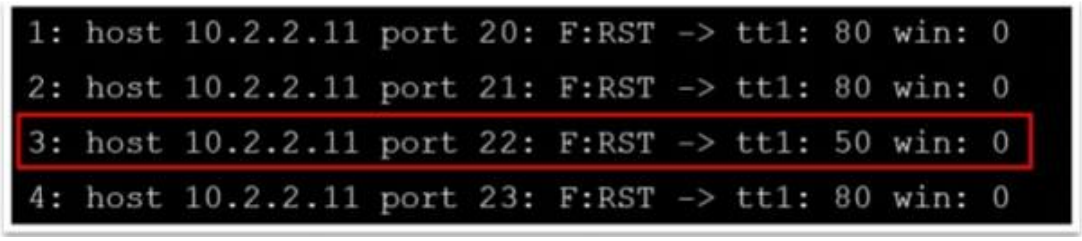
ACK flag probe is used to map out firewall rulesets, determining whether they are stateful or not and which ports are filtered. ACK scan is enabled by specifying the -sA option. Its probe packet has only the ACK flag set (unless you use --scanflags ). The ACK flag probe exploits vulnerabilities of BSD-derived TCP/IP stacks, so this technique is only effective for targets running BSD-derived OSes. There are 2 categories of ACK flag probe scans:

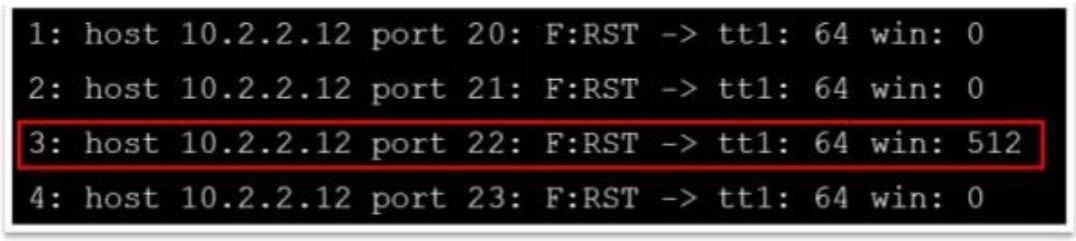
1. **TTL Based ACK flag probe scan**

   In this technique, ACK probe packets are first sent to various TCP ports and then the TTL field value analysis of the received RST packets is performed. In nmap, use the -ttl [time] [target] option to perform a TTL based ack flag probe scan. If the TTL value of the RST packet on a particular port is less than 64, it means the port is open. In the example below, port 22 has a TTL of 50 (below 64), meaning port 22 is open.




2. **Window based ACK flag probe scan**

   Just like TTL, ACK probe packets will be sent to various ports, but what will be analyzed is the window field value of the received RST packet. If the window value of the RST packets on a particular port is non-zero, it means the port is open. In the image below, port 22 is open because it has a non-zero win value. If there is no response from the target and an ICMP unreachable error is obtained (type 3, code 1, 2, 3, 9, 10, or 13), it can be concluded that the port is a filtered port, meaning that the target network uses a filtered mechanism such as a firewall. The advantage of this scanning is that it can generally bypass IDS. While the drawback is that it is very slow, and can only exploit older versions of BSD-derived OS. In nmap, use the -sA option to perform the ACK flag probe scan.



Iptables is a tool or application that functions as a firewall on the Linux operating system, especially on servers that do not use a graphical control panel such as cPanel or Plesk Panel. With iptables, you can manage network traffic on the server such as allowing, blocking or passing incoming and outgoing connections, managing ports and so on. Iptables has several tables that function to determine the direction of data rotation. Each of these tables has rules or a set of rules called a chain.

1. **Filter table** = This table is used to filter incoming, outgoing, or passing packets.

    - ACCEPT : Receive incoming packets
    - REJECT : Reject/Block incoming packets
    - DROP : Drop packet connection
    - LOG : Logging packets
    - INPUT : This chain handles all incoming packets to the server.
    - OUTPUT : This chain handles all packets leaving the server.
    - FORWARD : This chain handles packets that are forwarded through the server.

2. **NAT (Network Address Translation) table** = This table is used to change the origin address of a packet.

    - PRE-ROUTING (dstnat) : Changing the destination address in a data packet.
    - POST-ROUTING (srcnat) : Changes the source address of a data packet.
    - OUTPUT : Used to translate the address of data packets originating from the firewall itself

3. **Mangle table** = This table is used to refine the package management process, and has the ability to use all chains in the iptables above.

**How to use iptables :**

1. Installing iptables
2. Create iptables rules
3. Permanently save the iptables configuration

**Example of using iptables :**

Iptables -I INPUT -s 11.22.33.44/32 -j DROP

**COMMAND** = Commands and rules that are installed on iptables (firewall) have conditions. Basically iptables on the computer is considered an IP TABLE according to its name. The system will only run rules which is in the table. Meanwhile, existing rules on iptables can also be deleted or replaced with other rules. Here are some commands for addition, deletion and similar operations to be treated against the rule.

-A or -append - Adds a rule
-D or -delete - Performs a rule deletion
-R or-replace - Performs replacing rule
-L or-list -I Displays a list of iptables
-F or -flush - Clears the iptables/empty list
-I or -insert - Inserts a rule
-N or -new-chain - Adds a new chain
-X or -delete-chain - Performs a chain delete
-P or -policy - Provides standard rules
-E or -rename - Provides a rename
-h or -help - Displays the help utility

**PARAMETER** = The iptables parameters are used as a necessary complement for the purposes of the rule specification

-p or -protocol - This parameter specifies the treatment of the protocol.
-m or -match-option - similar to -p but the module used and are free to specify the name of the module used and vary it in subsequent commands.
-s or -source - hostname/Ip address.
-d or -destination - etc Parameters to determine the destination of the packet.
-j or -jump - gives a decision after the data packet is checked by the rules.
-i or -in-interface - Mask through interface (eth0, eth1 etc).
-o or -out-interface - alias name of the interface that will send the packet out (on chain FORWARD or OUTPUT and POSTROUTING).
-c or -counter - to count packets passing from a rule.
-n or -numeric - returns a numeric output such as hostname or Ip or port or name network.
-v or -verbose - which means to display the information as a whole.

**TARGET** = Target is the purpose of treating the rule. At this target lies the decision, what to do with data packets, whether to be rejected, or forwarded or processed first.

ACCEPT - The packet chain is accepted in the rule
DROP - The packet chain is "dropped"
REJECT - The packet chain is rejected as DROP
DNAT - Chain packets at "destination Nat" to another address
SNAT - The packet chain points to a specific source Nat
REDIRECT - The packet chain is redirected to a specified address and port
MASQUERADE - Works like SNAT but doesn't require a source
REJECT - Works like DROP