Firewalls with iptables

Linux

Sirindhorn International Institute of Technology
Thammasat University

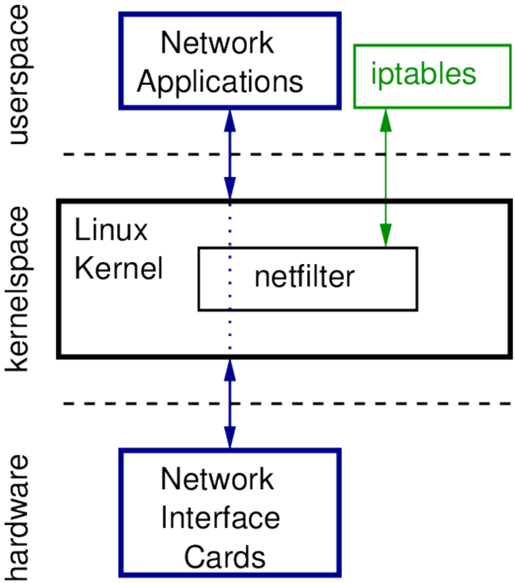
Prepared by Steven Gordon on 14 October 2013
Common/Reports/iptables-introduction.tex, r715

Contents

Linux, netfilter and iptables

- `netfilter`: module for filtering packets in Linux kernel
- `iptables`: user space application to manipulate packet filters of `netfilter`
- Administrator privileges needed for manipulating kernel packet filters
  - Prefix `iptables` commands with `sudo`

Linux, netfilter and iptables



iptables Concepts: Tables

- Different tables of filters (depend on kernel configuration)
- Selected using `-t` option
  - `filter`: default table (if no option used)
  - `nat`: Network Address Translation
  - `mangle`: Altering packets
  - ...
- Tables contain chains

iptables Concepts: Chains

Different filtering rules depending on how/where packet processed by kernel

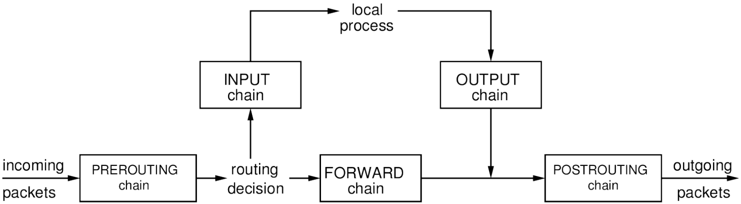
INPUT packets destined to this computer

OUTPUT packets originating from this computer

FORWARD packets being forwarded by this computer

PREROUTING altering packets as they come in to this computer (e.g. nat, mangle)

POSTROUTING altering packets as they go out of this computer (e.g. nat, mangle)



iptables Concepts: Rules

- Chains contain packet filtering rules
- Rules consist of:

  Matching condition(s) desired packet characteristics

  - protocol, source/dest. address, interface
  - many protocol specific extensions

  Target action to take if packet matches specified conditions

  - ACCEPT, DROP, RETURN, . . .
- A packet is checked against rules in chain, from 1st to last
- If rule does not match, check against next rule in chain
- If rule matches, take action as specified by target

Common iptables Syntax

iptables [-t *table*] [-*operation chain*] [-p *protocol*] [-s *srcip*] [-d *dstip*] [-i *inif*] [-o *outif*] [–param1 value1 . . .] -j *target*

- *table*: `filter`, `nat`, `mangle`
- *operation*: (first uppercase letter) `Append`, `Delete`, `Insert`, `List`, `Flush`, `Policy`, . . .
- *chain*: `INPUT`, `OUPTUT`, `FORWARD`, `PREROUTING`, `POSTROUTING`
- *protocol*: `tcp`, `udp`, `icmp`, `all`, . . .
- *srcip*, *dstip*: IP address, e.g. `1.1.1.1`, `2.2.2.0/24`
- *inif*, *outif*: interface name, e.g. `eth0`
- *param*, *value*: protocol specific parameter and value
  - `sport`, `dport`, `tcp-flags`, `icmp-type`, . . .
- *target*: `ACCEPT`, `DROP`, `RETURN`, . . .

`man iptables` to see detailed syntax and parameters

Contents

Concepts

Examples

Example 1: Drop ICMP Packets

Aim

Drop all ICMP packets sent by this computer

Design

- Assume default policy is ACCEPT
- Assume filter table empty → append a new rule
- Packets sent → OUTPUT chain
- Protocol is icmp
- Target is DROP

Implementation

```
iptables -A OUTPUT -p icmp -j DROP
```

Example 2: Allow Access Only to Web Server

Aim

Prevent others from sending to this computer, except to the local HTTP web server

Design

- Packets received → INPUT chain
- HTTP uses TCP → protocol is tcp
- Web server listens on port 80 → destination port 80
- Set the default policy to DROP
- Target is ACCEPT

Implementation

```
iptables -P INPUT DROP
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

Example 3: View Current Rules

Aim

List the current set of rules, showing actual addresses

Design

- Numeric addresses → `-n`

Implementation

```
iptables -L -n
```

```
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  0.0.0.0/0            0.0.0.0/0
```

Example 4: Delete All Previous Rules

Aim

Delete all (flush) the rules from the default filter table, and reset policy to default accept

Implementation

```
iptables -F
iptables -P INPUT ACCEPT
iptables -L
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Example 5: Block Packets Through Router

Aim

On this router, block all packets arriving on interface `eth0` and destined to subnet `2.2.2.0/24` (and then view the rules)

Design

- Packets forwarded through routers → FORWARD chain
- Verbose output needed to see interfaces → `-v`

Implementation

```
iptables -A FORWARD -i eth0 -d 2.2.2.0/24 -j DROP
iptables -L FORWARD -n -v
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in    out  source     destination
    0     0 DROP    all  --  eth0 *    0.0.0.0/0  2.2.2.0/24
```