Netfilter / iptables

- Every packet is inspected by firewall rules.
- The iptables firewall **uses tables** to organize its rules.
- Within each iptables table, rules are further organized within separate "chains". **Rules are placed within a specific chain of a specific table.**
- Within a chain, a packet starts at the top of the chain and is matched rule by rule.
- When a match is found **the target is executed.**
- **A target is the action that is triggered** when a packet meets the matching criteria of a rule. If the target is terminating no other rule will evaluate the packet.

CHAINS

1. INPUT - used for filtering **incoming packets**. Our host is the packet destination

2. OUTPUT - used for filtering **outgoing packets**. Our host is the source of the packet

3. FORWARD - used for filtering routed packets. Our host is router.

4. PREROUTING - used for DNAT/PortForwarding

5. POSTROUTING - used for SNAT

Tables

1. filter
- filter is the default table for iptables.
- iptables filter table has the following built-in chains: INPUT, OUTPUT and FORWARD

2. nat
- nat table is specialized for SNAT and DNAT (Port Forwarding)
- iptables NAT table has the following built-in chains: PREROUTING, POSTROUTING and OUTPUT (for locally generated packets)

3. mangle
- iptables mangle table is specialized for packet alteration
- mangle table has the following built-in chains: PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING

4. raw
- The raw table is only used to set a mark on packets that should not be handled by the connection tracking system. This is done by using the NOTRACK target on the packet.
- raw table has the following built-in chains: PREROUTING and OUTPUT

User-defined CHAINS

- By default, the *iptables* filter table consists of three chains: **INPUT**, **OUTPUT** and **FORWARD**
- You can add as many custom chains as you like to help simplify managing large rule sets.
- User-defined chains are useful in optimizing the ruleset. They allow the rules to be organized in categories.
- Custom chains behave just as built-in chains, introducing logic that must be passed before the ultimate fate of a packet is determined.
- From a built-in chain using -j CUSTOM-CHAIN, you can jump into any number of custom defined chains and even jump between them.
- After the user-defined chain is traversed, control returns to the calling built-in chain, and matching continues from the next rule in the calling chain, unless the user-defined chain matched and took a terminating action on the packet.

User-defined CHAINS (cont)

- The RETURN jump at the end of the custom-chain makes processing resume back in the chain that called this one.
  -j RETURN can also be used inside a built-in chain. In this case no other rule will be inspected and packet executes the default POLICY
- If you want to stop using your custom chain temporarily, you can simply **delete the jump from the INPUT chain** (rather than flushing the entire custom chain)

Options:
-N NEW_CHAIN creates a new user-defined chain
-L NEW_CHAIN lists the content of the chain
-X NEW_CHAIN deletes the custom-chain (it must be emptied before using **-F**)
-F NEW-CHAIN flushes all rules from the chain

In a nutshell

- **Incoming traffic** is filtered on the **INPUT CHAIN** of the filter table
- **Outgoing traffic** is filtered on the **OUTPUT CHAIN** of the filter table
- **Routed traffic** is filtered on the **FORWARD CHAIN** of the filter table
- **SNAT/MASQUERADE** is done on the **POSTROUTING CHAIN** of the nat table
- **DNAT/PortForwarding** is done on the **PREROUTING CHAIN** of the nat table
- To modify values from the packet headers we add rules to the **mangle table**
- To skip connection tracking we add rules with **NOTRACK target** to the **raw table**