# Network concepts introduction
# &
# wireshark

# W0RKSH0P

@KirilsSolovjovs

# Why am I doing this?

- Many people attending hacker conferences are not in fact experts, but come here to learn and have fun
  - Opportunity to learn something new
- Those who are experts may well not be experts at networking
  - Widening your area of interest
- Lack of understanding of basic principles of operation
  - forbids you to fully understand how attacks are carried out
  - impedes your ability to invent novel ideas and techniques

# What will we learn about?

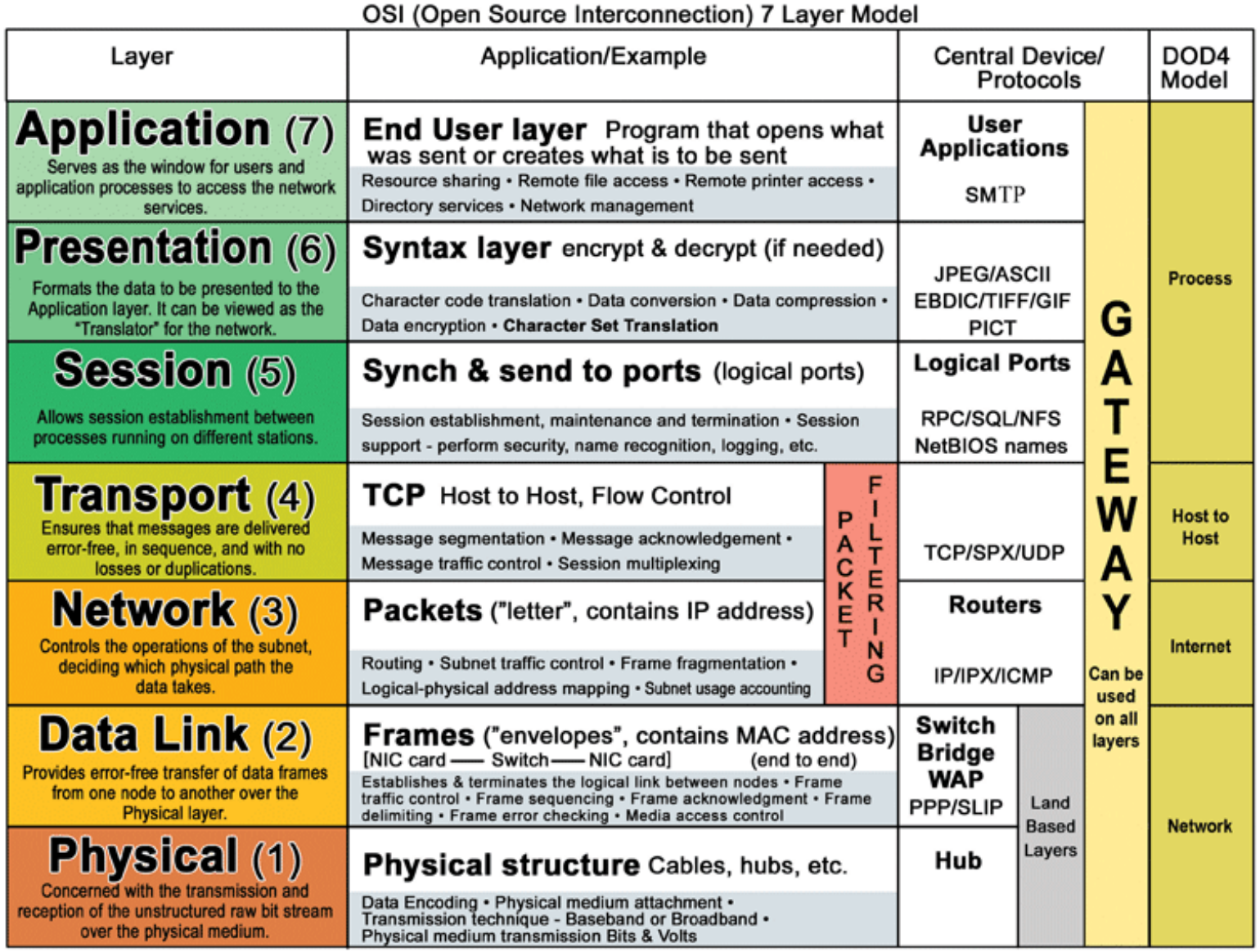
- Network layer models
- Ethernet, WiFi
- Layer3: ARP, ICMP, IPv4, IPv6
- Layer4: UDP, TCP
- Routing
- Application level protocols: DNS, SMTP, FTP, HTTP, …
- Punching holes in firewalls, breaking WPA2 and much more

