Estonian Information Technology College

Firewall

Operating systems I800

Edmund Laugasson
edmund.laugasson@itcollege.ee

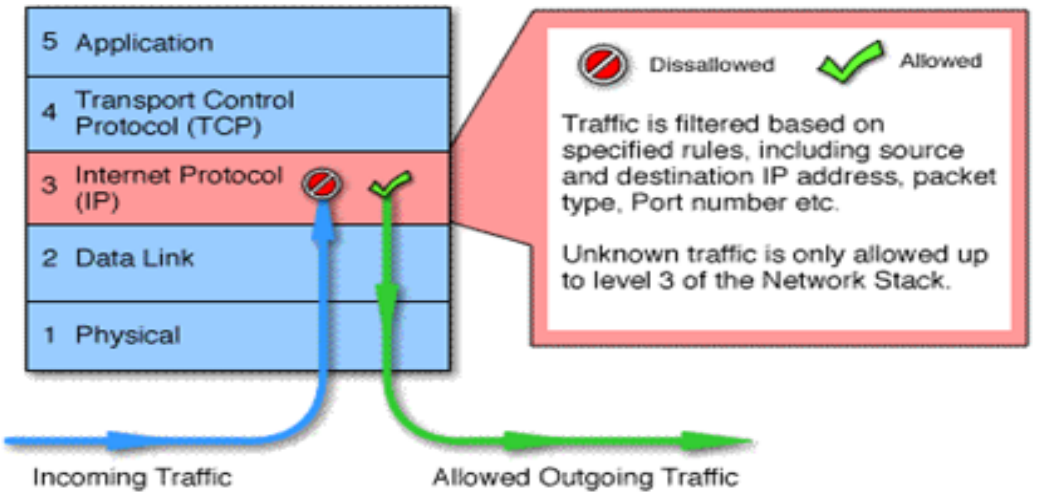
There has been used materials from Margus Ernits, Katrin Loodus when creating current slides.

Firewalls

- Firewalls can be divided into two categories
  - IP packets filtering firewalls (*packet filter*)
    - Make decisions on the basis of the IP packet header information (e.g. source / destination address / port basis)
  - *application-layer firewall*
    - Make decisions on the basis of the contents of the package (e.g. will not be allowed MS Exchange e-mail server in the direction of random challenges RPC functions, only those that are needed to use the service)
  - a further distinction between network firewalls, between two or more network (Internet, intranet) for traffic filtering and firewalls in computer (host-based firewall) to a particular computer network traffic filtering
  - This lecture will discuss the IP packet filtering for Linux systems

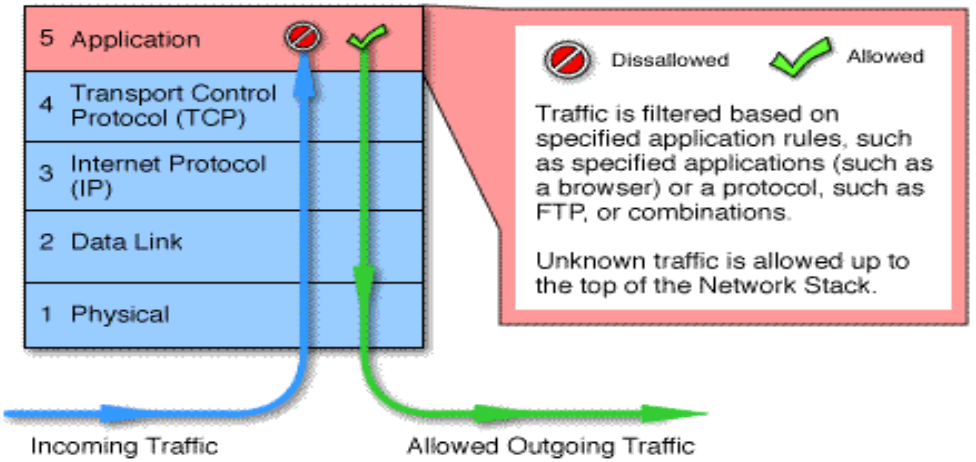
Firewalls

Packet filter



http://www.crazylearner.org/wp-content/uploads/2013/05/packet-filter-firewall.png

