LAB8: Firewalls

Sven Relovský

11.04.2018

Introduction

e-mail contact: 409787@mail.muni.cz
You can use sli.do with #V284 to ask questions at any time (if you are too shy).

Introduction - Firewalls

- Network defense mechanism
- Inside and outside of network requires different levels of security
  - Firewall stationed at boundary
- Allows or denies packets according to specified policy/rule-set
- Allow/deny, inbound/outbound, symmetric/asymmetric
- Techniques:
  - Service control - what can be accessed
  - User control - who can use a particular service
  - Behavior control - how the service is used
  - Direction control - inbound and outbound traffic rules
- Types:
  - Packet filter
  - Stateful inspection
  - Application gateway

Stateful / Stateless FW

Stateless firewall

1. Easier to imagine
2. Doing exactly what the rules specified
3. Has to be configured for both sides
4. Does not fit real-world scenarios

Stateful firewall

1. Easier to set up
2. Has to be configured on a single side
3. Does fit real-world scenarios

Setup

1. Start VirtualBox, download files from O:\PA197\LAB8
2. Import fw-pa197.ova
3. Load virtual machines

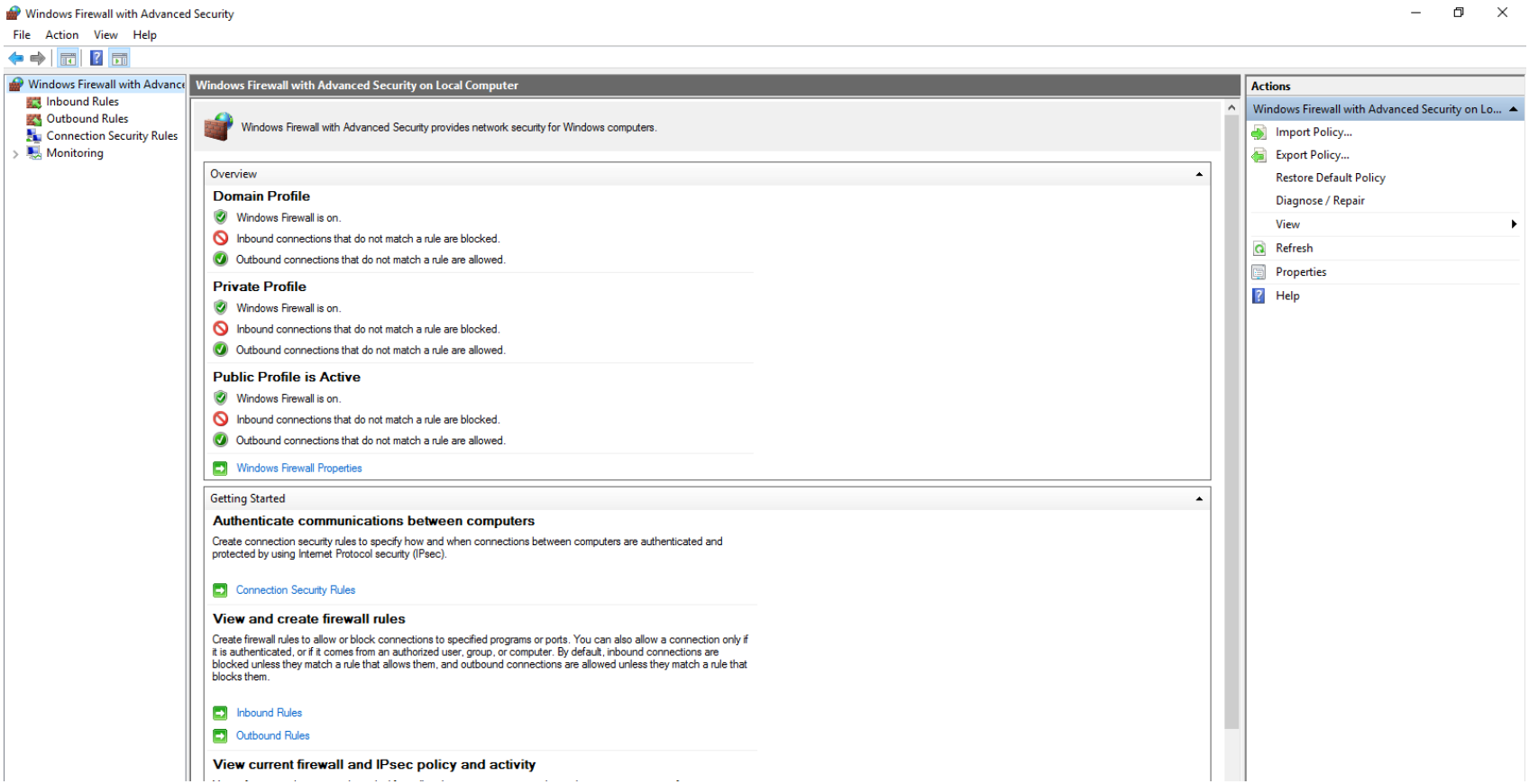
Nmap - Network Mapper

- Free, open source utility for network discovery and security auditing
- Scanning of the network
  - What hosts are available
  - What services are they offering
  - What operating systems
  - And more
- Useful for scanning single hosts, and even very large networks (100k+)
- Large support from developers and community
  - Well documented

