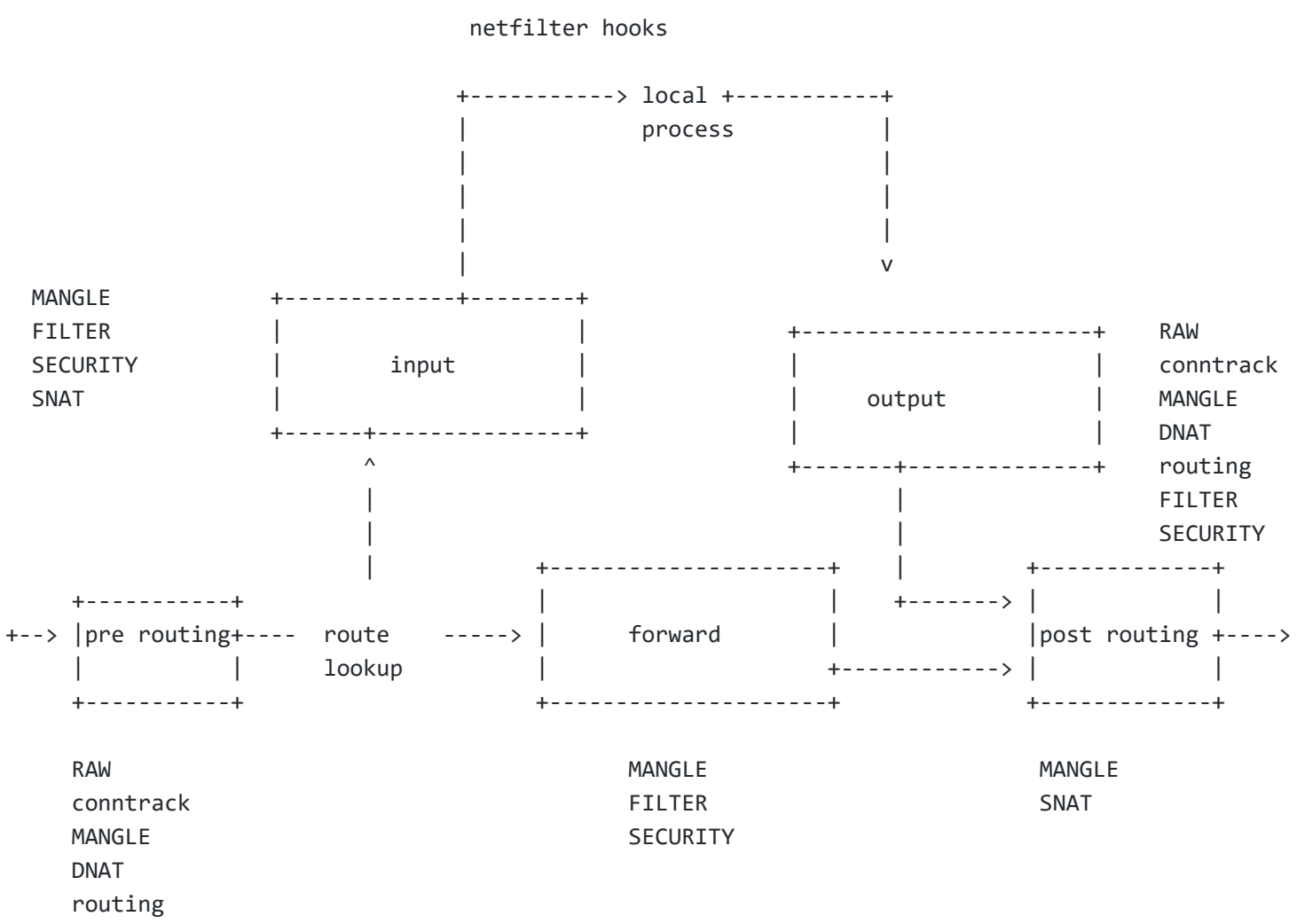
The netfilter hooks in the kernel and where they hook in the packet flow