Application-layer firewall



http://www.di.unisa.it/~ads/corso-security/www/CORSO-0102/astaro/app_layer_f.gif

Firewall in OSI model

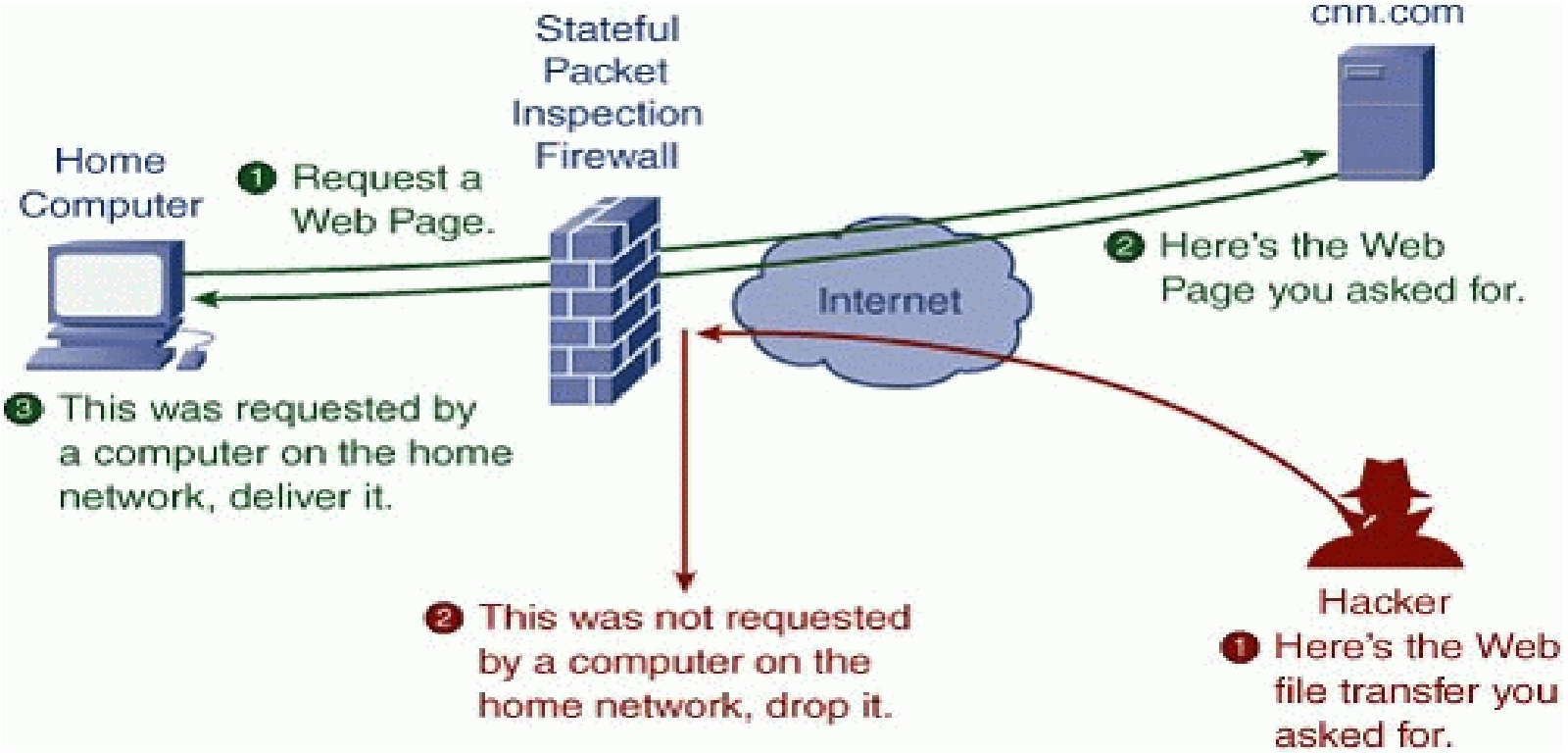
OSI - *Open Systems Interconnection*

OSI model	PDU Protocol Data Unit		
7. *Application layer*	*Data*	*application-layer firewall* (*3rd generation firewall*)	Make decisions on the basis of the contents of the IP packet (such as specific websites, malware, etc.)
6. *Presentation layer*			
5. *Session layer*		*circuit gateway / firewall* TCP/UDP connection security, there is monitored TCP packet handling, the firewall rules and enforcement policies (*2nd gen FW*)	When the web request goes through the circuit gateway / firewall, user-based information is filtered (e.g. IP address). The proxy server transmits queries to the Web server. Sending a response to an external server sees the proxy servers IP address within the network, but nothing about (query the actual sender) on. Web-based or other external network, the server responds to the proxy server, which is transmitted by the circuit gateway / firewall client.
4. *Transport layer*	*Segment* (*TCP*) / *Datagram* (*UDP*)		
3. *Network layer*	*Packet*	*packet filter* (*1st gen FW*)	IP packet filtering, decisions are made on the basis of the IP packet header information (such as source / destination address / port basis)
2. *Data link layer*	*Frame*	*MAC-layer firewall*	*Exchange layer MAC sub layer is filtering packets according to the access control list (ACL) entries in which the specific MAC addresses are connected.*
1. *Physical layer*	*Bit*		

The need of firewall

- For what do we need a firewall?
  - By allowing connections to certain services and certain host computers
  - By allowing outbound connections only from allowed computers and connect only to allowed services
  - Forward packets according to predefined rules (for example, the connection sharing)
  - Restrict / tune traffic - various service packages are dealt with in different ways. For example, you can set the quota amount of data - which ISP's do.
  - The intranet services may not all be visible on the Internet (e.g. available only through the VPN)

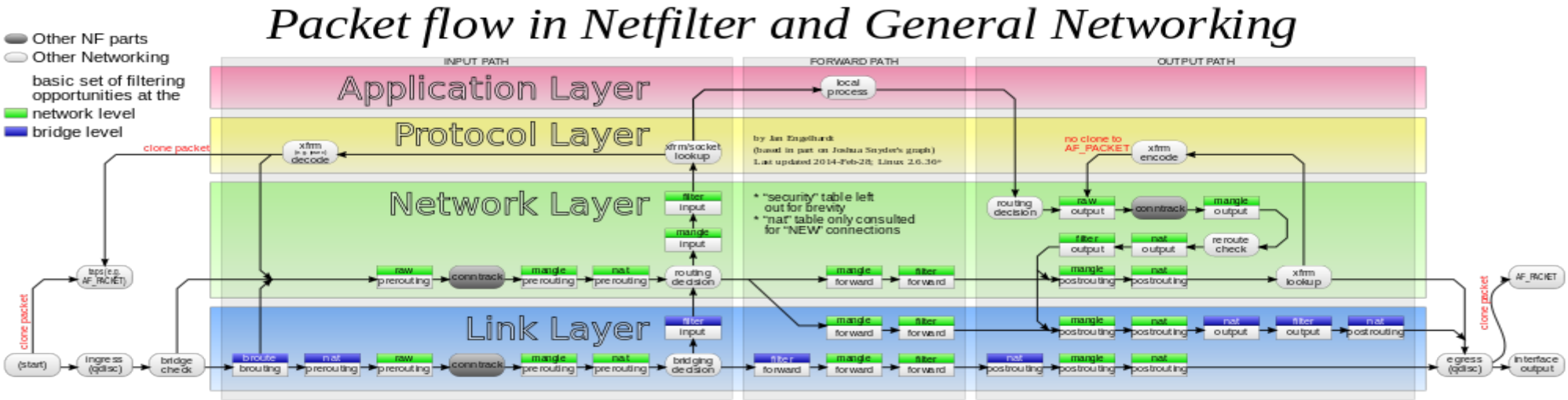
An overview of firewall



http://2.bp.blogspot.com/-Iblk0555nyY/UB-NbNjYpHI/AAAAAAAAAMw/fr6KGlr63JE/s1600/8-6+firewall1.gif

