-generated packets before routing). Since kernel 2.4.18, three other built-in chains are also supported: INPUT (for packets coming into the box itself), FORWARD (for altering packets being routed through the box), and POSTROUTING (for altering packets as they are about to go out).

OPTIONS

The options that are recognized by iptables can be divided into several different groups.

COMMANDS

These options specify the specific action to perform. Only one of them can be specified on the command line unless otherwise specified below. For all the long versions of the command and option names, you need to use only enough letters to ensure that iptables can differentiate it from all other options.

-A, --append chain rule-specification
Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.

-D, --delete chain rule-specification

-D, --delete chain rulenum

Delete one or more rules from the selected chain. There are two versions of this command: the rule can be specified as a number in the chain (starting at 1 for the first rule) or a rule to match.

-I, --insert chain [rulenum] rule-specification

Insert one or more rules in the selected chain as the given rule number. So, if the rule number is 1, the rule or rules are inserted at the head of the chain. This is also the default if no rule number is specified.

-R, --replace chain rulenum rule-specification

Replace a rule in the selected chain. If the source and/or destination names resolve to multiple addresses, the command will fail. Rules are numbered starting at 1.

-L, --list [chain]

List all rules in the selected chain. If no chain is selected, all chains are listed. As every other iptables command, it applies to the specified table (filter is the default), so NAT rules get listed by

iptables -t nat -n -L

Please note that it is often used with the -n option, in order to avoid long reverse DNS lookups. It is legal to specify the -Z (zero) option as well, in which case the chain(s) will be atomically listed and zeroed. The exact output is affected by the other arguments given. The exact rules are suppressed until you use

iptables -L -v

-F, --flush [chain]

Flush the selected chain (all the chains in the table if none is given). This is equivalent to deleting all the rules one by one.

-Z, --zero [chain]

Zero the packet and byte counters in all chains. It is legal to specify the -L, --list (list) option as well, to see the counters immediately before they are cleared. (See above.)

-N, --new-chain chain

Create a new user-defined chain by the given name. There must be no target of that name already.

-X, --delete-chain [chain]

Delete the optional user-defined chain specified. There must be no references to the chain. If there are, you must delete or replace the referring rules before the chain can be deleted. If no argument is given, it will attempt to delete every non-builtin chain in the table.

-P, --policy chain target
Set the policy for the chain to the given target. See the section TARGETS for the legal targets. Only built-in (non-user-defined) chains can have policies, and neither built-in nor user-defined chains can be policy targets.

-E, --rename-chain old-chain new-chain
Rename the user specified chain to the user supplied name. This is cosmetic, and has no effect on the structure of the table.

-h Help. Give a (currently very brief) description of the command syntax.

PARAMETERS
The following parameters make up a rule specification (as used in the add, delete, insert, replace and append commands).

-p, --protocol [!] protocol
The protocol of the rule or of the packet to check. The specified protocol can be one of tcp, udp, icmp, or all, or it can be a numeric value, representing one of these protocols or a different one. A protocol name from /etc/protocols is also allowed. A "!" argument before the protocol inverts the test. The number zero is equivalent to all. Protocol all will match with all protocols and is taken as default when this option is omitted.

-s, --source [!] address[/mask]
Source specification. Address can be either a network name, a hostname (please note that specifying any name to be resolved with a remote query such as DNS is a really bad idea), a network IP address (with /mask), or a plain IP address. The mask can be either a network mask or a plain number, specifying the number of 1's at the left side of the network mask. Thus, a mask of 24 is equivalent to 255.255.255.0. A "!" argument before the address specification inverts the sense of the address. The flag --src is an alias for this option.

-d, --destination [!] address[/mask]
Destination specification. See the description of the -s

(source) flag for a detailed description of the syntax. The flag --dst is an alias for this option.

-j, --jump target
This specifies the target of the rule; i.e., what to do if the packet matches it. The target can be a user-defined chain (other than the one this rule is in), one of the special builtin targets which decide the fate of the packet immediately, or an extension (see EXTENSIONS below). If this option is omitted in a rule, then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented.

-i, --in-interface [!] name
Name of an interface via which a packet is going to be received (only for packets entering the INPUT, FORWARD and PREROUTING chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

-o, --out-interface [!] name
Name of an interface via which a packet is going to be sent (for packets entering the FORWARD, OUTPUT and POSTROUTING chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

[!] -f, --fragment
This means that the rule only refers to second and further fragments of fragmented packets. Since there is no way to tell the source or destination ports of such a packet (or ICMP type), such a packet will not match any rules which specify them. When the "!" argument precedes the "-f" flag, the rule will only match head fragments, or unfragmented packets.

-c, --set-counters PKTS BYTES
This enables the administrator to initialize the packet and byte counters of a rule (during INSERT, APPEND, REPLACE operations).

MATCH EXTENSIONS

iptables can use extended packet matching modules. These are loaded in two ways: implicitly, when -p or --protocol is specified, or with the -m or --match options, followed by the matching module name; after these, various extra command line options become available, depending on the specific module. You can specify multiple extended match modules in one line, and you can use the -h or --help options after the

module has been specified to receive help specific to that module.

The following are included in the base package, and most of these can be preceded by a ! to invert the sense of the match.

tcp

These extensions are loaded if '--protocol tcp' is specified. It provides the following options:

--source-port [!] port[:port]

Source port or port range specification. This can either be a service name or a port number. An inclusive range can also be specified, using the format port:port. If the first port is omitted, "0" is assumed; if the last is omitted, "65535" is assumed. If the second port greater then the first they will be swapped. The flag --sport is a convenient alias for this option.

--destination-port [!] port[:port]

Destination port or port range specification. The flag --dport is a convenient alias for this option.

--tcp-flags [!] mask comp

Match when the TCP flags are as specified. The first argument is the flags which we should examine, written as a comma-separated list, and the second argument is a comma-separated list of flags which must be set. Flags are: SYN ACK FIN RST URG PSH ALL NONE. Hence the command

iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST SYN

will only match packets with the SYN flag set, and the ACK, FIN and RST flags unset.

[!] --syn

Only match TCP packets with the SYN bit set and the ACK and FIN bits cleared. Such packets are used to request TCP connection initiation; for example, blocking such packets coming in an interface will prevent incoming TCP connections, but outgoing TCP connections will be unaffected. It is equivalent to --tcp-flags SYN,RST,ACK SYN. If the "!" flag precedes the "--syn", the sense of the option is inverted.

--tcp-option [!] number

Match if TCP option set.

--mss value[:value]

Match TCP SYN or SYN/ACK packets with the specified MSS value

(or range), which control the maximum packet size for that connection.

icmp

This extension is loaded if '--protocol icmp' is specified. It provides the following option:

--icmp-type [!] typename

This allows specification of the ICMP type, which can be a numeric ICMP type, or one of the ICMP type names shown by the command

iptables -p icmp -h