# How is this different?
## (from other networking courses)

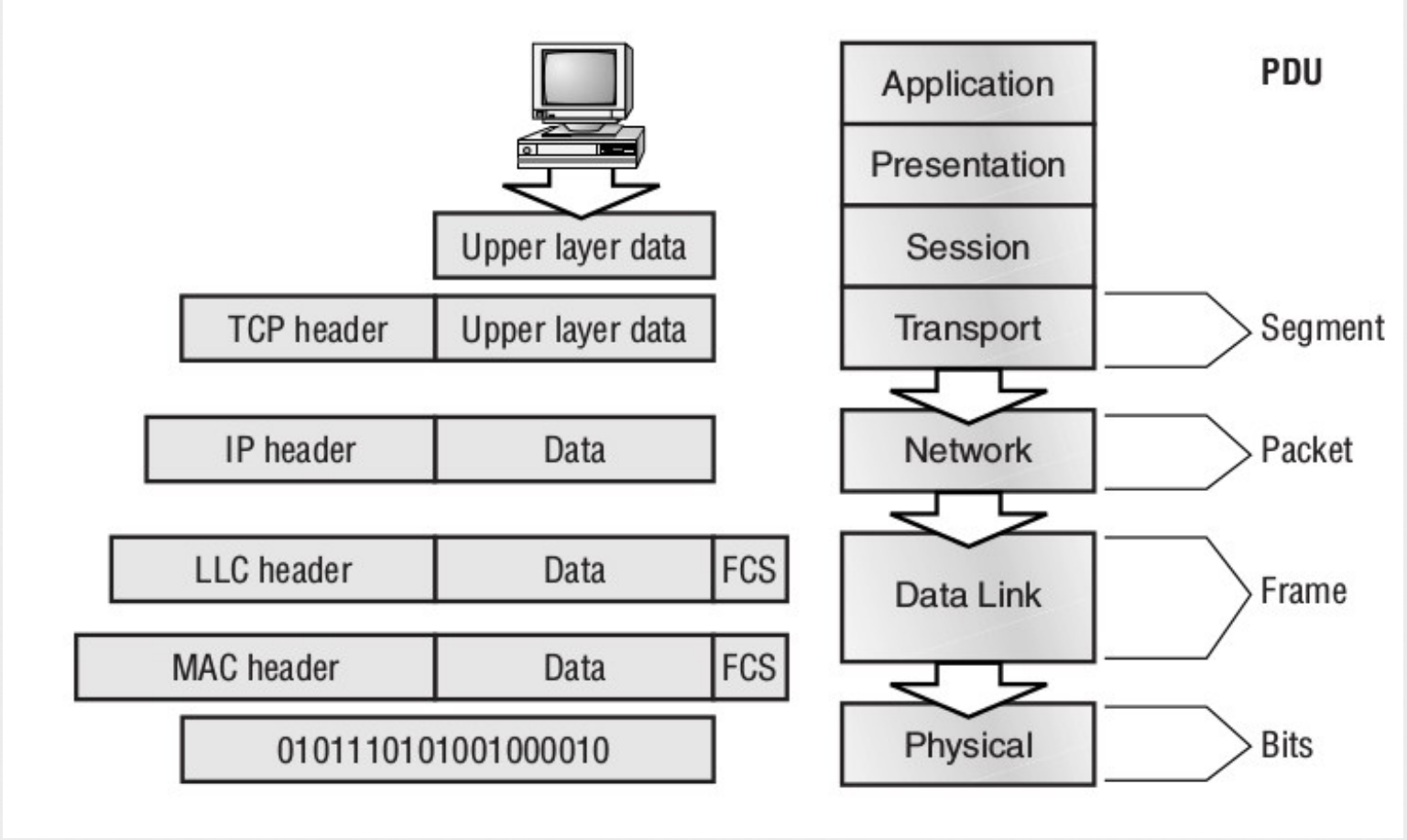
- We'll be taking the academic approach and talking a lot:
  - about what we see
  - about why stuff happens
- We'll be taking the hacker approach and start the other way around: with the hands-on
- “Shoot first, ask questions later”