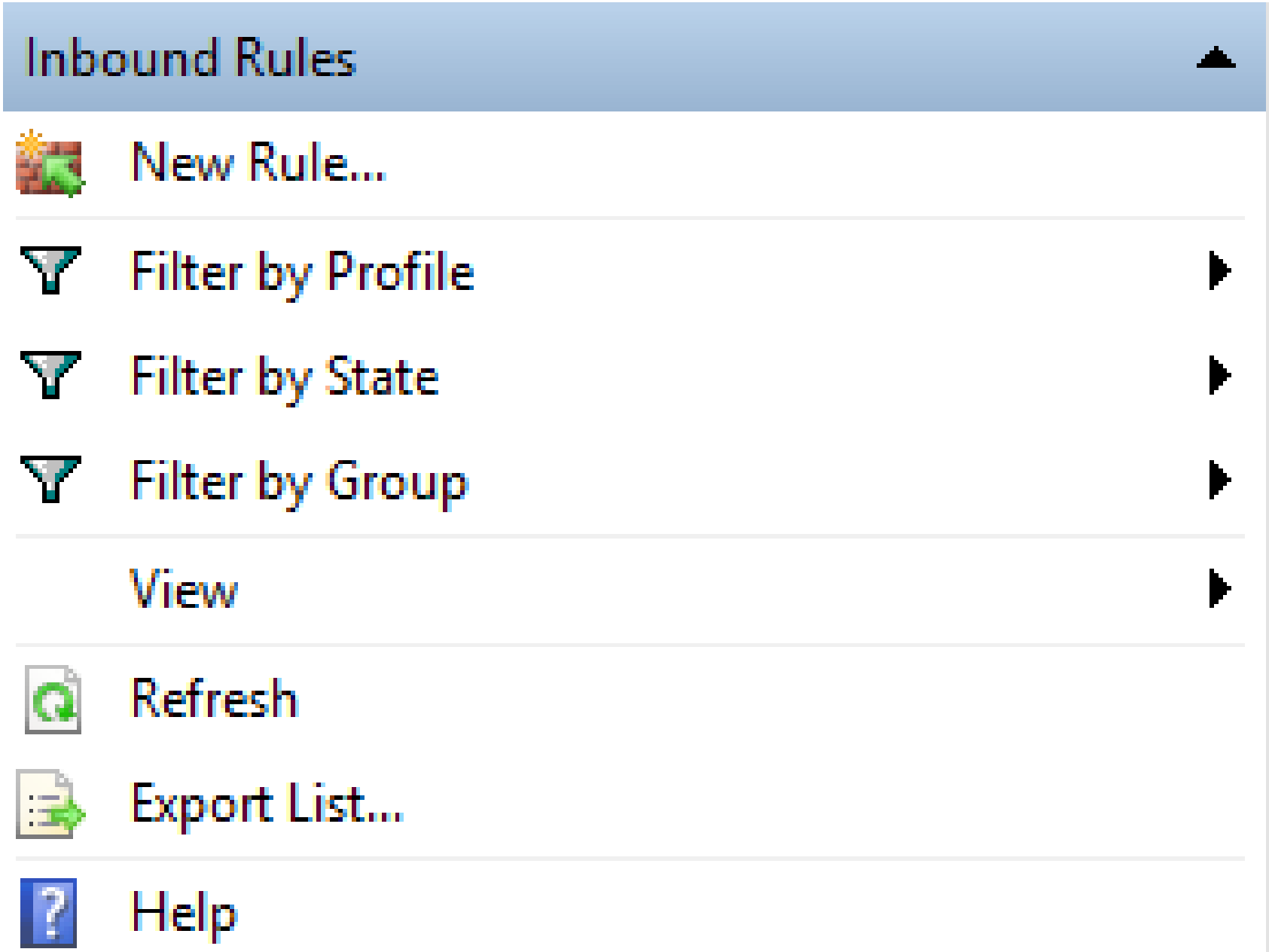
Part 1: Windows Firewall

- Introduced in 2001 for Windows XP Service Pack 2
  - Not capable of controlling outgoing connections
- Vista introduces multiple improvements
  - Outbound packet filtering
  - More advanced packet-filtering rules - destination IP, port range
  - IPsec integration - connections allowed or denied based on security certificate
- Windows 7 uses the same firewall as Vista
  - Minor improvements such as multiple active profiles