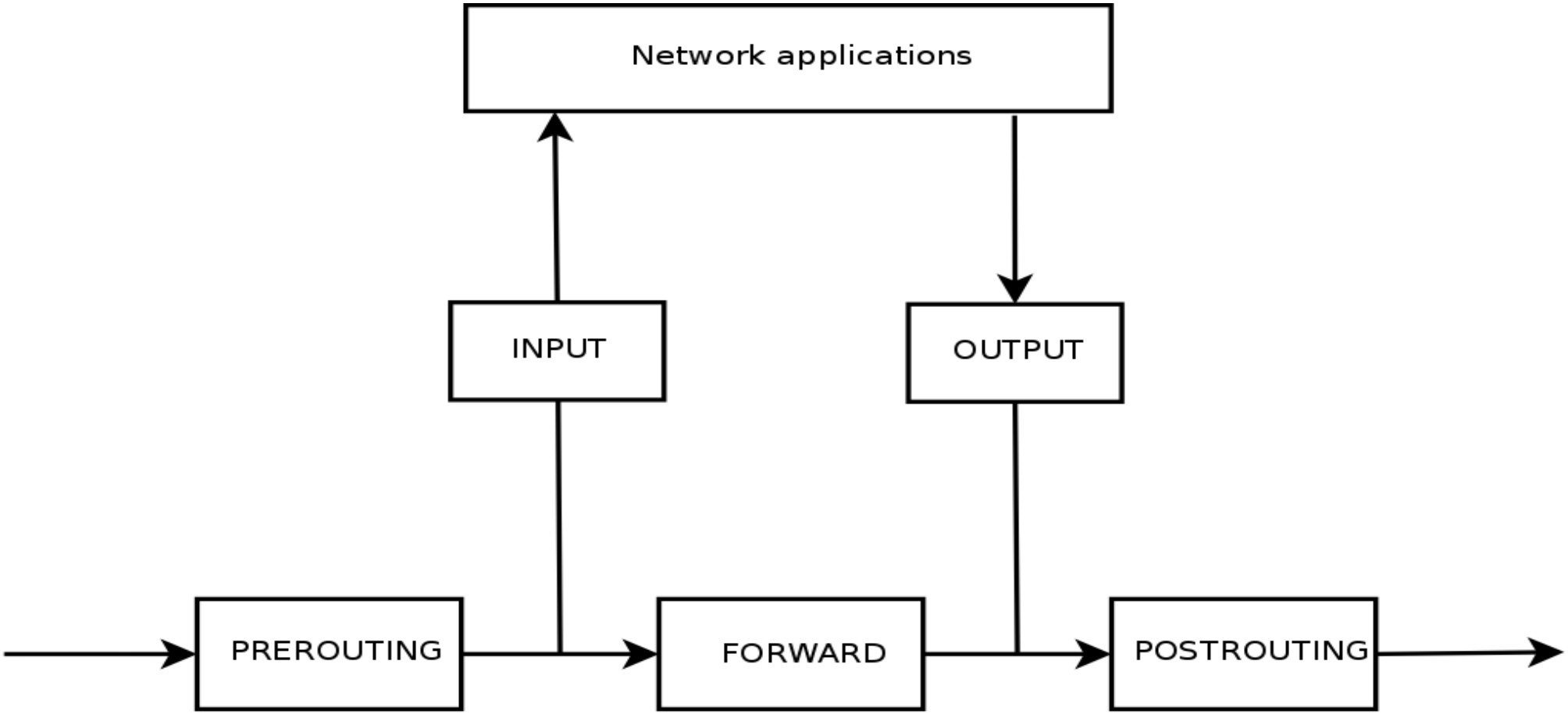
Netfilter in Linux

The program *iptables* is meant for configuring of Linux network packets filtering software *netfilter*



https://en.wikipedia.org/wiki/Firewall_(computing)#/media/File:Netfilter-packet-flow.svg

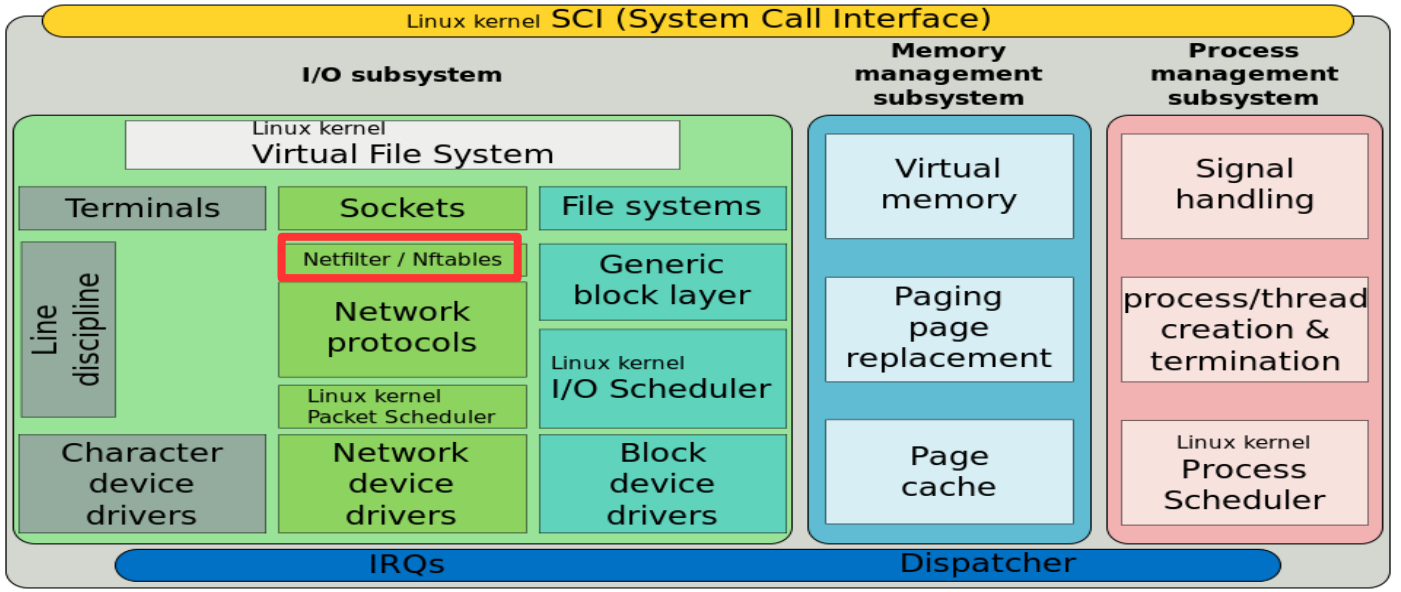
Netfilter



https://debian-handbook.info/browse/stable/images/netfilter.png

nftables in Linux

- since the version 3.13 of kernel
  - https://launchpad.net/ubuntu/+source/nftables
  - https://home.regit.org/netfilter-en/nftables-quick-howto/
  - http://askubuntu.com/questions/517136/list-of-ubuntu-versions-with-corresponding-linux-kernel-version
  - https://en.wikipedia.org/wiki/List_of_Ubuntu_releases#Table_of_versions
- meant to replace *netfilter*