# Getting to know wireshark

# ISO/OSI+DoD model



# Encapsulation



# Physical layer

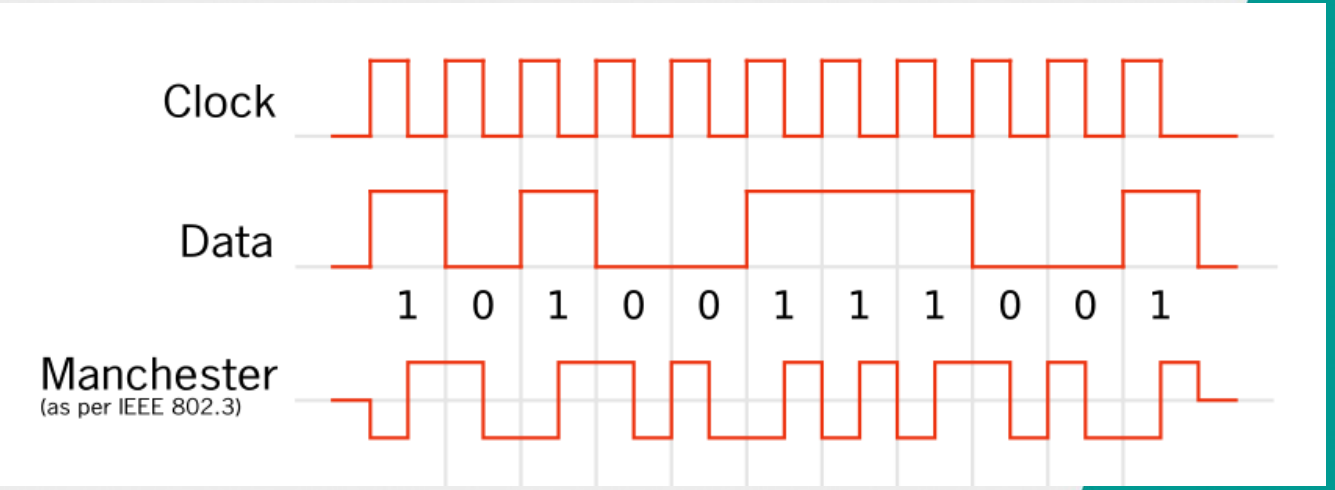
- Specifies the electrical, mechanical, procedural, and functional requirements for activating, maintaining, and deactivating a physical link between end systems.

# Data Link Layer

- Delivers messages to the proper device.
- Formats the message into data frames and adds a header containing the hardware destination and source address
  - Ethernet = MAC addresses (6 bytes)
- Consists of two parts:
  - Media Access Control
  - Logical Link Control

# Ethernet

- e.g. Manchester encoding
- MAC addresses = 6 bytes
- First 3 bytes = OUI
  - Organizationally Unique Identifier assigned by the IEEE
  - First byte usually xxxxxx00
- Last 3 bytes = Vendor assigned



# WiFi standards

| Standard | Year | Frequency | Bandwidth | Modulation | Speeds |
|---|---|---|---|---|---|
| 802.11-1997 | 1997 | 2.4 GHz | 22 MHz | DSSS & FHSS | 1 – 2 Mbps |
| 802.11a | 1999 | 5 GHz | 20 MHz | OFDM | 6 – 54 Mbps |
| 802.11b | 1999 | 2.4 GHz | 22 MHz | DSSS | 1 – 11 Mbps |
| 802.11g | 2003 | 2.4 GHz | 20 MHz | OFDM | 6 – 54 Mbps |
| 802.11n | 2009 | 2.4 & 5 GHz | 40 MHz | MIMO-OFDM | 7.2 – 135 Mbps |
| 802.11ac | 2013 | 5 GHz | 160 MHz | MIMO-OFDM | 7.2 – 780 Mbps |
| 802.11ad | 2012 | 60 GHz | 2.16 GHz | OFDM | 626 – 6756.75 Mbps |

# WiFi security

- no encryption
- WEP
- WPA
- WPA2
- 802.1x

# Network layer

- Responsible for addressing and routing between devices that are not locally attached.