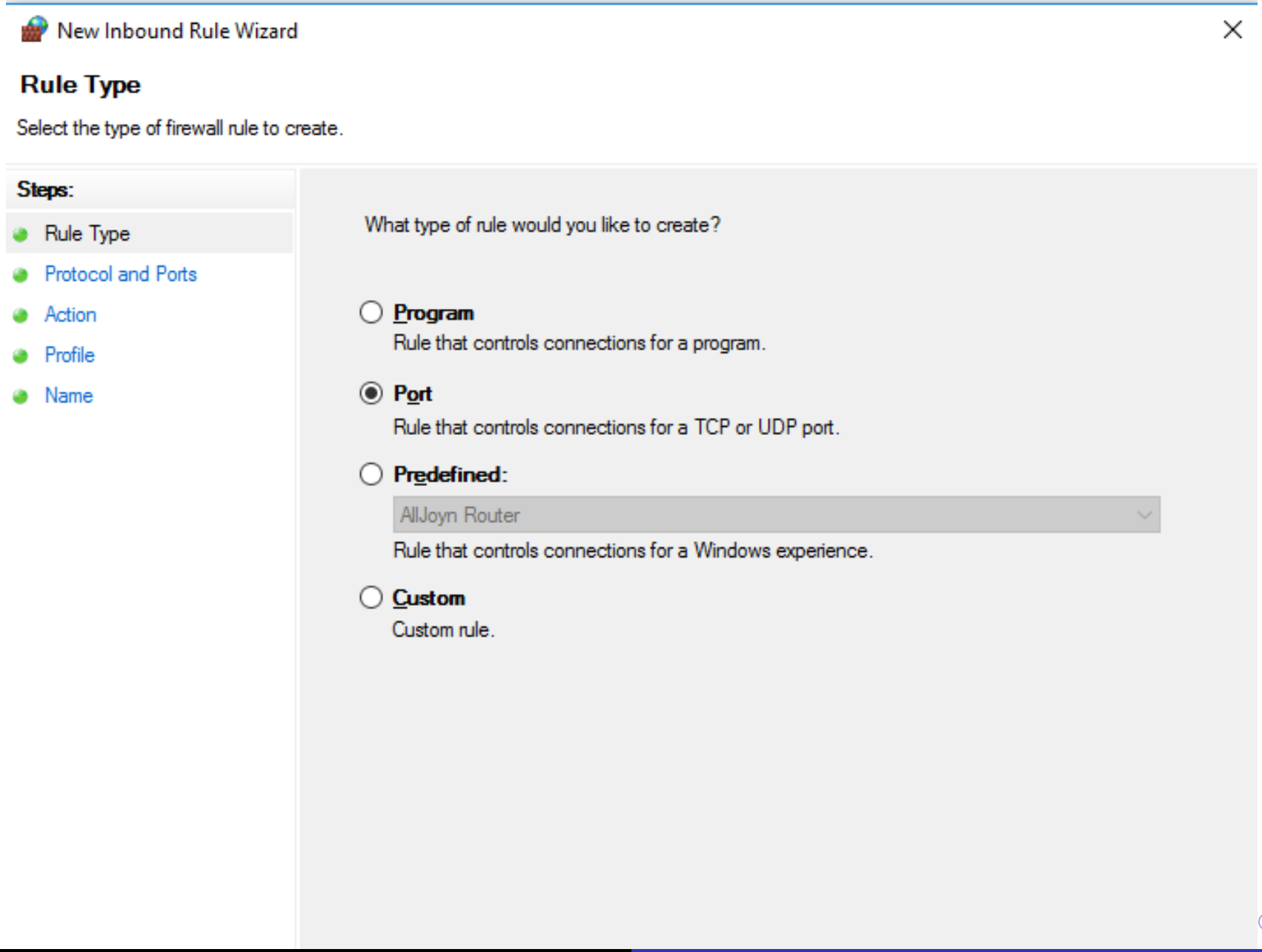
Windows Firewall



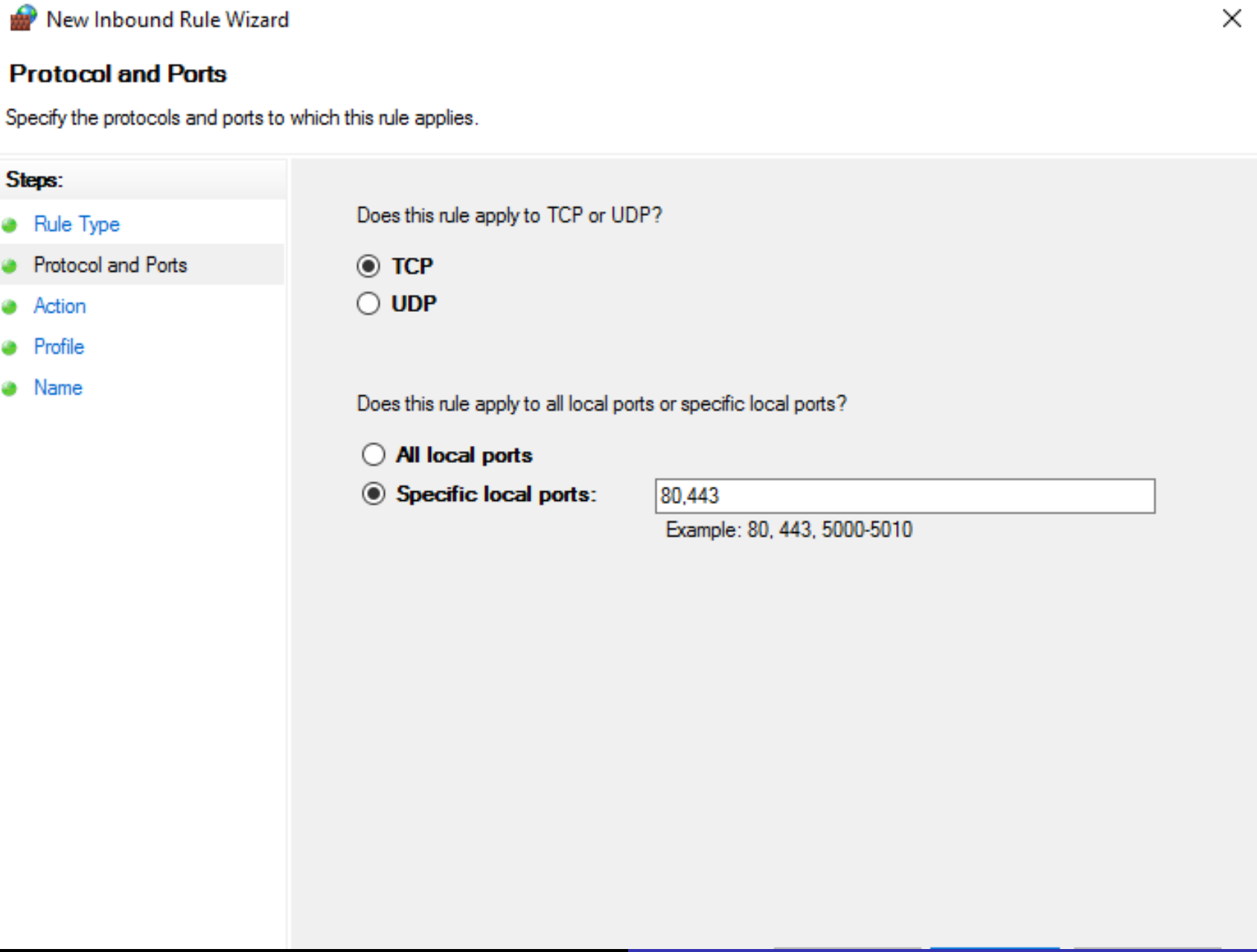
Windows Firewall



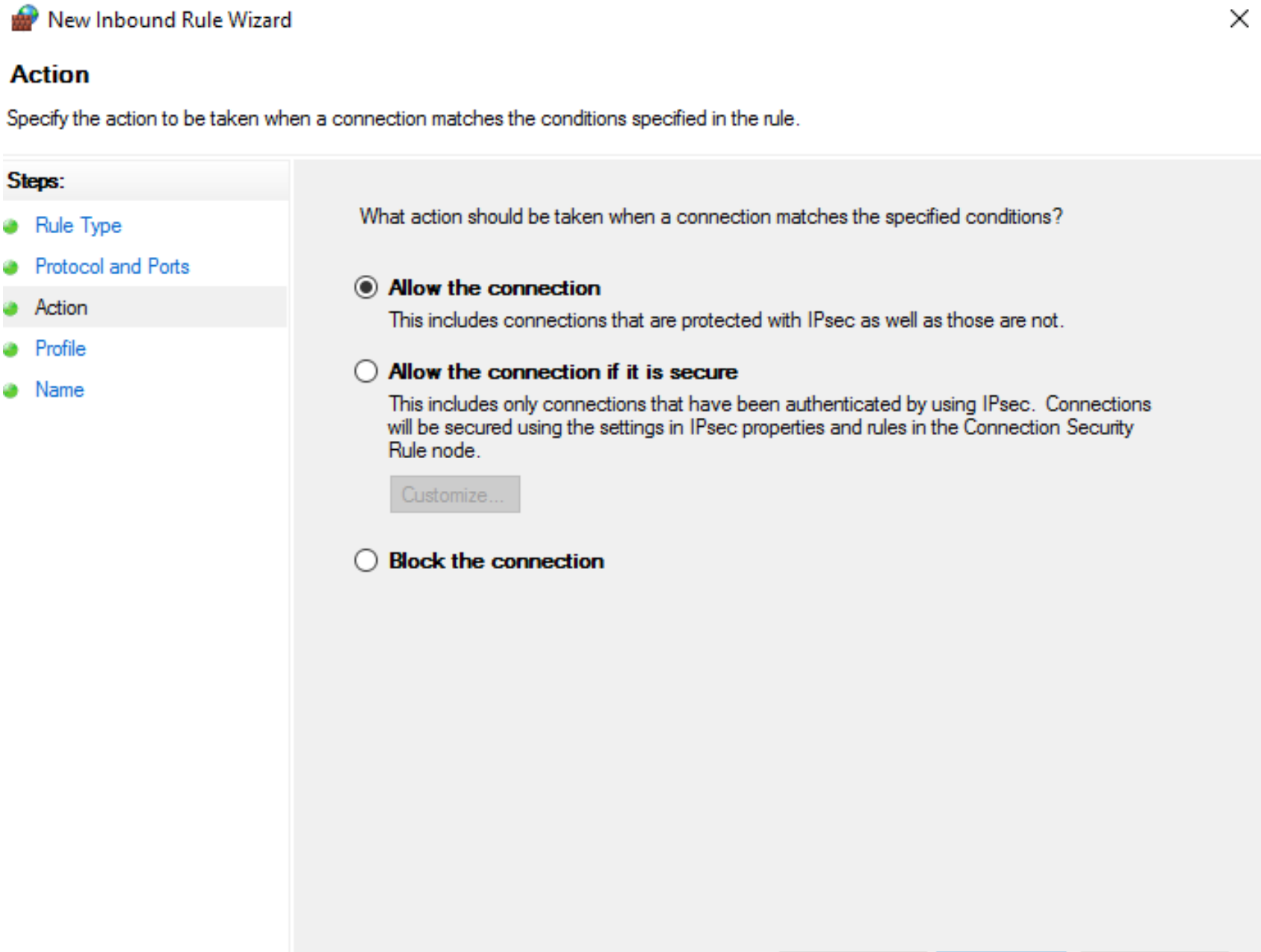
Windows Firewall



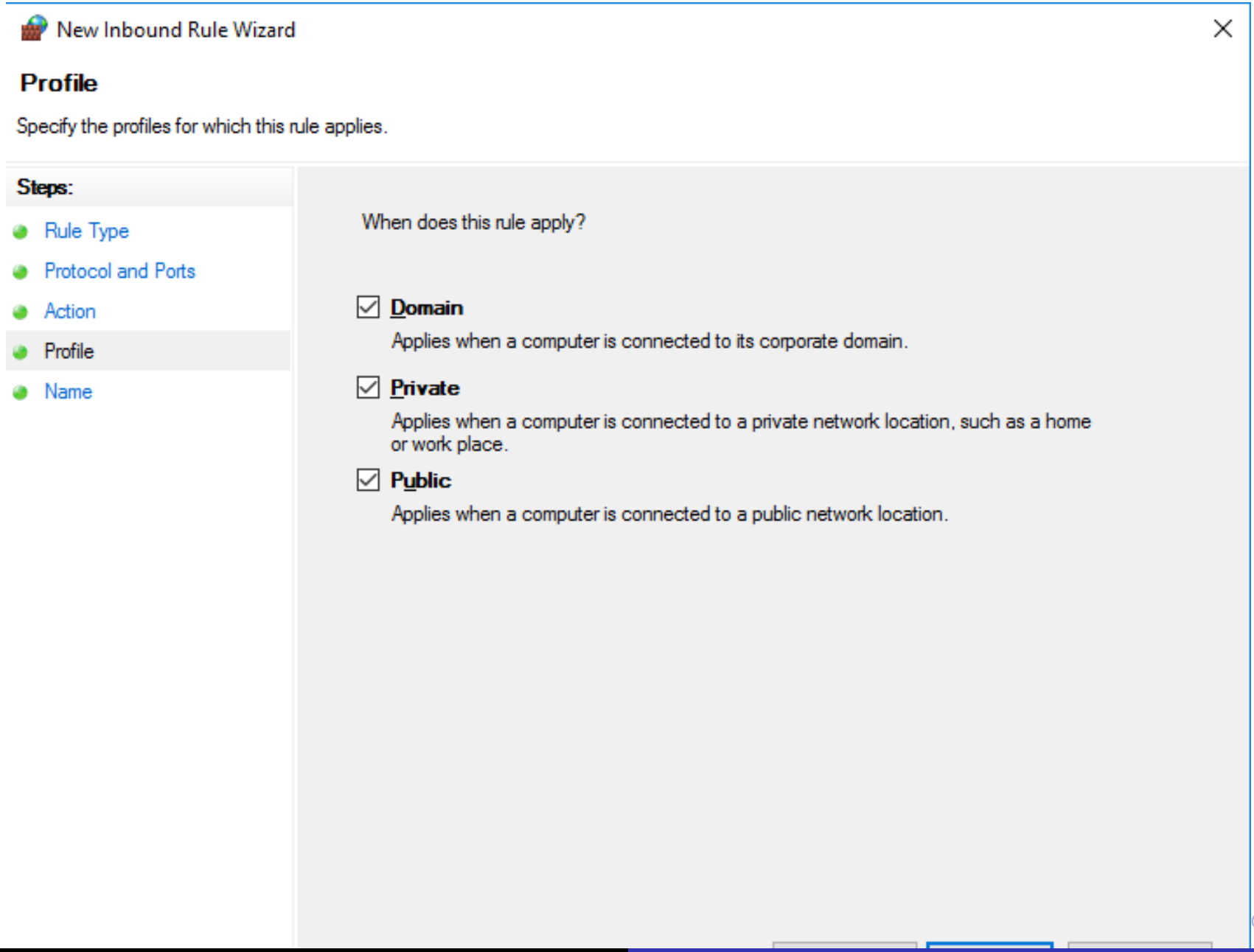
Windows Firewall



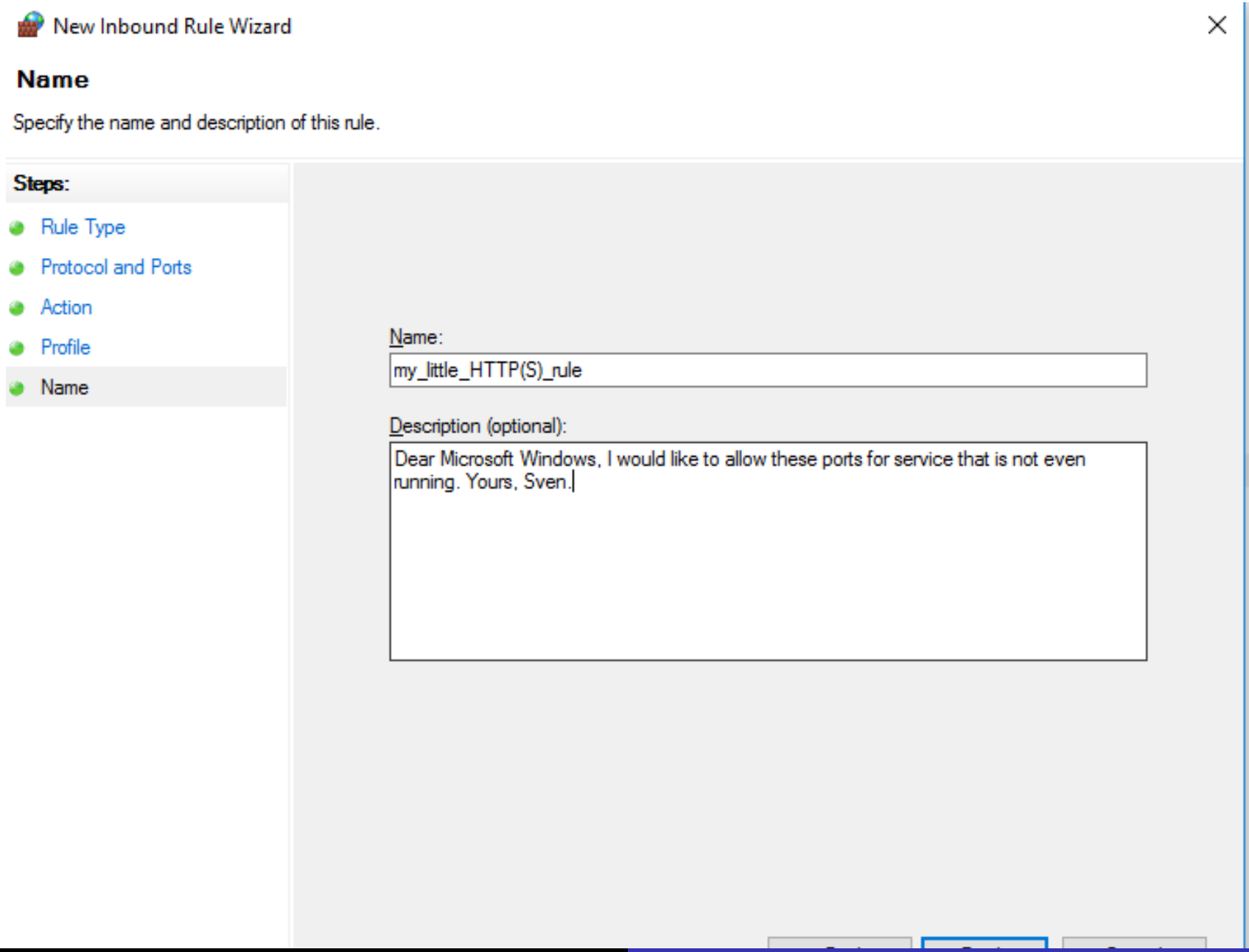
Windows Firewall



Windows Firewall