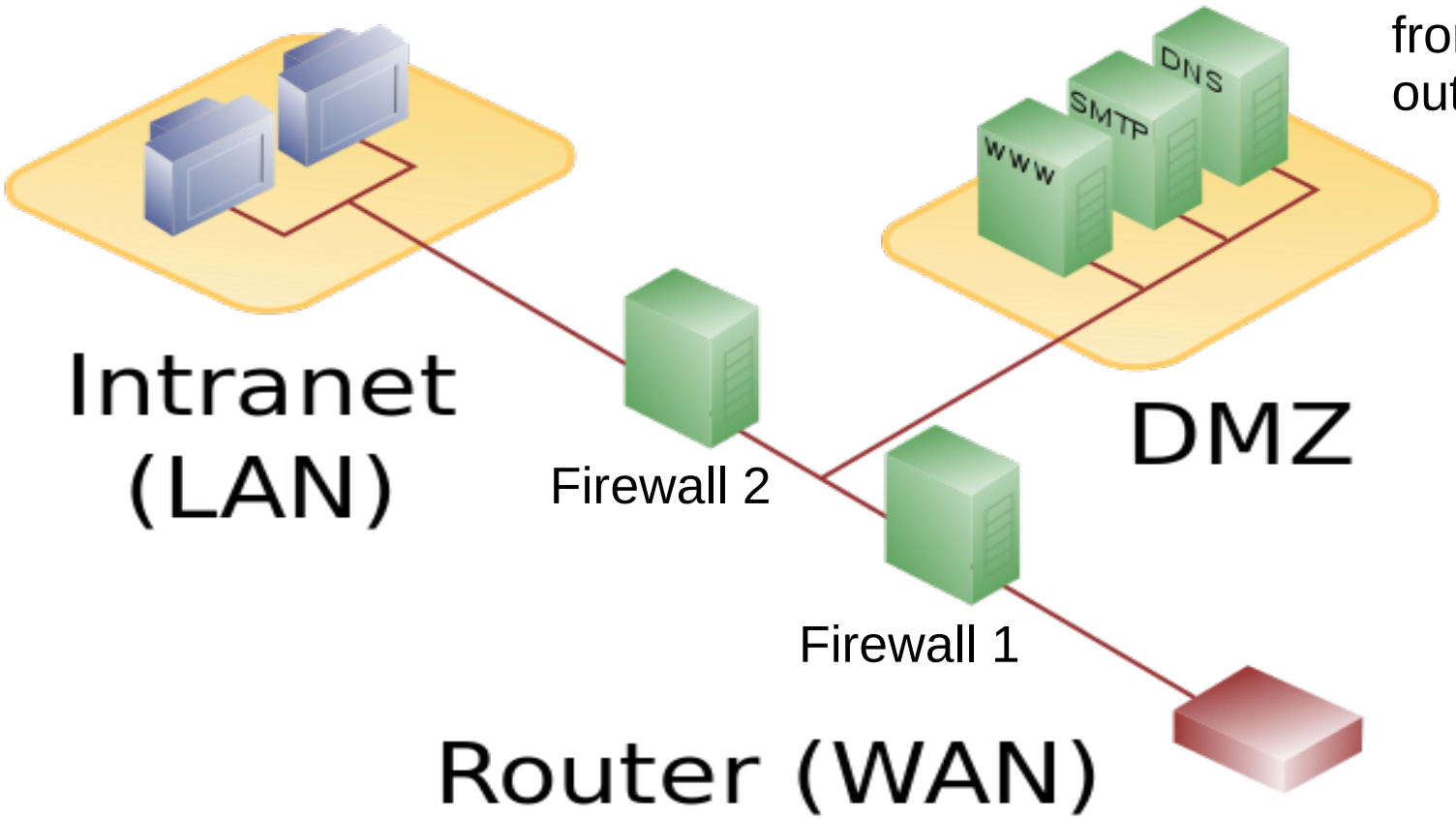
https://en.wikipedia.org/wiki/Nftables
https://en.wikipedia.org/wiki/Nftables#/media/File:Simplified_Structure_of_the_Linux_Kernel.svg

DMZ 1

- DMZ - *demilitarized zone*
- Meant to separate external network services from internal network
- Internal and external network can use the services located in the DMZ zone
- Delimiters are firewalls
- DMZ - the most dangerous zone where fighting goes :)

DMZ 2

Network services can be accessed from inside and outside.



https://en.wikipedia.org/wiki/DMZ_(computing)
https://en.wikipedia.org/wiki/Local_area_network
https://en.wikipedia.org/wiki/Wide_area_network

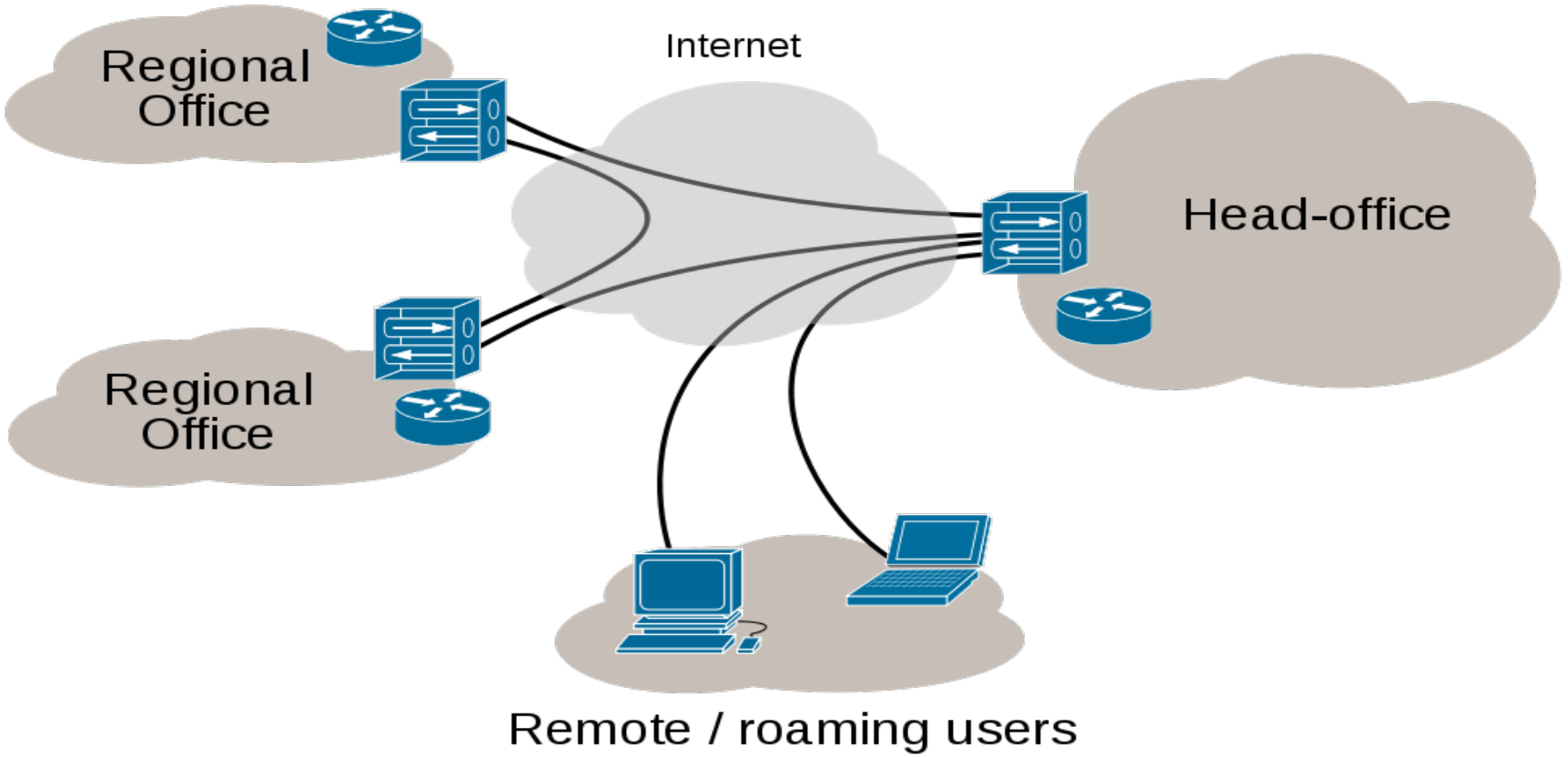
http://upload.wikimedia.org/wikipedia/commons/thumb/6/60/DMZ_network_diagram_2_firewall.svg/640px-DMZ_network_diagram_2_firewall.svg.png