# ARP

- Address Resolution Protocol allows to find the hardware address of a host from a known IP address.
- 10.0.1.254 → 00:c0:3a:21:11:99

# ICMP

- ICMP is a management protocol and messaging service provider for IP.
- e.g.
  - Destination unreachable
  - TTL exceeded
  - echo request and echo reply

# IP

- Internet Protocol checks the destination address of each packet, and, using a routing table, decides where a packet is to be sent next, choosing the best path.
- IP addresses are assigned in a hierarchical system
- Network part and host part
- IPv4 vs IPv6
  - NB! Addresses are by far not the only difference between IPv6 and IPv4.

# IPv4 addresses

- 4 bytes, e.g. 203.0.113.237
- Classes:
  - A 1.0.0.0 to 126.255.255.255
  - B 128.0.0.0 to 191.255.255.255
  - C 192.0.0.0 to 223.255.255.255
  - D 224.0.0.0 to 239.255.255.255
    - multicast
  - E 240.0.0.0 to 254.255.255.255
    - r&d

# IPv4 addresses (cont.)

- CIDR notation
- All “1” = “all” networks/nodes
- All “0” = “this” network/host
- 0.0.0.0 – default route
- 127.0.0.1 – loopback
- 255.255.255.255 – all nodes on the current network (broadcast)

# Private IPv4 address space

- 10.0.0.0 to 10.255.255.255
- 172.16.0.0 to 172.31.255.255
- 192.168.0.0 to 192.168.255.255
- Can be used with NAT
  - Network address translation intended to help limit the effects of IPv4 address exhaustion

# IPv6

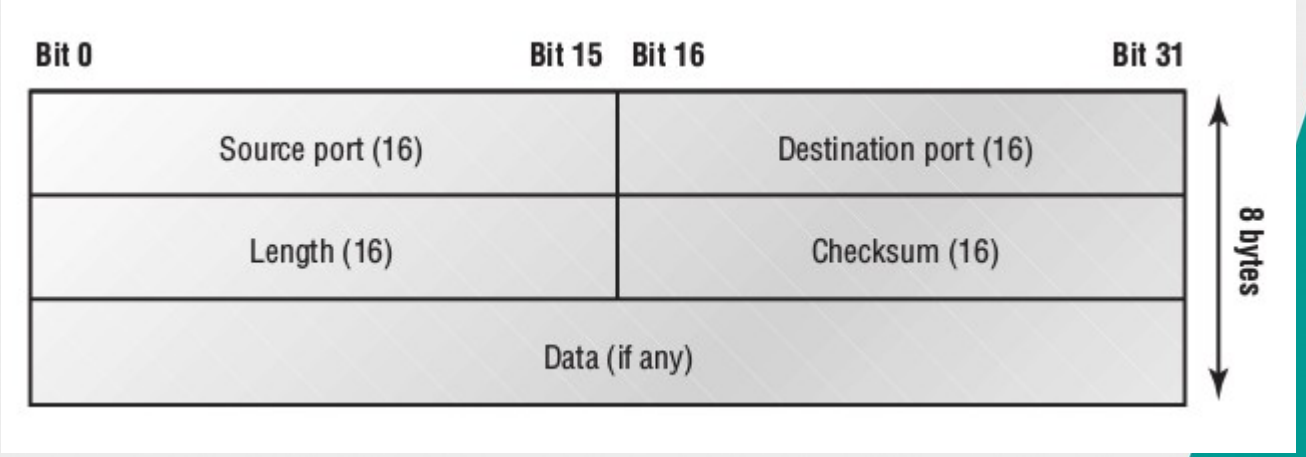
- IPv6 essentially creates a parallel, independent Layer3 network.
- 340282366920938463463374607431768211456 addresses
- 2001:14d8:ffa2:0000:0000:0000:0312:7007
- 2001:14d8:ffa2::312:7007

# Transport layer

- Responsible for the reliable transfer of data, by ensuring that data arrives at its destination error-free and in order.
  - Connection-oriented – requires that a connection with specific agreed-upon parameters be established before data is sent.
  - Connectionless – requires no connection before data is sent.