Windows Firewall



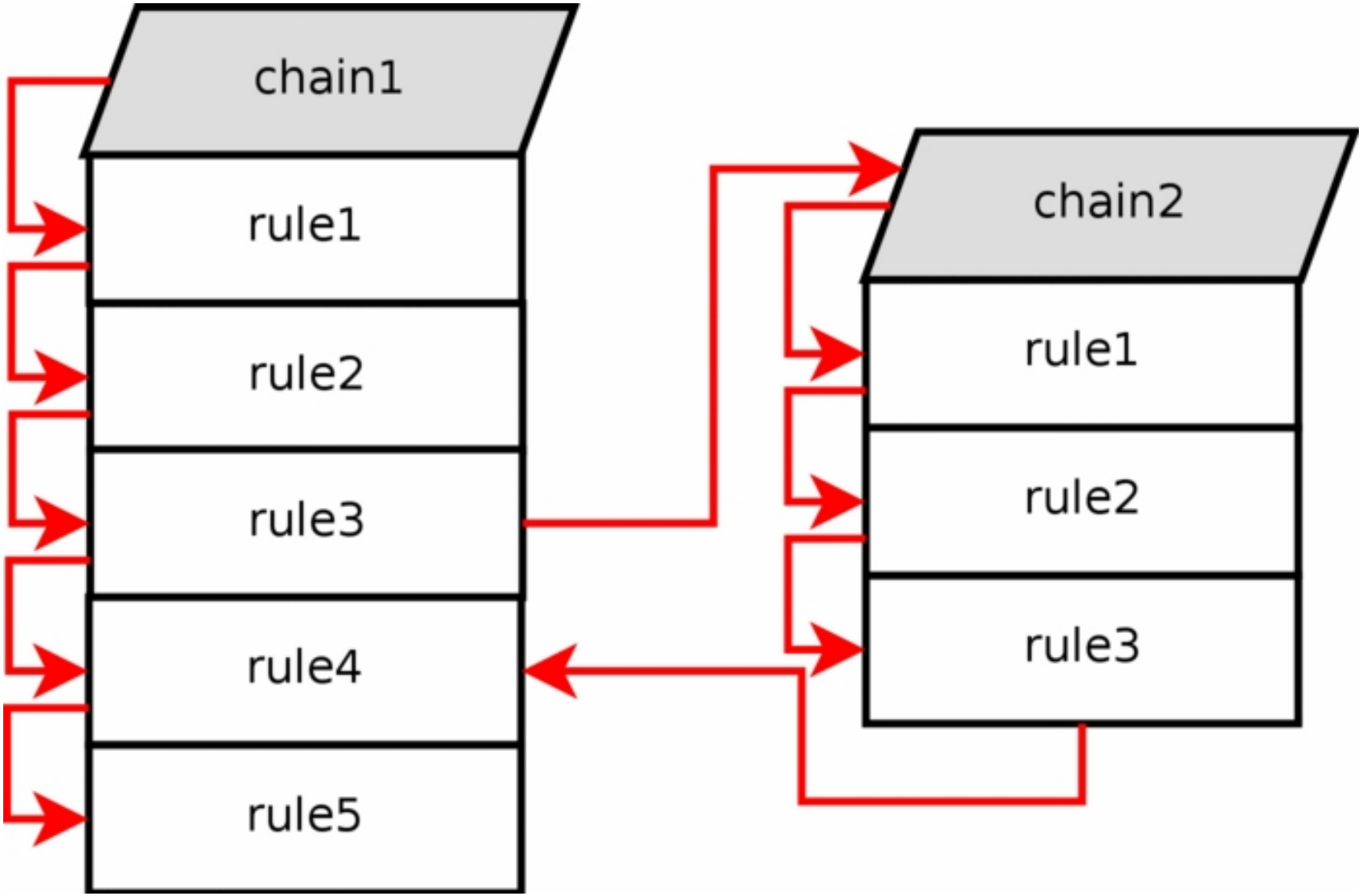
Windows Firewall

Name	Group	Profile	Enabled	Action	Override	Program	Local Address	Remote Address	Protocol	Local Port
my_little_HTTP(S)_rule		All	Yes	Allow	No	Any	Any	Any	TCP	80, 443
@{Microsoft.AAD.BrokerPlugin_1000.105...	@{Microsoft.AAD.BrokerPlu...	Domai...	Yes	Allow	No	Any	Any	Any	Any	Any
@{Microsoft.MicrosoftEdge_25.10586.0.0...	@{Microsoft.MicrosoftEdge...	Domai...	Yes	Allow	No	Any	Any	Any	Any	Any

Part 2: Linux IPtables

- Packet filtering, connection tracking, logging, NAT
- Administrator uses tables to define chains of rules for the treatment of packets
  - Packets assigned chain based on origin
  - Built in chains: INPUT (incoming packets), OUTPUT, FORWARD
- Packet filtering process
  1. Matching chain is selected
  2. Each rule in the chain is examined for a match
  3. If a match is found, the defined action of the rule is performed
  4. If no match is found the default chain policy is applied