VPN

- VPN - *Virtual Private Network*
- Widely used when accessing to the internal network to access the services (such as access to commercial databases in Library etc)
- The user is detected, and network traffic is encrypted
  - An anonymous attacker can not access the internal network services
  - A computer virus can still :(
- Some companys do not have an intranet and all services are used through VPN connections

https://en.wikipedia.org/wiki/Virtual_private_network

VPN overview

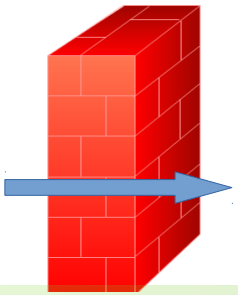


https://en.wikipedia.org/wiki/Virtual_private_network#/media/File:Virtual_Private_Network_overview.svg

Network Address Translation 1

- **NAT** (*Network Address Translation*) – network address conversion (connecting different address spaces over the network by changing network address in IP packet header)
  - https://en.wikipedia.org/wiki/Network_address_translation
  - http://www.ipv6.com/articles/nat/NAT-In-Depth.htm
- **DNAT** (*Destination NAT*) conversion of external IP address to internal; practical usage: *port forwarding, DMZ* – queries from external to internal network

SourceIP: 193.40.xxx.xxx
Dest IP: 193.40.194.200



SourceIP: 193.40.xxx.xxx
Dest IP: 172.16.0.170

Firewall converts the recipient's address (*Dest IP*)

Intranet

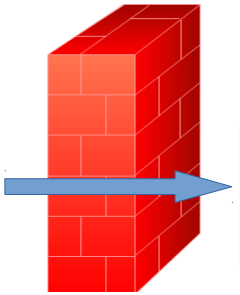
Intranet aadresses

- IPv4 aadresses (RFC 1918)
  - 10.0.0.0 – 10.255.255.255, qty: 16 777 216 (24 bit = 2^{24})
  - 172.16.0.0 – 172.31.255.255, qty: 1 048 576 (20 bit = 2^{20})
  - 192.168.0.0 – 192.168.255.255, qty: 65 536 (16 bit = 2^{16})
- IPv6 aadresses (RFC 4193)
  - fc00::/7, qty: 2^{121}
- device itself (*localhost*):
  - IPv4 127.0.0.1
  - IPv6 ::1
- please see also reserved IP addresses https://en.wikipedia.org/wiki/Reserved_IP_addresses

Network Address Translation 2

- **SNAT** (Source NAT) – internal network address conversion into external one, practical usage: queries from internal network to external one (most common)

SourceIP: 192.168.10.15
Dest IP: 193.40.194.200



SourceIP: 193.40.194.2
Dest IP: 193.40.194.200

Intranet

Firewall (193.40.194.2) converts the source address (sender). (Source *IP*)

ufw

- Firewalls and other computer systems there is also a need to filter IP packets
- Ubuntu is using by default:
  ufw (*uncomplicated firewall*) (CLI)
  gufw (GUI) – there is also possible to configure a firewall in another computer over network (even from MS Windows) – there must be at least *gufw* version 13.10.2 or newer
- is a CLI for *iptables* but does not allow all but is easier to use
- https://help.ubuntu.com/community/UFW
- https://help.ubuntu.com/lts/serverguide/firewall.html

ufw 2

- **ufw [--dry-run] [options] [rule syntax]**
  - *--dry-run* will simulate but not apply
  - a simple syntax is used (port number and protocol if wanted)
  - rules are located at */etc/ufw/*.rules*
  - log file */var/log/ufw.log*
  - settings */etc/default/ufw* and */etc/ufw/*.conf*
  - the order of rules is important
    - first ones will be enacted (put info effect) first
    - more detail rules before
    - more general rules afterwards
  - application profiles (using .ini syntax)
    - */etc/ufw/applications.d/* and also there will be checked the file */etc/services*
    - GUI */etc/gufw/app_profiles/*
  - *man ufw (man gufw)*

ufw options

- *allow* – enable (will not be investigated further)
- *deny* - we reject and do not report it
- *reject* – we reject and we will report it
- *limit* – for *DoS, DDoS* prevention
  (please ensure that IPv6 is supported – *man ufw*)
  (by default after 6th try the query will be blocked for 30 second)
- *status* – whether the ufw is turned on of not
- *show* – current rules
- *reset* – will turn off the firewall and restore default settings
- *reload* – will reload rules

ufw syntax 1

- turn on/off (permanently – also after restarting computer):
  - *ufw [--dry-run] enable|disable|reload*
- allow/disable ports
  - *sudo ufw allow|deny|reject port|[/protocol]*
  - *protocol: tcp, udp*
    - *sudo ufw allow 22*
    - sudo ufw allow 22/tcp
  - by default there will be disabled incoming traffic but also outgoing can be disabled,
    e.g. *sudo ufw reject out ssh*

ufw syntax 2

- deleting a rule
  - rule: *sudo ufw deny 80/tcp*
  - deleting: *sudo ufw **delete** deny 80/tcp*
- service names and ports
  - *less /etc/services* (q to quit from less)
  - *sudo ufw allow|deny servicename*
    - *sudo ufw allow ssh* (both tcp and udp will be allowed)
    - *sudo ufw allow ssh/tcp* (only tcp will be allowed)
  - by using the service name there will be looked into the file */etc/services* and when there will be required name found then there will be turned on all ports related with looked service

ufw syntax 3

- logging
  - *sudo ufw logging <on/off/LEVEL>*
  - *LEVEL: off, low, medium, high, full; on=low*
    - *sudo ufw logging on*
- investigating the status
  - *sudo ufw status*
  - *sudo ufw status [verbose, numbered]*
  - *sudo ufw app list* (all available application profiles)

ufw example

- let us check current policy
  - **grep 'DEFAULT_' /etc/default/ufw**
  - *DEFAULT_INPUT_POLICY="REJECT"*
  - *DEFAULT_OUTPUT_POLICY="ACCEPT"*
  - *DEFAULT_FORWARD_POLICY="DROP"*
  - *DEFAULT_APPLICATION_POLICY="SKIP"*
- before turning on the firewall we will deny all incoming traffic
  - *sudo ufw default deny* (by default for incoming traffic will be applied)
  - more precise:
    - *sudo ufw default allow outgoing*
    - *sudo ufw default deny incoming*
- will allow SSH to connect with server (would be crucial part when we are connected over network using SSH)
  - *sudo ufw allow ssh*
- turn on the firewall
  - *sudo ufw enable*

ufw example 2

- when SSH is allowed and firewall is turned on then remote acces is granted
- now can we allow more services in firewall
- let us check the status
  - *sudo ufw app list* (all available application profiles)
  - *sudo ufw status* (ports, protocols)
- allowing web server (*tcp* and *udp* will be allowed)
  - *sudo ufw allow http* (or also *sudo ufw allow 80/tcp*)
  - *sudo ufw allow https* (or also *sudo ufw allow 443/tcp*)

ufw example 3

- allow Samba (MS Windows file server)
  - *sudo ufw allow 137,138/udp*
  - *sudo ufw allow 139,445/tcp*
- when appropriate application profile exist then
  - *sudo ufw allow Samba*
  - profile can be copied */etc/gufw/app_profiles/samba.jhansonxi* to the location */etc/ufw/applications.d/samba* (file extension is not important)
  - restrictiing access with one subnet:
    sudo ufw allow from 192.168.0.0/16 to any app Samba
- allow logging (by default the logging level is *low*):
  sudo ufw logging on

ufw application profile example

- Samba profile example */etc/ufw/applications.d/samba*
  (the keyword in square brackets will be used when writing a ufw rule, it is case sensitive):