# User Datagram Protocol

- Stateless, transaction-oriented
- "Best effort" transport
- Notable features include:
  - Minimalist design
  - No control
  - No retransmissions

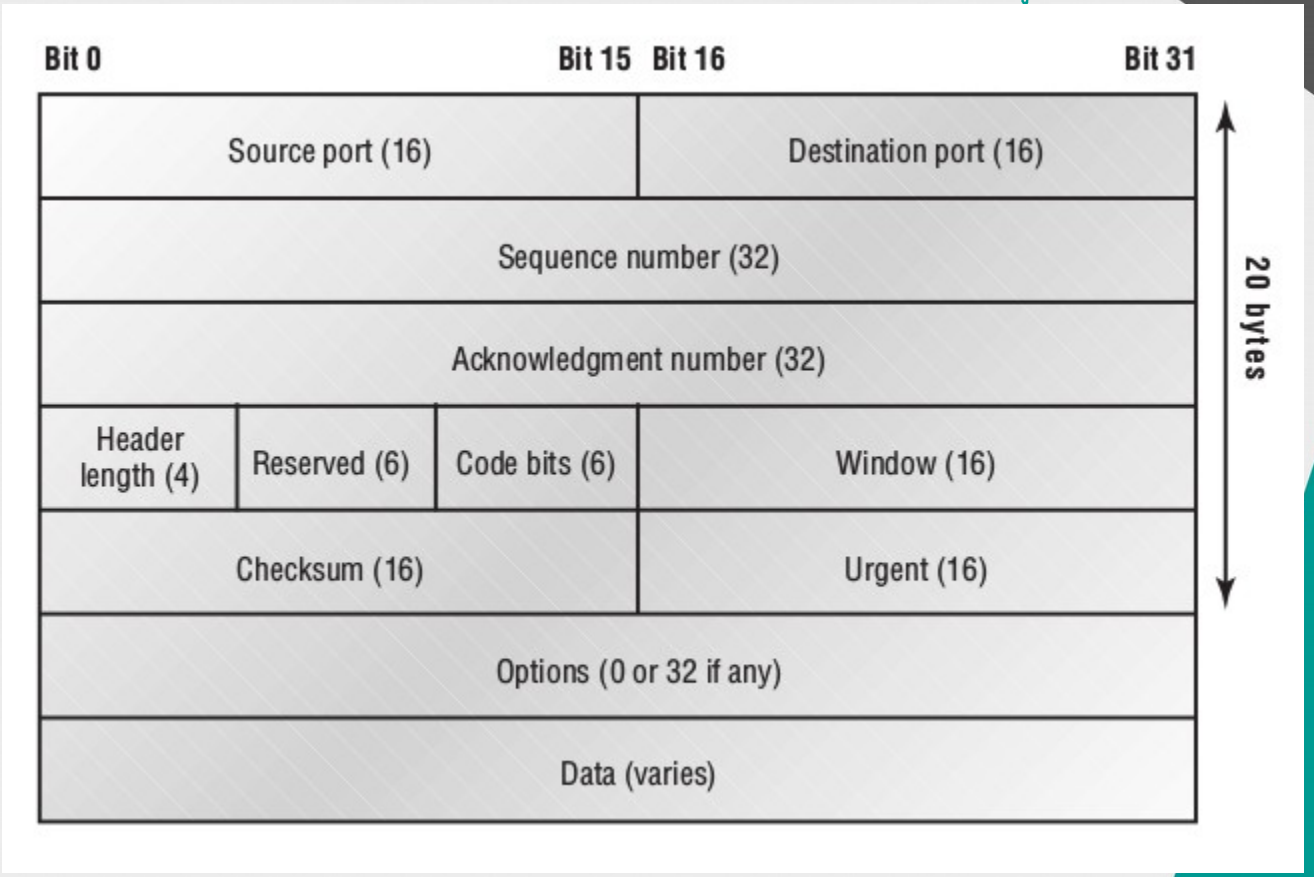


# Fun demo

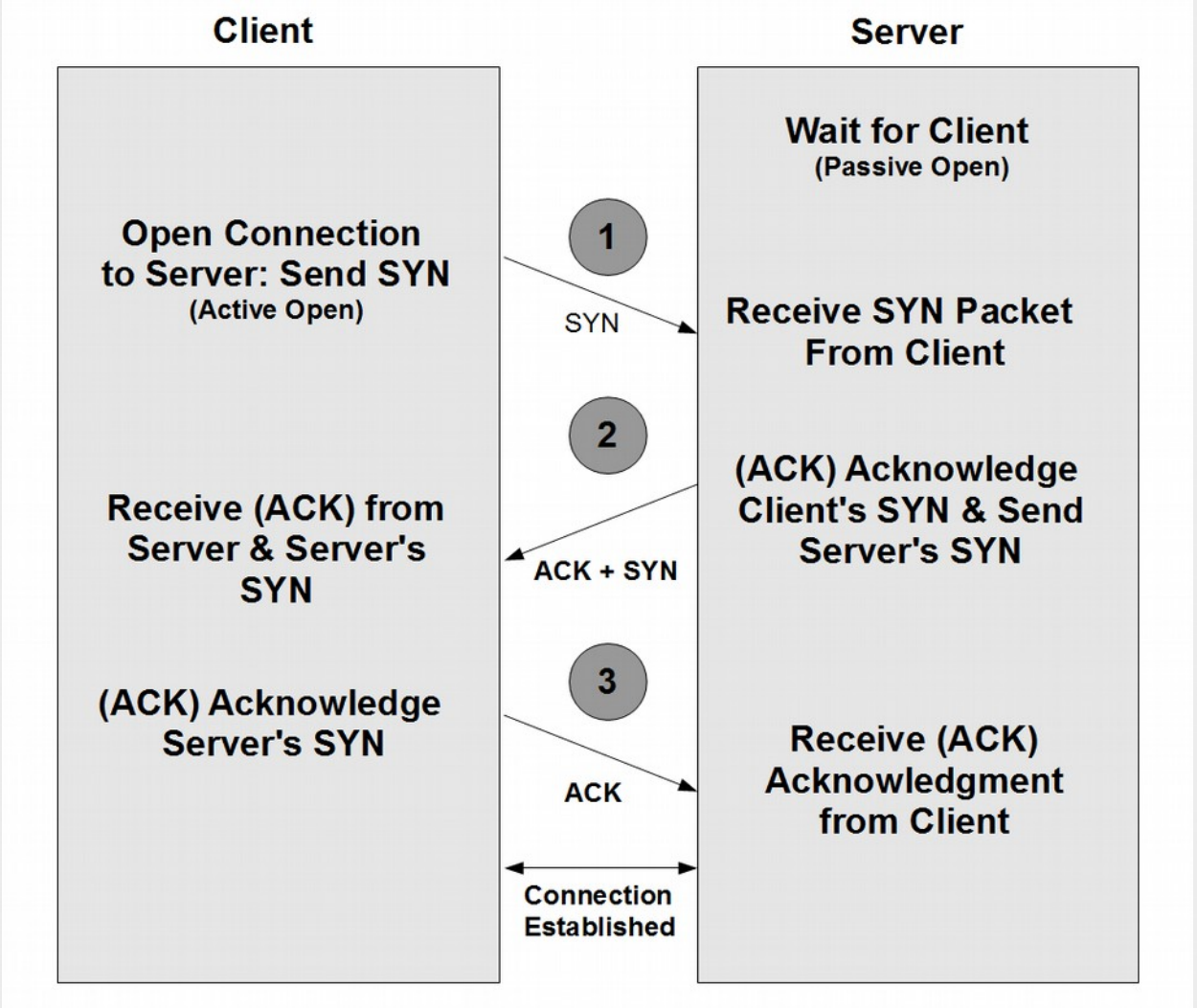
- Punching holes in NAT routers via UDP

# Transport Control Protocol

- Stateful, connection-oriented
- "Reliable" transport
- Notable features include:
  - 3-way handshake
  - Error detection
  - Ordered transfer
  - Flow control