The figure below calls out

- The netfilter hooks
- The order of table traversal



- Incoming packets destined for the local system: PREROUTING -> INPUT
- Incoming packets destined to another host: PREROUTING -> FORWARD -> POSTROUTING
- Locally generated packets: OUTPUT -> POSTROUTING

Tables

- The iptables firewall uses tables to organize its rules
- These tables classify rules according to the type of decisions they are used to make

Chains

- Within each iptables table, rules are further organized within separate "chains"
- Chains map to netfilter hooks

Different Tables

- filter: Do not modify traffic. Mostly used for firewalling
- nat:
- mangle: used to modify or mark packets: Mark is on the skbuf and not on the packet itself
- raw: used to help skip conntrack
- security used by selinux

Order of Chain evaluation across tables

- raw : Used to bypass connection tracking
- (connection tracking enabled)
- mangle
- nat (DNAT)
- (routing decision)
- filter
- security
- nat (SNAT)

IPTables Rules

- Rules are placed within a specific chain of a specific table
- Note: The table determines order of evaluation
- A target is the action that are triggered when a packet meets the matching criteria of a rule.

Targets

- Terminating targets: Terminating targets perform an action which terminates evaluation within the chain and returns control to the netfilter hook
- Non-terminating targets: Non-terminating targets perform an action and continue evaluation within the chain
- special class of non-terminating target: the jump target

User-Defined Chains (sub chain)

- user-defined chains can only be reached by "jumping" to them from a rule via the jump target
- and they can jump to other chains

```
iptables -N <chain name>
iptables -A INPUT -p tcp -j <chain name>
```

- if a packet is ACCEPTed within one of the sub chains, it will be ACCEPT'ed in the superset chain also and it will not traverse any of the superset chains any further (in that table!). However, do note that the packet will traverse all other chains in the other tables in a normal fashion.

Targets

- -j RETURN: will cause the current packet to stop traveling through the chain (or sub-chain)
- -j ACCEPT : the rule is accepted and will not continue traversing the current chain or any other ones in the same table. Note however, that a packet that was accepted in one chain might still travel through chains within other tables, and could still be dropped there
- -j DNAT : only available within PREROUTING and OUTPUT chains in the nat table, and any of the chains called upon from any of those listed chains
- -j SNAT: valid only in nat table, within the POSTROUTING chain
- -j DROP: Drops the packet, right there right then
- -j REJECT: Sends a response back (unlike drop). Valid in the INPUT, FORWARD and OUTPUT chains or their sub chains
- -j LOG: Note: Does not work on namespaces. Also can fill up your kernel log.

```
iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"
```

- -j ULOG: packet information is multicasted together with the whole packet through a netlink socket. One or more user-space processes may then subscribe to various multicast groups and receive the packet
- -j MARK: Only valid in mangle table. Note that the mark value is not set within the actual package, but is a value that is associated within the kernel with the packet. In other words does not make it out of the machine

```
iptables -t mangle -A PREROUTING -p tcp --dport 22 -j MARK --set-mark 2
```

- -j MASQUERADE: Similar to SNAT but used on a outbound network *interface* when the outbound IP can change. Say a DHCP interface Only valid within the POSTROUTING
- -j REDIRECT: redirect packets and streams to the machine itself. Valid within the PREROUTING and OUTPUT chains of the nat table. It is also valid within user-defined chains that are only called from those chains