[Samba]

title=SAMBA

description=SMB/CIFS protocol for Unix systems, allowing you to serve files and printers to Windows, NT, OS/2 and DOS clients

ports=137,138/udp|139,445/tcp

categories=Network;Services;|Network;File Transfer

reference=[http://www.samba.org/samba/docs/server_security.html]

- info mooduli kohta, nt CUPS:
  sudo ufw app info CUPS

ufw application profile

- into one file */etc/ufw/applications.d/appsprofiles* there could be also multiple apps profiles added (the file name of profiles can be freely to choose):

[puppet]

title=puppet configuration manager

description=Puppet Open Source from http://www.puppetlabs.com/

ports=80,443,8140/tcp

[AMANDA]

title=AMANDA Backup

description=AMANDA the Advanced Maryland Automatic Network Disk Archiver

ports=10080

ufw application profile

- different profiles of same application in one file *(/etc/ufw/applications.d/appsprofiles*

```
===start of apache2.2-common file===
[Apache]
title=Web Server
description=Apache v2 is the next generation of the omnipresent Apache web server.
ports=80/tcp

[Apache Secure]
title=Web Server (HTTPS)
description=Apache v2 is the next generation of the omnipresent Apache web server.
ports=443/tcp

[Apache Full]
title=Web Server (HTTP,HTTPS)
description=Apache v2 is the next generation of the omnipresent Apache web server.
ports=80,443/tcp

===end of file===
```

ufw example 4

- disabling responding to *ping*
  - in file */etc/ufw/before.rules*
    (in case of IPv6 there is *before6.rules*) there has to be changed:
    # ok icmp codes
  - -A ufw-before-input -p icmp --icmp-type destination-unreachable -j **DROP**
  - -A ufw-before-input -p icmp --icmp-type source-quench -j **DROP**
  - -A ufw-before-input -p icmp --icmp-type time-exceeded -j **DROP**
  - -A ufw-before-input -p icmp --icmp-type parameter-problem -j **DROP**
  - -A ufw-before-input -p icmp --icmp-type echo-request -j **DROP**
- preventing attacks to SSH port (⩾6 queries during 30 s)
  - *sudo ufw limit SSH (sudo ufw limit 22/tcp)*
  - *sudo ufw limit from 205.184.2.4 to tcp 22*

ufw black list

- in file /etc/ufw/before.rules

```
## blacklist section
# block just 199.115.117.99
-A ufw-before-input -s 199.115.117.99 -j DROP
# block 184.105.*.*
-A ufw-before-input -s 184.105.0.0/16 -j DROP
# don't delete the 'COMMIT' line or these rules won't be processed
COMMIT
```

ufw routing (masquerade)

- allowing packets forwarding in file */etc/default/ufw*
  - DEFAULT_FORWARD_POLICY="ACCEPT"
- allowing also in file */etc/ufw/sysctl.conf*
  - net.ipv4.ip_forward=1
  - net.ipv6.conf.default.forwarding=1 #(when IPv6 is in use)
- /etc/ufw/before.rules

```
# nat Table rules
*nat
:POSTROUTING ACCEPT [0:0]
# Forward traffic from eth1 through eth0.
-A POSTROUTING -s 192.168.0.0/24 -o eth0 -j MASQUERADE
# don't delete the 'COMMIT' line or these nat table rules won't be processed
COMMIT
```

- restart ufw: *sudo ufw disable && sudo ufw enable*

ufw port forwarding

- in file */etc/ufw/before.rules* there has to be changed:

```
# NAT table rules
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-F
# Port Forwardings
-A PREROUTING -i eth0 -p tcp --dport 22 -j DNAT --to-destination 192.168.1.10
# Forward traffic through eth0 - Change to match you out-interface
-A POSTROUTING -s 192.168.1.0/24 -o eth0 -j MASQUERADE
# don't delete the 'COMMIT' line or these nat table rules won't
# be processed
COMMIT
```

ufw and modules

- in file */etc/default/ufw* in the end there is a row *IPT_MODULES="nf_conntrack_ftp nf_nat_ftp nf_conntrack_netbios_ns"*
- the preceding description of various modules that can be added - if the module is not active then despite promising firewall rules problems will arise
- for instance in case of Samba the module *nf_conntrack_netbios_ns* would be needed
- *uname -r* will show kernel version in use and *dpkg --list* | *grep linux-image* will show installed kernels
- then there can be all usable kernel modules for ufw checked from file (x should be replaced with appropriate kernel version) */usr/src/linux-headers-x.x.x-xxxxxx/net/netfilter/Kconfig*

ufw allows also more precisely

- disabling traffic from address 12.34.56.78 to port 22 in local network (intranet)
  sudo ufw deny proto tcp from 12.34.56.78 to any port 22
- allowing access from certain IP or network segment
  - *sudo ufw allow from <ip address>*
  - *sudo ufw allow from 205.26.134.122*
  - *sudo ufw allow from 192.168.1.0/24*
- specifying access:
  - *sudo ufw allow from <target> to <destination> port <port number>*
  - *sudo ufw allow from 192.168.0.4 to any port 22*

ufw more precisely...

- allowing more precisely
  - *sudo ufw allow from <target> to <destination> port <port number> proto <protocol name>*
  - *sudo ufw allow from 192.168.0.4 to any port 22 proto tcp*
- denying more precisely
  - *sudo ufw deny from <ip address>*
  - *sudo ufw deny from 205.26.134.122*
- denying with port number and protocol
  - *sudo ufw deny from <ip address> to <protocol> port <port number>*
  - *sudo ufw deny from 192.168.0.1 to any port 22*

ufw precisely, example

- denying access from two IP address but allowing for others in same network segment the port 22 (usually SSH)
  - *sudo ufw deny from 192.168.0.1 to any port 22*
  - *sudo ufw deny from 192.168.0.7 to any port 22*
  - *sudo ufw allow from 192.168.0.0/24 to any port 22 proto tcp*
- here will be clarifying rules before general one set – also the order of rules is important!

iptables

- *ufw* basis is *iptables*
- the program *iptables* is meant for configuring of Linux network packages filtering software *netfilter*
- *netfilter/iptables* software can be divided into two
  - *kerneli* modules
  - *user space* software