# Three-way handshake



# zmap

- Modular and open-source network scanner specifically designed for Internet-wide scans
- Scans the whole IPv4 address space in 45 minutes (1Gbps)
- How does it work?

# Routing

- TTL decreased with every hop
- Routing decisions taken based on the routing table and route distance
- Routing types
  - Static routing
  - Default routing
  - Dynamic routing

# Static routing

- Manually setting up routes on each router
- Does not scale well

# Default routing

- Used to send packets having a destination address in a remote network not in the routing table to the next hop router.

# Dynamic routing

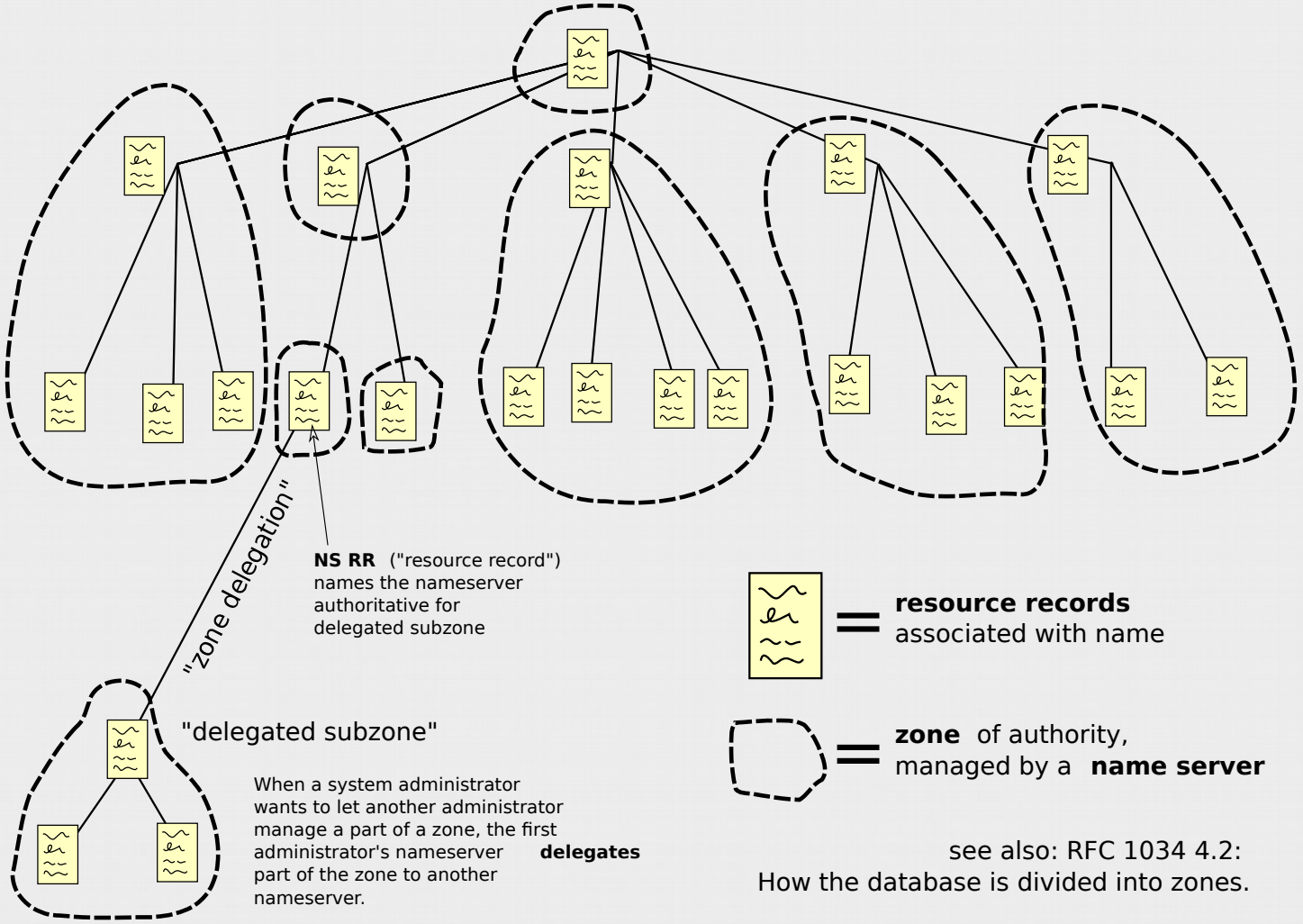
- Dynamically updates routing tables on the router using routing protocols:
  - distance-vector protocols determine the route with the least number of hops to be the best route
  - RIP, IGRP, etc.
  - link state protocols (also called shortest path first) use additional metrics and recreate the topology representation on each router; e.g. they can take congestion into account
  - OSPF, etc.

