Firewalls

Miguel Frade & Francisco Santos

Introduction

INTRODUCTION

Packet filtering firewall

- uses a list of rules based on matches to fields in the TCP/IP headers
- if there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet
- if there is no match to any rule, then a default action is taken

INTRODUCTION

Packet filtering firewall

- uses a list of rules based on matches to fields in the TCP/IP headers
- if there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet
- if there is no match to any rule, then a default action is taken

Two default policies are possible

- **deny** – what is not expressly permitted is prohibited
- **allow** – what is not expressly prohibited is permitted

INTRODUCTION

Examples of software packet filtering firewalls

- Linux
    - netfilter/iptables – pre-installed on all distros
    - nftables – too new, not standard yet
- Windows builtin firewall
- FreeBSD
    - pfSense – standalone firewall with graphical interface

INTRODUCTION

Examples of software packet filtering firewalls

- Linux
  - netfilter/iptables – pre-installed on all distros
  - nftables – too new, not standard yet
- Windows builtin firewall
- FreeBSD
  - pfSense – standalone firewall with graphical interface

In this class we'll learn

netfilter/iptables

INTRODUCTION

netfilter/iptables

- netfilter – kernel module responsible for the filtering
- iptables / ip6tables – commandline interfaces to configure the firewall
- iptables / ip6tables configuration
  - commandline
  - scripts
  - graphical tools

Basic IPtables configuration

IPTABLES

Lets create 2 scripts to configure iptables

```
user@linux:~$ touch firewall-on.sh
user@linux:~$ touch firewall-off.sh
```

Give execution permission

```
user@linux:~$ chmod +x firewall-*.sh
```

BASIC IPTABLES CONFIGURATION

IPtables has three base lists

- INPUT – list to apply rules to incoming packets
- OUTPUT – list to apply rules to outgoing packets
- FORWARD – list to apply rules to packets routed through the device

BASIC IPTABLES CONFIGURATION

IPtables has three base lists

- INPUT – list to apply rules to incoming packets
- OUTPUT – list to apply rules to outgoing packets
- FORWARD – list to apply rules to packets routed through the device

Define default policy

```
# what is not expressly permitted is prohibited
iptables -P <list_name> DROP

# what is not expressly prohibited is permitted
iptables -P <list_name> ACCEPT
```

BASIC IPTABLES CONFIGURATION

Edit `firewall-on.sh` and type

```
#!/bin/bash

# Define iptables full path
IPT=/sbin/iptables

echo "Set default policy to DENY"
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

echo "Flush any existing filter rules"
$IPT -F

# Type any filter rules after this line
```

BASIC IPTABLES CONFIGURATION

Edit **`firewall-on.sh`** and type

```
#!/bin/bash

# Define iptables full path
IPT=/sbin/iptables

echo "Set default policy to DENY"
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

echo "Flush any existing filter rules"
$IPT -F

# Type any filter rules after this line
```

Edit **`firewall-off.sh`** and type

```
#!/bin/bash

# Define iptables full path
IPT=/sbin/iptables

echo "Set default policy to ALLOW"
$IPT -P INPUT ACCEPT
$IPT -P OUTPUT ACCEPT
$IPT -P FORWARD ACCEPT

echo "Flush any existing filter rules"
$IPT -F

echo "The firewall is now OFF"
```

BASIC IPTABLES CONFIGURATION

Turn ON the firewall

```
user@linux:~$ sudo ./firewall-on.sh
Set default policy to DENY
Flush any existing filter rules
```

Test connectivity

```
user@linux:~$ ping localhost
```

Did `ping` got any answer?

BASIC IPTABLES CONFIGURATION

Add filtering rules at the end of `firewall-on.sh` to allow the loopback interface

```
echo "Allow loopback interface"
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT
```

-A	– append rule
-i lo	– loopback incoming
-o lo	– loopback outgoing
-j ACCEPT	– jump to ACCEPT

Test connectivity again

Terminal

```
user@linux:~$ sudo ./firewall-on.sh
Set default policy to DENY
Flush any existing filter rules
Allow loopback interface
user@linux:~$ ping localhost
```

BASIC IPTABLES CONFIGURATION

Test connectivity with the gateway

- get the gateway IP address

```
Terminal
user@linux:~$ ip r
default via 192.168.1.1 dev eth0 proto dhcp metric 100
...
```

- ping the gateway

```
Terminal
user@linux:~$ ping 192.168.1.1
```

- did `ping` got any answer?