Part 2: Linux IPtables



Part 2: Linux IPtables

- Listing rules

```
iptables -L
```

- Appending rules

```
iptables -A INPUT -p tcp --dport 23 -j ACCEPT
```

- Inserting rules
  - Block all communication

```
iptables -A INPUT -j DROP
```

  - Insert rule above previous one

```
iptables -I INPUT 1 -i lo -j ACCEPT
```

- Replace rule

```
iptables -R INPUT 3 -j ACCEPT
```

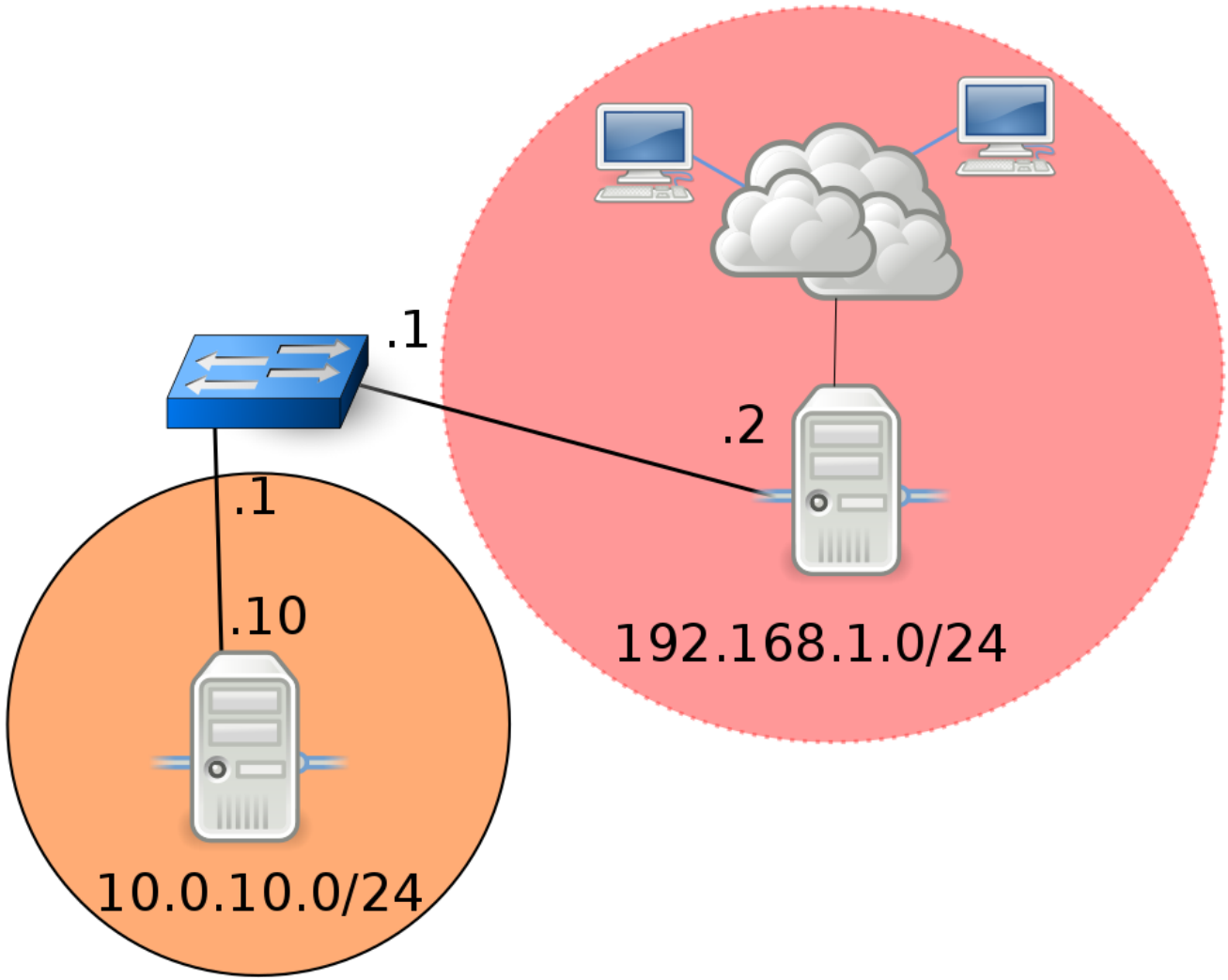
- Flush rules - clear IPtables

```
iptables -F
```

Part 2: Linux IPtables Task

Company has a simple server running few services. You as a network administrator were asked to secure the connection. You know that such server should not communicate on its own. You heard about using iptables as stateless firewall and it sounds pretty nice for the job.

Scenario



Part 2: Linux IPtables Task

Set up IPtables with the following conditions

`http://linux.die.net/man/8/iptables`

1. Create stateless rules on orange server, (src/dst address/port for INPUT and OUTPUT)
2. Reject other packets than listed here)
3. Accept SSH (Secure Shell) connection from source 10.0.0.0/24 and 192.168.1.0/24
4. Accept TCP communication through HTTP for all sources
5. Accept ICMP for any source and any destination
6. Accept FTP communication
   - Note: Which rules are applied first?
7. Test
   - nmap -sS 10.0.10.10

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

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

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 20 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 20 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 21 -j ACCEPT
```

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 20 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 21 -j ACCEPT
```

4. Reject other

Part 2: Linux IPtables Task

1. Accept SSH (Secure Shell - incoming)

```
iptables -A INPUT -p tcp -s 10.0.0.0/24 --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp -d 10.0.0.0/24 --sport 22 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j
ACCEPT
iptables -A OUTPUT -p tcp -d 192.168.1.0/24 --sport 22 -j
ACCEPT
```

2. Accept TCP communication through HTTP

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

3. Accept FTP communication

```
iptables -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 20 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 21 -j ACCEPT
```

4. Reject other

```
iptables -A [INPUT|OUTPUT] -j DROP
```

Part 2: Linux IPtables - stateful firewall rules

5. Accept established connections, and connections related to already allowed connections

```
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED
-j ACCEPT
```

6. Reject invalid packets

```
iptables -A INPUT -m conntrack --ctstate INVALID -j REJECT
```

7. Protect firewall against Brute Force SSH attacks, limiting incoming SSH requests to 4 per minute
   - 7.1 Create a set of incoming new SSH connections

```
iptables -I INPUT 3 -p tcp --dport 22 -m state
--state NEW -m recent --set
```