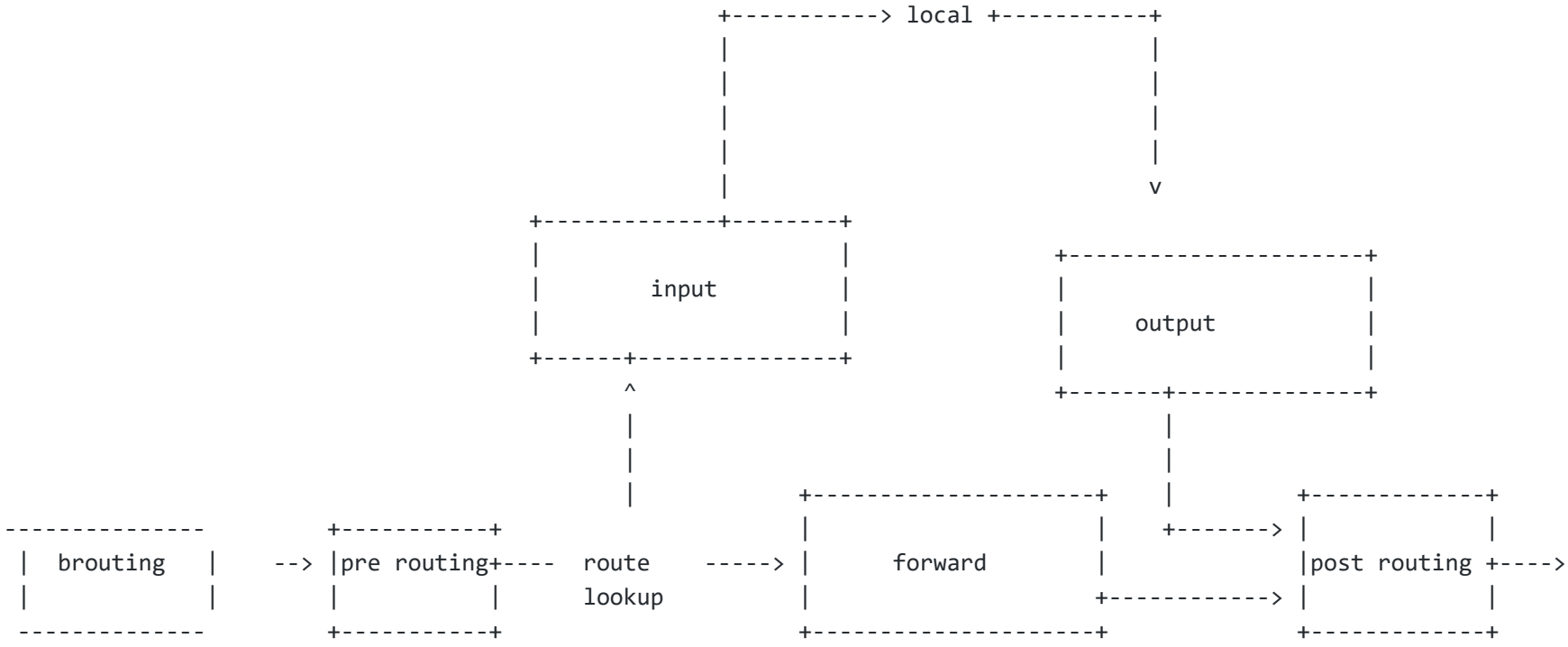
Modules

- iptables can use extended packet matching modules with the -m or --match options, followed by the matching module name
- Some important ones
  - connmark [!] --mark value[/mask] Matches packets in connections with the given mark value (if a mask is specified, this is logically ANDed with the mark before the comparison).
  - conntrack
  - ipvs
  - mark
  - redirect This target is only valid in the nat table, in the PREROUTING and OUTPUT chains, and user-defined chains which are only called from those chains. It redirects the packet to the machine itself by changing the destination IP to the primary address of the incoming interface (locally-generated packets are mapped to the localhost address, 127.0.0.1 for IPv4 and ::1 for IPv6).

References

- https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture
- https://www.netfilter.org/documentation/HOWTO/netfilter-hacking-HOWTO-3.html
- https://www.frozentux.net/iptables-tutorial/chunkyhtml/c3965.html
- http://ipset.netfilter.org/iptables-extensions.man.html
- http://ebtables.netfilter.org/br_fw_ia/br_fw_ia.html

Bridge Filters



- The broute table has the BROUTING chain.
- The filter table has the FORWARD, INPUT and OUTPUT chains.
- The nat table has the PREROUTING, OUTPUT and POSTROUTING chains.