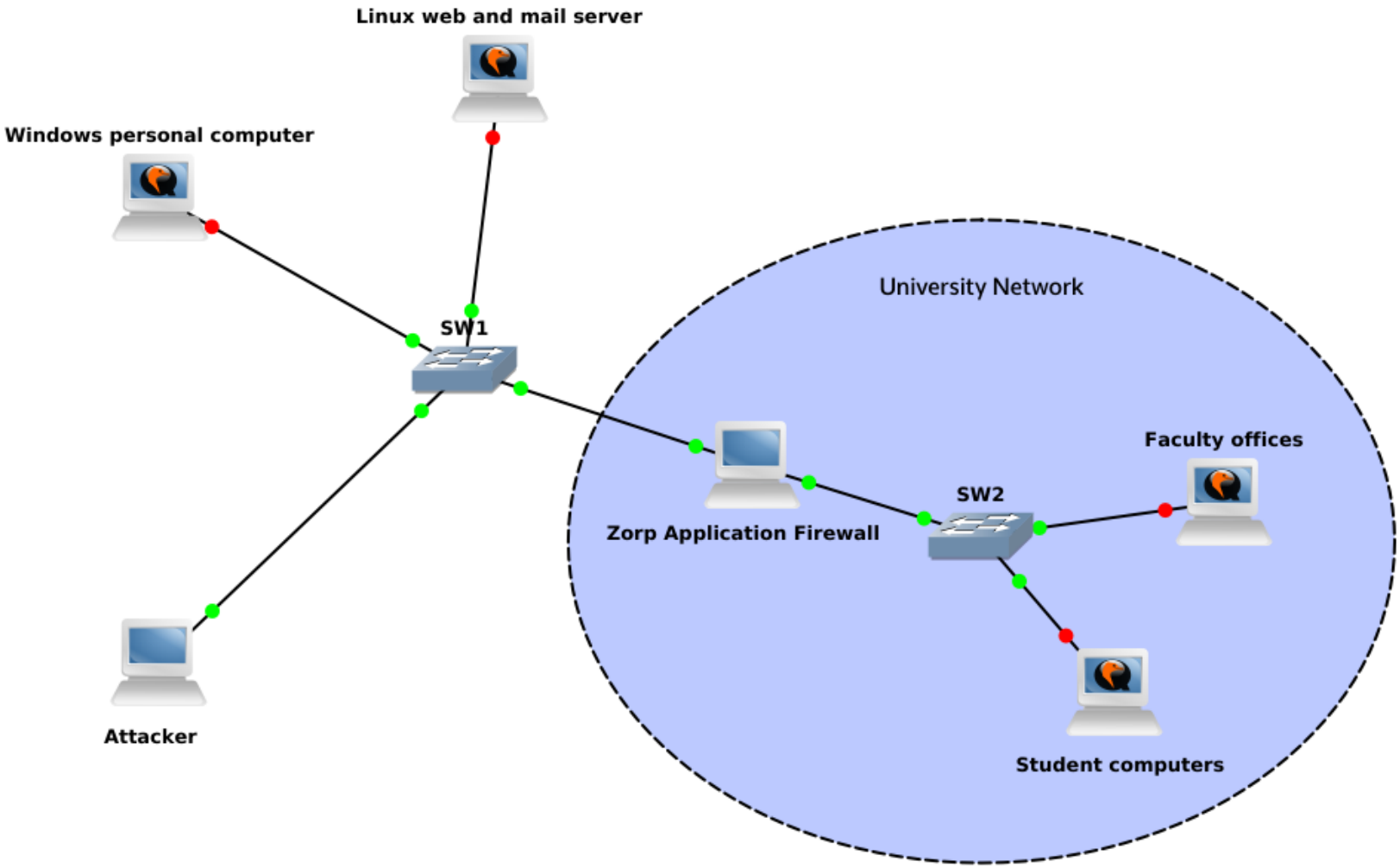
LAB7: Firewalls

Ondrej Bilčík

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

Scenario


Linux web and mail server
Windows personal computer
SW1
University Network
Faculty offices
SW2
Zorp Application Firewall
Attacker
Student computers

Setup

1. Start VirtualBox
2. Import PA197-Firewall Seminar.ova
3. Load virtual machine
4. Start VBox inside the machine
5. Load Windows and Linux 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