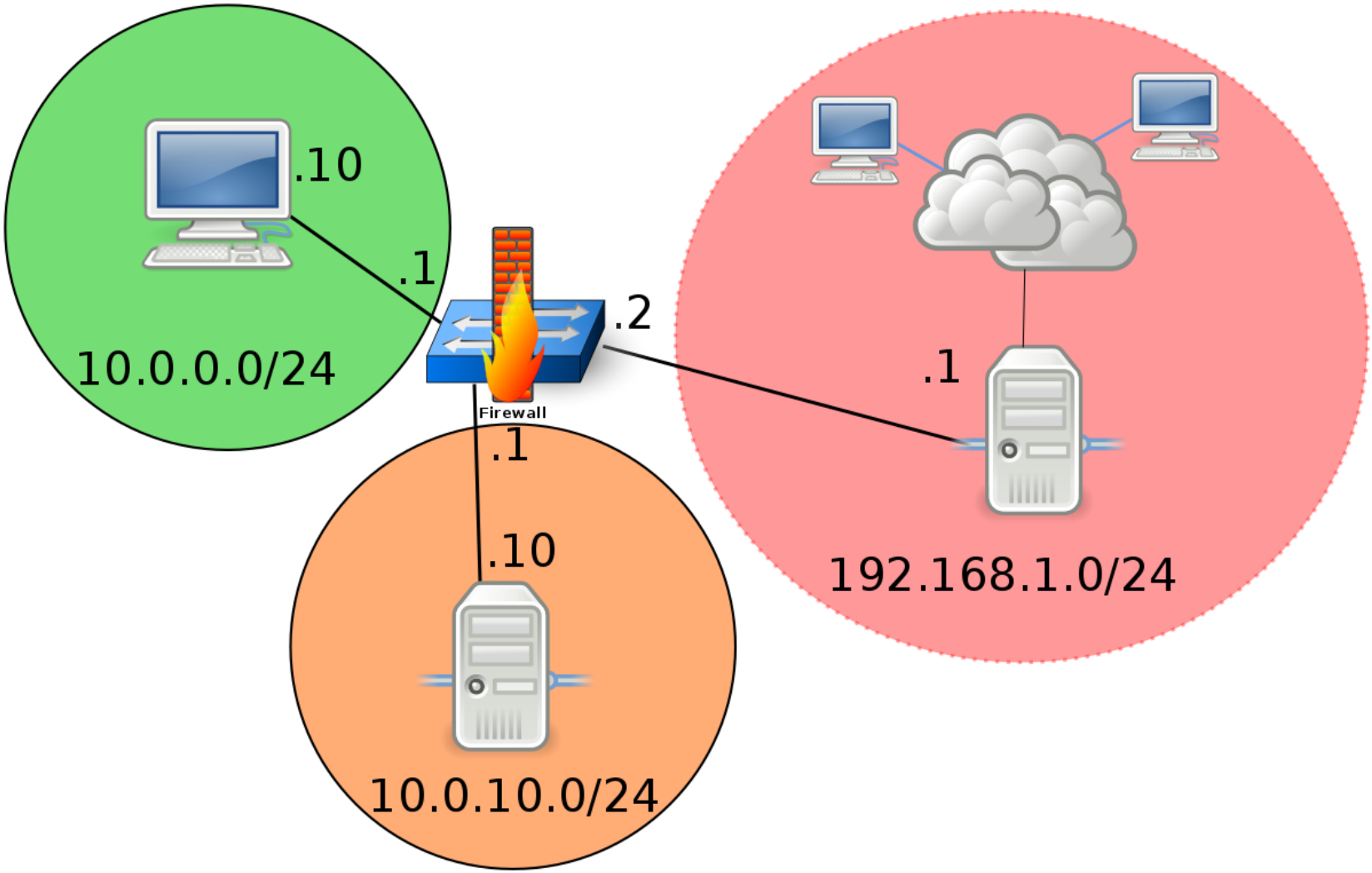
   - 7.2 Limit incoming SSH connections to 4 per minute (any more will be dropped)

```
iptables -I INPUT 4 -p tcp --dport 22 -m state
--state NEW -m recent --update --seconds 60
--hitcount 5 -j REJECT
```

Part 3: Setup firewall using ipFire

After server was infected with malicious software, the company decided to buy a "brand new" :) firewall and replace the simple switch provided. Your goal is to prevent such software to communicate. You know that stateful firewall should do the job.

Scenario



Part 3: Setup firewall using ipFire

connect to https://10.0.0.1:444 from green computer.

Part 3: Setup firewall using ipFire

Set up ipFire with the following conditions

1. Reject other packets than listed here
2. Accept SSH (Secure Shell) connection for source 10.0.0.0/24 192.168.1.0/24
3. Accept TCP communication through HTTP for all
4. Accept ICMP
5. Accept FTP communication with server
   - Note: We are using stateful firewall now
6. Test
   - nmap -sS 10.0.10.10

Part 3: Setup firewall using ipFire - scapy

```
# scapy
>>>send(IP(dst="192.168.1.2", src="10.0.10.10")/TCP(sport=21,
dport=10000, flags='S'))
```

Part 3: Proxy firewalls - Zorp GPL

Proxy Firewalls

- Network security systems that filter communication at the application layer
  - Also known as Application firewall or Gateway firewall
- Similarly to a proxy server, application firewalls act as an intermediary between the server and host
  - Also monitors layer 7 (application layer) protocols
  - Stateful inspection and deep layer packet inspection to protect from incoming attacks
- Negatives:
  - Additional processing overhead can cause bottleneck in the network
  - Support for only certain protocols limits which applications can be used within the network

Zorp GPL

- Open source proxy firewall
- Access control
  - Based on zones instead of hosts or IP ranges
- Information leakage prevention
  - Change or remove information from packets, such as internal IP addresses
- Content filtering
  - Used in conjunction with external application (virus scanner, spam filter, . . .)
- Zones - sets of IP subnetworks
  - Administrative hierarchy independent of physical network
  - Can be linked into a tree hierarchy

Part 3: Proxy firewalls - Zorp GPL

- Accepts rules based on the best match
  1. Evaluation order
  2. Condition scope

- Services - determines how the desired action is performed
  - PFService - Packet Filter services
  - Service - Application level services
  - DenyService - Reject connections, handle exceptions

Homework

Reading assignment (until exam)

References

Nmap

- https://nmap.org/

IPtables

- https://help.ubuntu.com/community/IptablesHowTo
- https://wiki.archlinux.org/index.php/iptables

Zorp GPL

- https://www.balabit.com/network-security/zorp-gpl
- http://zorp-gpl-tutorial.readthedocs.org/en/latest/index.html
- https://github.com/balabit/zorp