Iptables possibilities

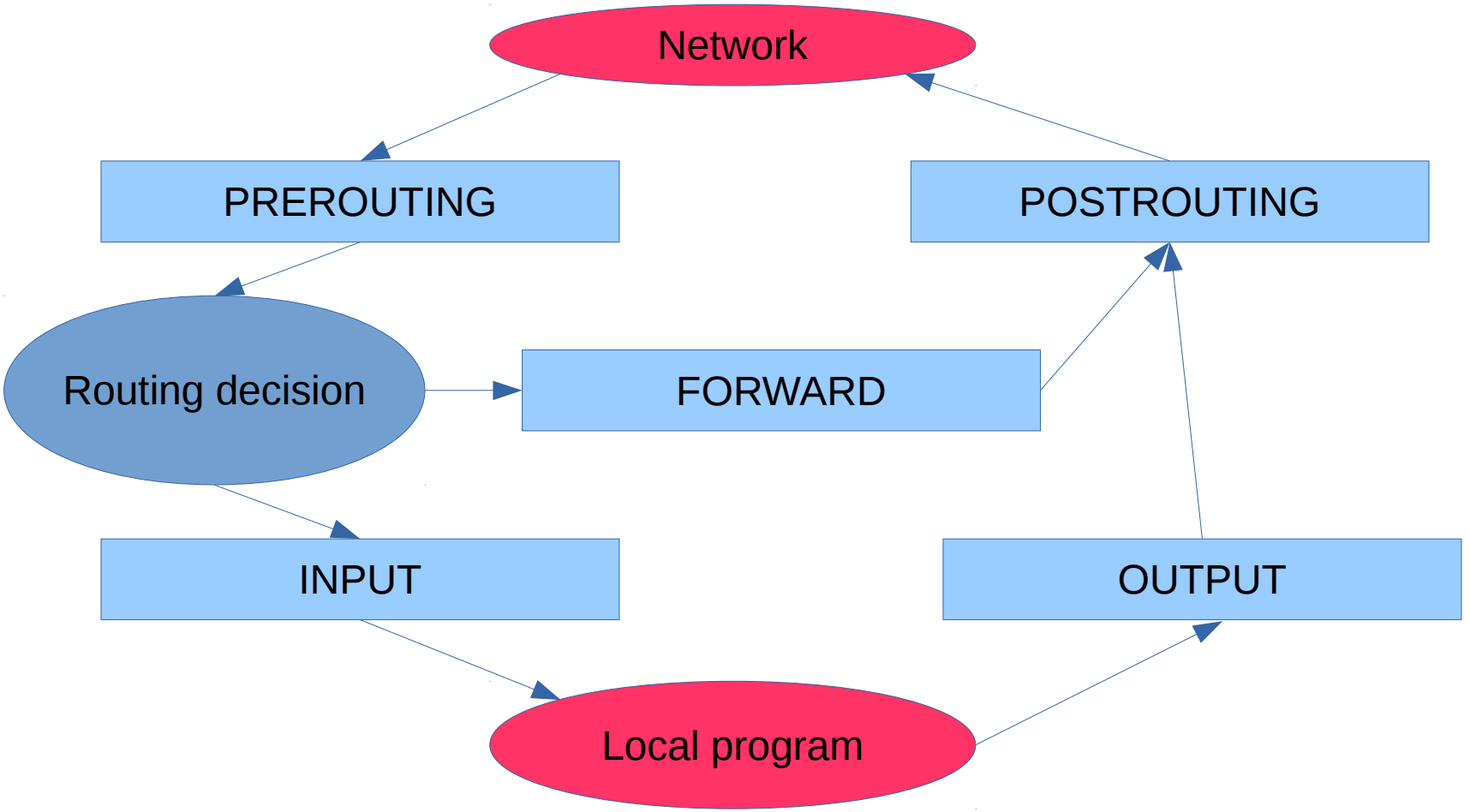
- A data packet filtering
- connection tracking
- NAT (Network Address Translation)
- *Mangling* (changing packets)
- Traffic shaping (limiting traffic speed)
- Logging

iptables tables and chains

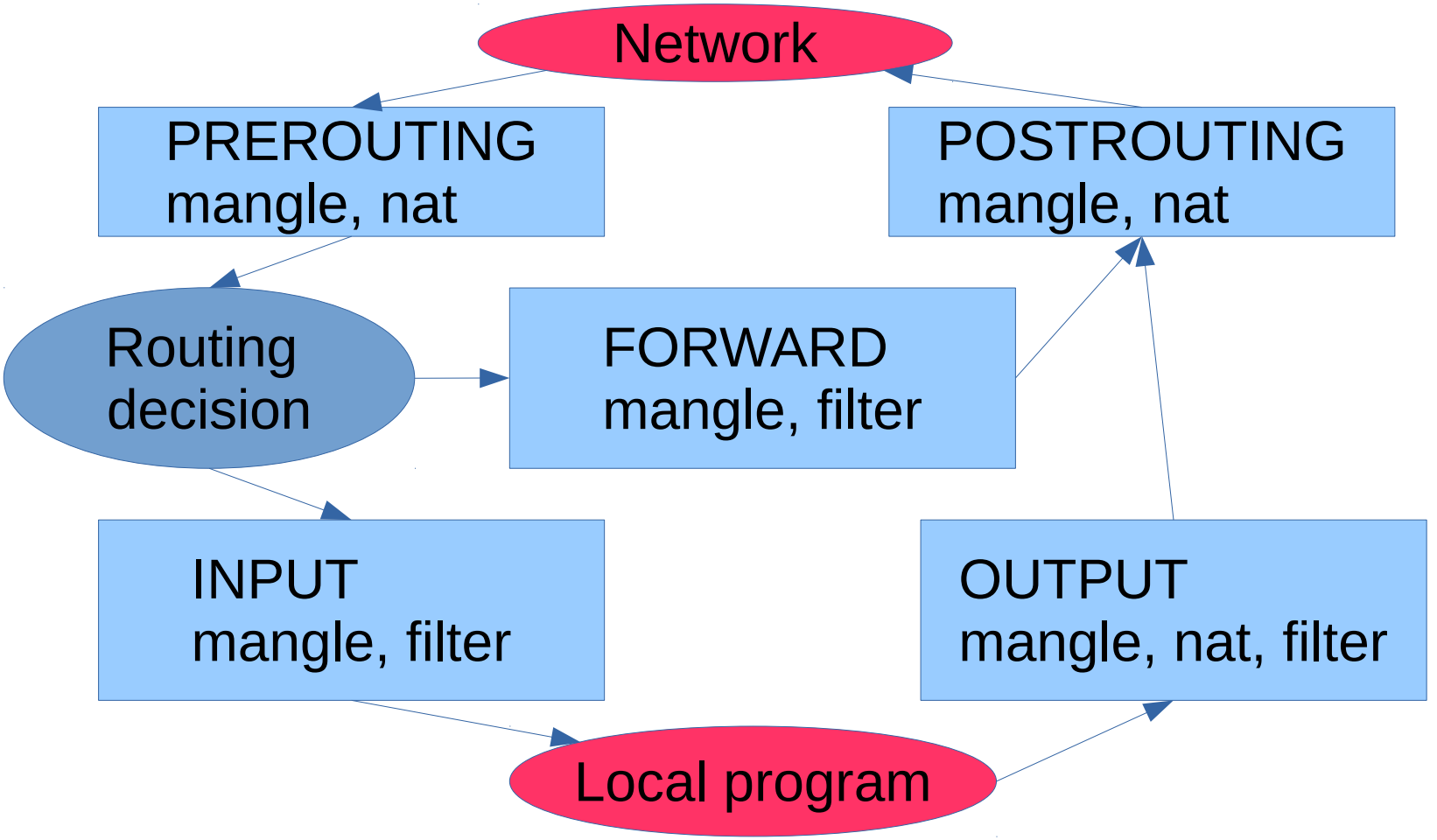
- Tables
  - Filter
  - Nat
  - Mangle
- Chains
  - Prerouting
  - Input
  - Output
  - Forward
  - Postrouting