Part 1: Windows Firewall

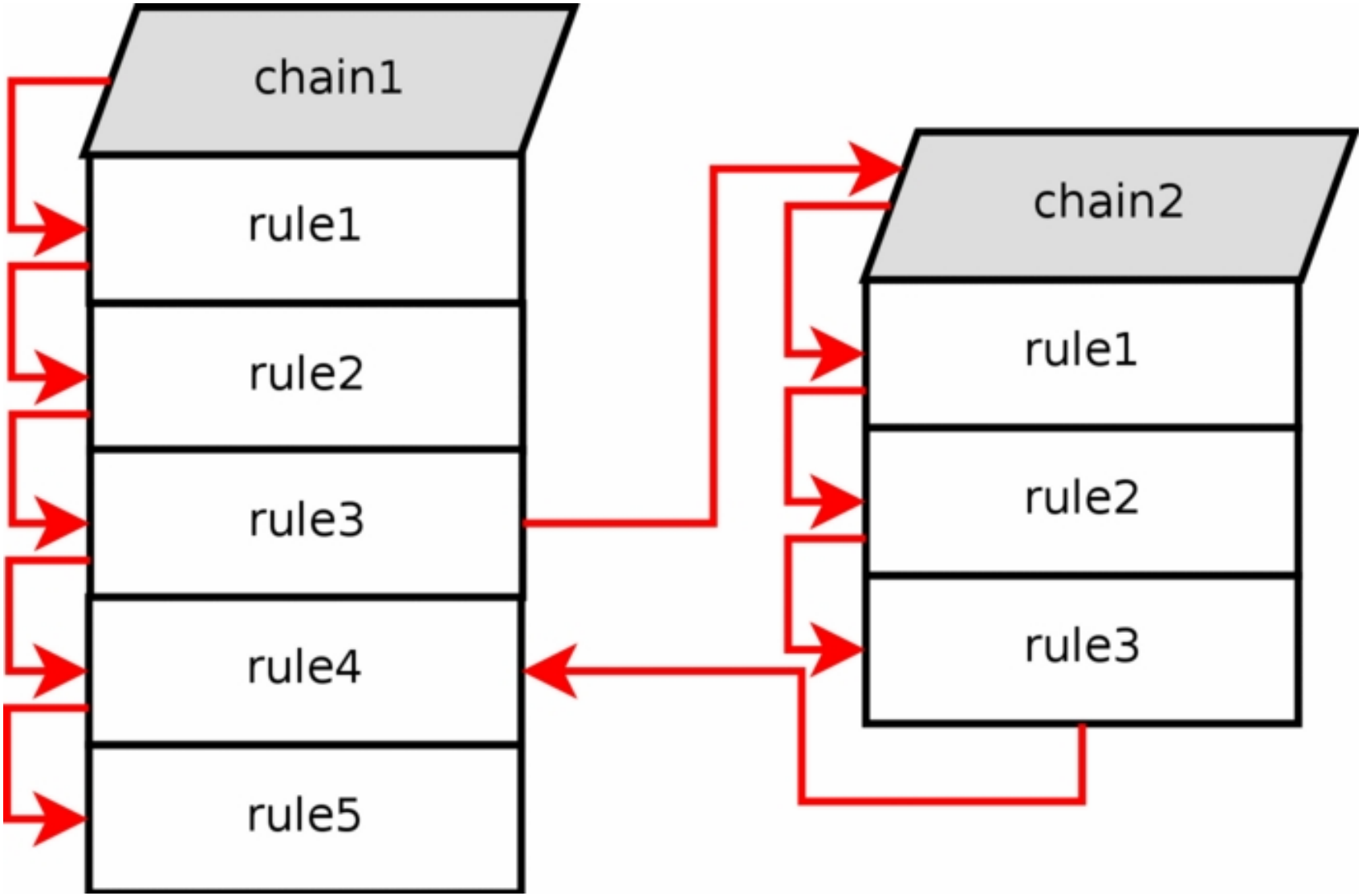
Set up the following:

1. Allow remote assistance (ping)
2. Configure exception for remote control without limited scope
3. Configure FTP services
4. Configure application exception for Internet Explorer
5. Test with Nmap
   - nmap -p 1-1000 -Pn 10.10.10.3

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

Set up IPtables with the following conditions

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

1. Accept established connections, and connections related to already allowed connections
2. Reject invalid packets
3. Accept SSH (Secure Shell)
4. Accept TCP communication through HTTP
5. Accept TCP communication through HTTPS
6. Accept SMTP(Simple Mail Transfer Protocol) communication
7. Protect firewall against Brute Force SSH attacks, limiting incoming SSH requests to 4 per minute
   - Note: Which rules are applied first?
8. Reject anything else
9. Test
   - nmap -sS 10.10.10.2

Part 2: Linux IPtables Task

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

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

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

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

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

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

5. Accept TCP communication through HTTPS

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

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

5. Accept TCP communication through HTTPS

   ```
   iptables -A INPUT -p tcp --dport 443 -j ACCEPT
   ```

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

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

5. Accept TCP communication through HTTPS

   ```
   iptables -A INPUT -p tcp --dport 443 -j ACCEPT
   ```

6. Accept SMTP(Simple Mail Transfer Protocol) communication

Part 2: Linux IPtables Task

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

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

2. Reject invalid packets

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

3. Accept SSH (Secure Shell)

   ```
   iptables -A INPUT -p tcp --dport 22 -j ACCEPT
   ```

4. Accept TCP communication through HTTP

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

5. Accept TCP communication through HTTPS

   ```
   iptables -A INPUT -p tcp --dport 443 -j ACCEPT
   ```

6. Accept SMTP(Simple Mail Transfer Protocol) communication

   ```
   iptables -A INPUT -p tcp --dport 25 -j ACCEPT
   ```

Part 2: Linux IPtables Task

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: 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

Part 3: Proxy firewalls - Zorp GPL

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

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