BASIC IPTABLES CONFIGURATION

Add filtering rules at the end of `firewall-on.sh` to allow `ping`

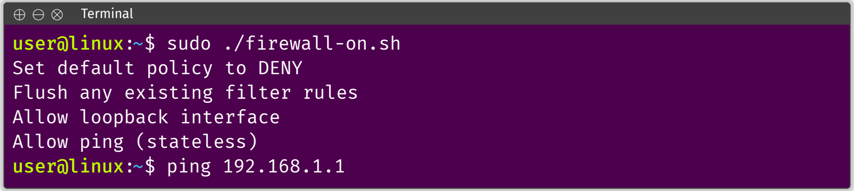
```
echo "Allow ping (stateless)"
$IPT -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

`-p icmp` – ICMP protocol

`--icmp-type` – specific ICMP packets

list options: **`iptables -p icmp -h`**

Apply the firewall rules and test `ping` again



Terminal

```
user@linux:~$ sudo ./firewall-on.sh
Set default policy to DENY
Flush any existing filter rules
Allow loopback interface
Allow ping (stateless)
user@linux:~$ ping 192.168.1.1
```

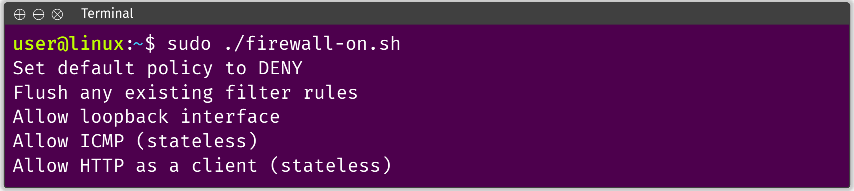
BASIC IPTABLES CONFIGURATION

Add filtering rules at the end of `firewall-on.sh` to allow HTTP as client

```
# Dynamic ports
DYN=1024-65535

echo "Allow HTTP as client (stateless)"
$IPT -A OUTPUT -p tcp --sport $DYN --dport http -j ACCEPT
$IPT -A INPUT -p tcp --sport $DYN --dport http -j ACCEPT
```

`-p tcp`	– TCP protocol
`--sport`	– source port
`--dport`	– destinatin port
`http`	– port number 80

Apply the firewall rules and test your browser



```
user@linux:~$ sudo ./firewall-on.sh
Set default policy to DENY
Flush any existing filter rules
Allow loopback interface
Allow ICMP (stateless)
Allow HTTP as a client (stateless)
```

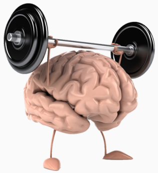
BASIC IPTABLES CONFIGURATION

Content of `firewall-on.sh` so far

```
#!/bin/bash

# Define iptables full path
IPT=/sbin/iptables

echo "Set default policy to DENY"
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

echo "Flush any existing filter rules"
$IPT -F

# Type any filter rules after this line

# (continues ->)
```

```
# (continuation)

echo "Allow loopback interface"
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

echo "Allow ping (stateless)"
$IPT -A OUTPUT -p icmp --icpm-type echo-request -j ACCEPT
$IPT -A INPUT -p icmp --icpm-type echo-reply -j ACCEPT

# Dynamic ports
DYN=1024-65535

echo "Allow HTTP as client (stateless)"
$IPT -A OUTPUT -p tcp --sport $DYN --dport http -j ACCEPT
$IPT -A INPUT -p tcp --sport $DYN --dport http -j ACCEPT
```

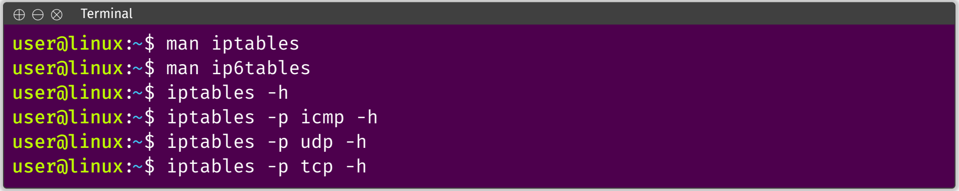
Exercises



1. add rules to the following protocols
   - DNS as client
   - HTTPs as client
   - reply `ping` as server
   - SSH as both client and server access
2. test your rules
   - open any web page on your browser
   - ask your colleague to turn off the firewall, then connect to the SSH server on his computer
   - then ask your colleague to:
     - `ping` your computer
     - connect to the SSH server on your computer

Questions?

BIBLIOGRAPHY



```
user@linux:~$ man iptables
user@linux:~$ man ip6tables
user@linux:~$ iptables -h
user@linux:~$ iptables -p icmp -h
user@linux:~$ iptables -p udp -h
user@linux:~$ iptables -p tcp -h
```