Firewalls

History

A firewall establishes a barrier between a trusted, secure internal network and another network (e.g., the Internet) that is assumed not to be secure and trusted. (from "Firewall (computing)" on Wikipedia).

The term "firewall" is old and refers to parts of a structure intended to protect other parts from fire, such as walls surrounding a kitchen.

In the late 1980's, network packets were used for purposes that caused damage, whether intentionally or not.
One such event was the release of the Morris Worm, which caused a Denial of Service (DOS) attack against a number of Unix hosts.

Literature on firewall design was first published in 1988 by Digital Equipment Corporation (DEC).

Today, firewalls are considered a necessity for any host connected to an external network.

Packet filtering

The original firewalls were simple packet filters.

A packet filter is software that governs whether to accept or reject packets using a set of rules.

Packet filter example: iptables

iptables is a native Linux program that is used to manage firewall rules. As one would expect, it is a command line tool.

There are other command line tools as well as GUI based firewall management tools for other operating systems, such as Windows, and they follow the same basic approach for rule management.

Version 1.4.21 contains five (5) built-in tables:

1) filter -- the default table; used for all traffic, generally
2) nat -- consulted when a new connection is attempted; used to reroute packets
3) mangle -- used for all other custom packet alterations
4) raw -- used to exempt certain packet types from connection tracking
5) security -- called after the filter table to implement Mandatory Access Control networking rules, which are used on SELinux

You can add custom tables by adding modules to the kernel, but this practice is unnecessary.

Each table consists of a series of rules. The rules are grouped into chains; therefore, each table contains a set of chains, and the chains themselves contain the rules.

filter table

- default table used for iptables commands when a table is not specified
- it is used to monitor ordinary network traffic
- it contains three chains (you can add custom chains)
    - INPUT - packets directed toward programs running on the host
    - OUTPUT - packets originating from a program on the host
    - FORWARD - packets to be forwarded, such as when the host is a router

Creating rules

- A rule consists of a set of criteria for the chain to try to match
- If a match is made, then the action specified in the rule is taken
- If the action is to jump to another chain, then the processing continues, although control will return to the calling chain if no match occurs
- If the action is to jump to a target that is not a chain, then the action designated for that target is taken and processing stops
- the builtin targets include:
    - ACCEPT -- allow the packet to continue on its way
    - DROP -- die, packet, die!
    - REJECT -- sends an error message back to the sender and then drops the packet
    - RETURN -- return immediately from the current chain to the calling chain

Each chain has a policy, which sets the default target to use when no match is made.

Note that default policy is ACCEPT, which is typical.

Parameters to match generally include:

- source and destination IP addresses (-s and -d)
- protocols, such as TCP, UDP, ICMP, etc. (-p [protocol])
- network interfaces (-i and -o, for input and output interfaces, respectively)

Extensions permit a wide variety and greater granularity of control over matching. TCP extensions include:

> --destination-port, --dport : destination port
>
> --source-port, --sport : source port
>
> --tcp-flags : SYN, ACK, FIN, RST, etc.
>
> --syn : when only the SYN bit is set; used to detect packets are attempting to establish a TCP connection
>
> --tcp-option : matches an option number

Examples of matching criteria

-s 192.168.10.1 indicates that the source host with address 192.168.10.1 is a match

-d 192.168.10.0/24 indicates that all destination hosts in the range match

-p tcp --dport 21 would match TCP traffic with a destination port of 21 (FTP)

-d 10.10.10.10 -p tcp --dport SSH would match TCP traffic directed to the SSH port (22) on 10.10.10.10

Commands

-L or --list : list the chains and rules in a table

-A or --append : add the rule at the end of a chain

-D or --delete : delete a rule

-I or --insert : insert a rule at a position within the chain

-R or --replace : replace a rule

-F or --flush : delete all rules from a chain

-N or --new-chain

-P or --policy : set the target for the chain's policy

Targets are usually specified using -j or --jump

Examples of iptables commands

iptables -L -n --line-numbers (displays the contents of the filter table, using numeric values and displaying line numbers)

iptables -S (displays the actual rules)

iptables -P FORWARD DROP (changes the policy for the FORWARD chain)

iptables -A INPUT -s 10.0.0.0/24 -j DROP (drops all inbound packets for the private IP address range 10.0.0.0/24)

iptables -A INPUT -s 10.0.0.0/24 -p tcp --dport 22 -j ACCEPT (permits all inbound packets from the private IP address range 10.0.0.0/24 that have a destination port of 22, which is SSH)

iptables -D INPUT 2

iptables -I INPUT 1 -s 10.0.0.0/24 -p tcp --dport 22 -j ACCEPT (permits all inbound packets from the private IP address range 10.0.0.0/24 that have a destination port of 22, which is SSH)

iptables -A FORWARD -p tcp --dport 22 -j DROP(prevents routing SSH packets in one direction)
iptables -A FORWARD -p tcp --sport 22 -j DROP(prevents routing SSH packets in the other direction)

Advancements

Stateful filters -- An enhanced version of a packet filter. Stateful filters track traffic in order to determine whether packets are being used to initiate connections, continue established communications, or simply do not belong.

Application layer filters -- An ever more enhanced type of filter, capable of what is also known as "deep packet inspection." Such filters are readily capable of obviating Net Neutrality, yet they can provide the finest level of granularity for regulating desirable traffic.

Firewalls can also exist at the network level.

A proxy server, used to communicate on behalf of a client host, can also be used as a firewall.

Some people consider NAT to be a technology that encompasses a firewall, but that claim is not accurate.