Contents

1 Firewalls

- A hardware and/or software system
- Prevents unauthorised access of packets from one network to another
- **All data leaving any subnet must pass through it**

1.1 Firewall Functions

- Implements single point security measures
- Security event monitoring through packet analysis and logging
- Network-based access control through implementation of a rule set

2 Location

- Network Firewalls: Placed between a subnet and the internet
- Host-based Firewalls: Placed on individual machines
- A standard home router is a good example of a *network firewall*
- A *demilitarized zone* is a small subnet that **separates externally facing services** from the internal network

3 Firewall Basic Function

- Defends a network against parties accessing internal services
- Can also restrict access from **inside to outside services** (e.g. IRC, P2P)
- Network Address Translation: **hides the internal machines** with private addresses

3.1 Firewalls are not enough

- Cannot protect against attacks that **bypass the firewall**. E.g. Tunnelling
- Cannot protect against **internal threats** or insiders. Might help a bit by egress filtering
- Network firewalls cannot always **protect against the transfer** of virus-infected programs or files

4 Packet Filters

- Specify which packets are allowed or dropped
- Rules based on: source / destination IP, TCP / UDP port numbers
- Possible for both inbound and outbound traffic
- Can be implemented **in a router** by only examining packet headers (IP / TCP)

4.1 Packet Filter Rules

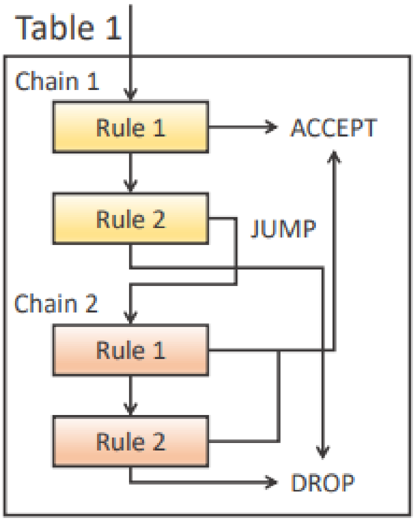
- Rule execution depends on implementation
  - IPTABLES: First rule to match is applied
  - PF: All rules are examined, the last match is applied
- Rules are organised in chains, which are logical subgroups of rules
- Depending on the packet, different chains are activated

4.2 IPTABLES

- An application that provides access to the Linux firewall rule tables
- **Not actually a firewall**, but configures the firewall
- The firewall is mostly implemented as *netfilter* modules

4.3 Tables and Chains

- IPTABLES uses tables to store chains.Default is the filtering table
- Chains are ordered lists of rules. Rules match, or they dont
- Matches result in a **jump**, else we check the next rule
- There can be multiple chains per table. E.g. a TCP handling chain
- Jumps can go to ACCEPT, DROP, LOG or another chain
- Complex behaviour can be built up



4.4 Defaults

- There are four built-in tables in IPTABLES: **Filter, NAT, Mangle Packet alteration, Raw Skips connection tracking**
- The default table is the **filtering table**, including Input, Output and Forward chains

4.5 Example

- Using the command line, we add rules onto the end of chains
- `iptables -A INPUT -i eth0 -p tcp --dport 80 -j ACCEPT`
- `iptables -A OUTPUT -o eth0 -p tcp --sport 80 -j ACCEPT`

4.6 Policies

- Permissive (Black listing) allow everything **except dangerous services**. Easy to make a mistake or forget something
- Restrictive (White listing) block everything except designated useful services
  - More secure by default
  - Fairly easy to DoS yourself!
- To use a blacklisting policy, we want to accept by default, then have rules that drop:
  - iptables -P INPUT ACCEPT
  - iptables -P FORWARD ACCEPT
  - iptables -P OUTPUT ACCEPT
  - iptables -A INPUT -s X.X.X.X -j DROP
  - iptables -A OUTPUT -p tcp –dport ssh -j DROP
- For a whitelisting policy, we want to drop by default, then let certain packets through:
  - iptables -P INPUT DROP
  - iptables -P FORWARD DROP
  - iptables -P OUTPUT DROP
  - iptables -A INPUT -p tcp –dport ssh -j ACCEPT
  - iptables -A OUTPUT -s 192.168.0.2 -j ACCEPT

5 Packet Filter Issues

- Packet filters are simple, low level and have high assurance
- **But, they cannot:**
  - Prevent attacks that employ **application-specific vulnerabilities**
  - Do not support higher-level **authentication schemes**
  - Easy to **accidentally allow or deny** packets incorrectly

5.1 Stateful Packet Filters

- Understand requests and replies (e.g. ACK/SYN)
- **Dynamically generate rules**
- E.g. FTP client, connect to 21, receive from 20
- Can support policies for a wider range or protocols

5.2 IPTABLES Rules

- IPTABLES has modules for stateful packet filtering
- Allow incoming / outgoing SSH connections
  - iptables -A INPUT -i eth0 -p tcp –dport 22 -m state –state NEW,ESTABLISHED -j ACCEPT
  - iptables -A OUTPUT -o eth0 -p tcp –sport 22 -m state –state ESTABLISHED j ACCEPT
- Allow HTTP(S):
  - iptables -A INPUT -i eth0 -p tcp –dport 80 -m state –state NEW,ESTABLISHED -j ACCEPT
  - iptables -A OUTPUT -o eth0 -p tcp –sport 80 -m state –state ESTABLISHED -j ACCEPT
  - iptables -A INPUT -i eth0 -p tcp –dport 443 -m state –state NEW,ESTABLISHED -j ACCEPT
  - iptables -A OUTPUT -o eth0 -p tcp –sport 443 -m state –state ESTABLISHED -j ACCEPT

6 Application-level Gateways

Reference section

placeholder