Man-in-the-Middle Attack
(Supplementary Materials)

Reverse Path Filtering

Introduction

- The main functionality of a router is to route packets from one place to another.
- Linux machine can be used as router on your network that will route substantial amount of traffic without any issues, if configured correctly.
- Due to the increasing amount of malicious and attack traffic on the internet, it has become very much necessary to take some extra care while configuring routes on a Linux machine or physical router's.
- One of the major problems that internet security people are dealing with today, is **spoofing**.

What is IP address spoofing?

- IP spoofing is a method adopted by attackers to send forged source address in their attack traffic:
  - i.e., they can send an IP packet with an IP address of their wish!
- Most of the times, spoofing is used by an attacker mainly for the following reasons:
  - To conduct a DDOS attack, and he does not want the response from the target machine to reach him.
  - To compromise source-based authentication.
- Spoofing can be controlled to a certain extent by using **Reverse Path filtering** (not fully though).

What is reverse path filtering (RPF)?

- Reverse path filtering is a mechanism adopted by the Linux kernel, as well as most of the networking devices out there **to check whether a receiving packet source address is routable**.
  - So in other words, when a machine with reverse path filtering enabled receives a packet, the machine will first check whether the source of the received packet is reachable through the interface it came in.
    - If it is routable through the interface which it came, then the machine will **accept the packet**.
    - If it is not routable through the interface, which it came, then the machine will **drop that packet**.

Reverse Path Filtering (cont'd)

- By default, routers route everything, even packets which 'obviously' don't belong on your network.
  - Lots of people will want to turn this feature off, so that if you have an interface with a route of 195.96.96.0/24 to it, you do not expect packets from 212.64.94.1 to arrive there!
  - There are files in **/proc** where you can tell the kernel to do this for you.
    - The method is called "Reverse Path Filtering".
      - Basically, if the reply to this packet wouldn't go out the interface this packet came in, then this is a bogus packet and should be ignored.

IP Tables-Based Masquerading

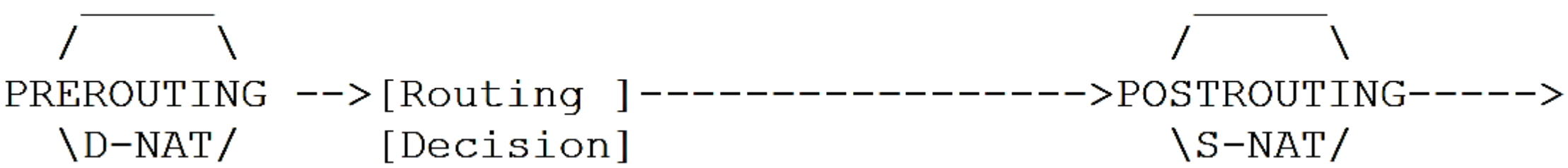
- The Linux kernel entails a packet filtering framework viz. netfilter that enables a Linux machine to spoof source or destination IP addresses!
  - ➢ The command for this is called iptables -t nat, and it manages the table that entails rules for address masquerading.
    - ❖ This table has two types of rules:

      (i) PREROUTING: responsible for packets that just arrived at the network interface, enforcing rules before any routing decision has been made.

      (ii) POSTROUTING: responsible for packets with a recipient outside the linux machine, enforcing rules before the packet leaves through the network interface (after routing rules).
    - ❖ each rule is examined in order until one matches.

Controlling What To NAT

- PREROUTING rules are used for Destination NAT (as packets first come in), and
- POSTROUTING rules for Source NAT (as packets leave).



- When a packet passes we look up what connection it is associated with.
  - If it's a new connection, we look up the corresponding chain in the NAT table to see what to do with it. The answer it gives will apply to all future packets on that connection.

Simple Selection using iptables

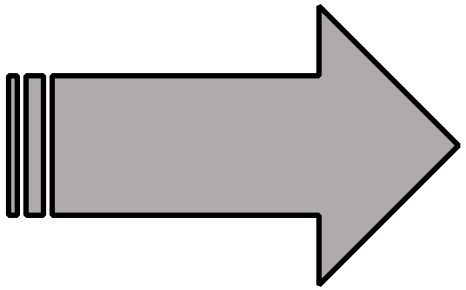
- iptables takes a number of standard options as listed below:
  - The most important option here is the table selection option, `-t'. For all NAT operations, you will want to use `-t nat' for the NAT table.
  - The second most important option to use is `-A' to append a new rule at the end of the chain (e.g. `-A POSTROUTING'), or `-I' to insert one at the beginning (e.g. `-I PREROUTING').
  - You can specify the source (`-s' or `--source') and destination (`-d' or `--destination') of the packets you want to NAT.
    - These options can be followed by a single IP address (e.g. 192.168.1.1), a name (e.g. www.gnumonks.org), or a network address (e.g. 192.168.1.0/24 or 192.168.1.0/255.255.255.0).
    - You can specify the incoming (`-i' or `--in-interface') or outgoing (`-o' or `--out-interface') interface to match, but which you can specify depends on which chain you are putting the rule into:
      - at PREROUTING you can only select incoming interface, and
      - at POSTROUTING you can only select outgoing interface.
      - If you use the wrong one, iptables will give an error.

Finer Points Of Selecting What Packets To Mangle

- You can also indicate a specific protocol (`-p' or `--protocol'), such as TCP or UDP; only packets of this protocol will match the rule.
  - Specifying a protocol of tcp or udp then allows extra options:
    - specifically the `--source-port' and `--destination-port' options (abbreviated as `--sport' and `--dport').
      - These options allow you to specify that only packets with a certain source and destination port will match the rule. This is useful for redirecting web requests (TCP port 80 or 8080) and leaving other packets alone.
- All the different qualities you can select a packet by are detailed in painful detail in the manual page (man iptables).

How To Mangle The Packets

- So now we know how to select the packets we want to mangle. To complete our rule, we need to tell the kernel exactly what we want it to do to the packets.



Source NAT

- You want to do Source NAT: change the source address of connections to something different.
  - This is done in the POSTROUTING chain, just before it is finally sent out! It also means that the `-o' (outgoing interface) option can be used.
  - Source NAT is specified using `-j SNAT', and the `--to-source' option specifies an IP address, a range of IP addresses, and an optional port or range of ports (for UDP and TCP protocols only).
    - There is also a '-j MASQUERADE' option that is used when we want to modify the source IP address if the masquerading IP address belongs to the host machine, i.e. it's eth0 IP address.

```
## Change source addresses to 1.2.3.4.
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

## Change source addresses to 1.2.3.4, 1.2.3.5 or 1.2.3.6
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6

## Change source addresses to 1.2.3.4, ports 1-1023
# iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

Destination NAT

- This is done in the PREROUTING chain, just as the packet comes in;
- It also means that the `-i' (incoming interface) option can be used.
- Destination NAT is specified using `-j DNAT', and the `--to-destination' option specifies an IP address, a range of IP addresses, and an optional port or range of ports (for UDP and TCP protocols only).

```
## Change destination addresses to 5.6.7.8
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8

## Change destination addresses to 5.6.7.8, 5.6.7.9 or 5.6.7.10.
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8-5.6.7.10

## Change destination addresses of web traffic to 5.6.7.8, port 8080.
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 \
        -j DNAT --to 5.6.7.8:8080
```