- all packages do not need to be redirected through all the rules
- various chains are to reduce tables quantity to be investigated by the program
- *man iptables*

iptables chains



iptables tables and chains



http://www.iptables.info/en/structure-of-iptables.html

iptables tables

- Filter
  - to filter packets
    - FORWARD will filter forwarding packets
    - INPUT will filter incoming packets
    - OUTPUT will filter outgoing packets
- Nat
  - for network address translation
    - PREROUTING address translation before routing DNAT (destination NAT)
    - POSTROUTING address translation after routing SNAT (source NAT)
    - OUTPUT the NAT carried out by firewall
- Mangle
  - Modifying IP packet header
    - modifying QoS bit

iptables common syntax

- **iptables [-t|--table table] -command [chain] [-i interface] [-p protocol] [-s address [port[:port]]] [-d address [port[:port]]] -j policy**
- default table is **filter**

iptables examples

- denying traffic to http port:
- iptables -A INPUT -d 0.0.0.0/0 -p tcp --destination-port 80 -j REJECT
  - REJECT – refuse of packet and announce about that also to sender
- allowing connections from localhost
  - iptables -A INPUT -i lo -j ACCEPT
- listing current rules
  - iptables -L

iptables examples

- redirection
  iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 3127
  - all into port 80 received packets will be redirected to port 3127 (e.g. proxy)
- clearing INPUT table
  iptables -F INPUT

Masquerading examples

- **iptables --flush**
- **iptables --table nat --flush**
- **iptables --delete-chain**
- **iptables --table nat --delete-chain**
- **iptables --table nat --append POSTROUTING --out-interface eth1 -j MASQUERADE**
- **iptables --append FORWARD --in-interface eth0 -j ACCEPT**

iptables policy

- ACCEPT – allowing. No further investigation
- DROP – blocking. No further investigation
- LOG – logging corresponding current rule and iptables will continue rule investigation
- REJECT – blocking. Sender will be notified about that (e.g. icmp-host-prohibited)
- DNAT – destination (receiver) address changing
- SNAT – source (sender) address changing
- MASQUERADE – sender’s address will be changed as firewall address

iptables parameters

- -t <-table> table, default filter
- -j <target> applied policy
- -A adds rule
- -F Flush. Deletes given table rules
- -p <protocol-type> Protocol (*icmp, tcp, udp* and *all*)
- -s <ip-address> Sender IP
- -d <ip-address> Destination IP
- -i <interface-name> Input device
- -o <interface-name> Output device
  - e.g. ***iptables -A INPUT -s 0/0 -i eth0 -d 10.0.0.1 -p TCP -j ACCEPT***

iptables example

- iptables -A FORWARD -s 0/0 -i eth0 -d 10.10.10.10 -o eth1 -p TCP --sport 1024:65535 --dport 80 -j ACCEPT
  - allowing packet forwarding if packet will be sent to IP address 10.10.10.10 and is sent from network interface eth0 and is needed to route through eth1. Source port should be higher from privileged ports and destination port should be http

iptables

- suggestions
  - investigate chains and tables to understand them
  - using some configuration tool to simplify rules creation would be useful, especially for beginners (searching from Internet would help)
  - start with simple rules (filter)
  - test every added rule
  - save rules into file and add comments

iptables disadvantages

- there is not possible determine multiple activities per one rule, e.g. LOG + DROP, MARK + ACCEPT etc.
- rules re-reading is arduous (takes more effort)
- reuse of similar code insufficient (the same mistake must be corrected in several places)
- Nftables – next generation (meant to replace ip, ip6, arp, ebtables and in one day also iptables)

OpenBSD PF

- PF (packet filter) comparison with iptables/netfilter
  - the rules are more readable as compared iptables rules
  - iptables allows the use of external modules
  - various performance tests indicate that PF is better in dealing with Stateful Rules (rule takes less memory)
  - OpenBSD SMP support is not the same level as of in Linux and therefore may be a lower performance compared with multicore Linux firewalls
  - OpenBSD has the advantage of system security

FirewallBuilder

- Many IT systems administrators prefer a commercial product because they want to get a nice new configuration interface
- FirewallBuilder able to generate rules for multiple firewalls
  - CISCO
  - BSD PF
  - Linux iptables
- https://wiki.itcollege.ee/index.php/FirewallBuilder - in Estonian
- http://www.fwbuilder.org/

Links (both in English and Estonian)

- Ubuntu default firewall ufw (GUI: gufw)
  https://wiki.itcollege.ee/index.php/Ufw
  https://wiki.itcollege.ee/index.php/Ubuntu_ruuter
  https://help.ubuntu.com/community/Firewall
  https://help.ubuntu.com/community/UFW
  https://wiki.ubuntu.com/UncomplicatedFirewall
  https://help.ubuntu.com/lts/serverguide/firewall.html
- https://wiki.itcollege.ee/index.php/Iptables
- Wikipedia - Comparison of firewalls
  http://en.wikipedia.org/wiki/Comparison_of_firewalls
- Netfilter and iptables http://www.netfilter.org/
- Debian firewalls https://wiki.debian.org/Firewalls
- next generation firewalls https://en.wikipedia.org/wiki/Next-Generation_Firewall
- Design and Performance of the OpenBSD Stateful Packet Filter (pf)
  http://www.benzedrine.cx/pf-paper.html
- MS Windows firewall https://wiki.itcollege.ee/index.php/Windowsi_tulem%C3%BC%C3%BCr
- VPN https://wiki.itcollege.ee/index.php/Virtuaalsed_privaatv%C3%B5rgud

attack rejection
- sshguard http://www.sshguard.net/
- fail2ban http://www.fail2ban.org/

alternatives:
- http://alternativeto.net/software/sshguard/
- http://alternativeto.net/software/fail2ban/

Questions?

Thank you for your attention!