# Application level protocols

- DNS
- SMTP
- FTP
- HTTP
- …

# DNS overview

## Domain Name Space



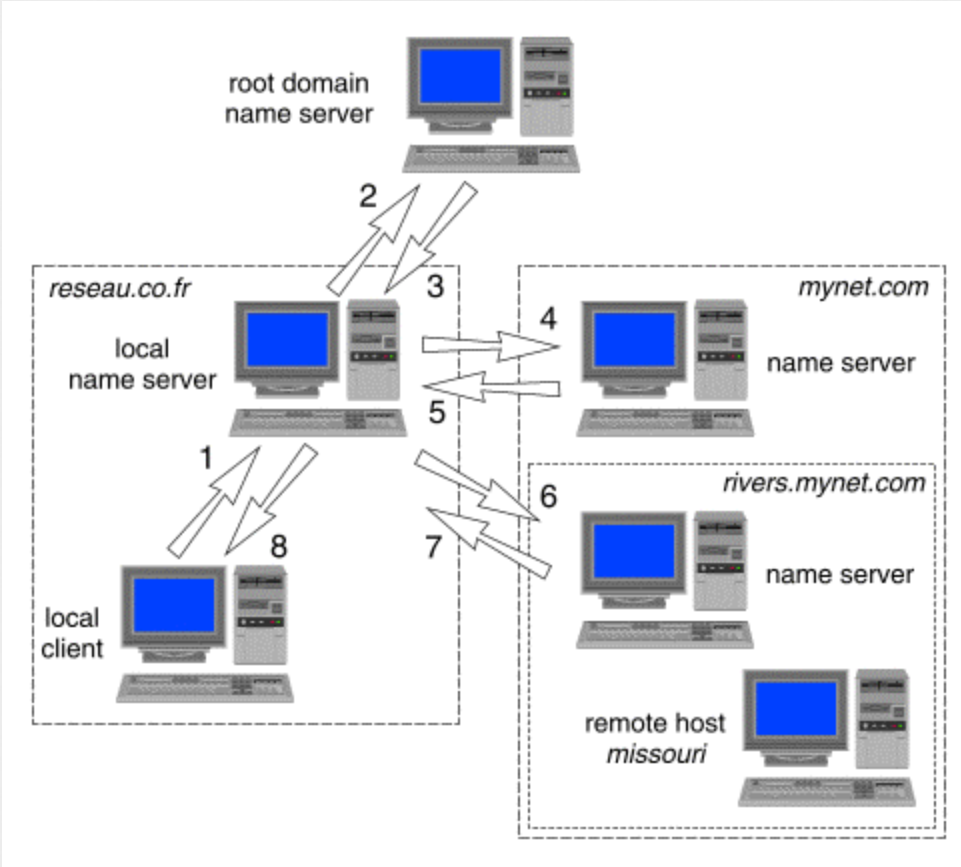

see also: RFC 1034 4.2:
How the database is divided into zones.

# (some) DNS record types

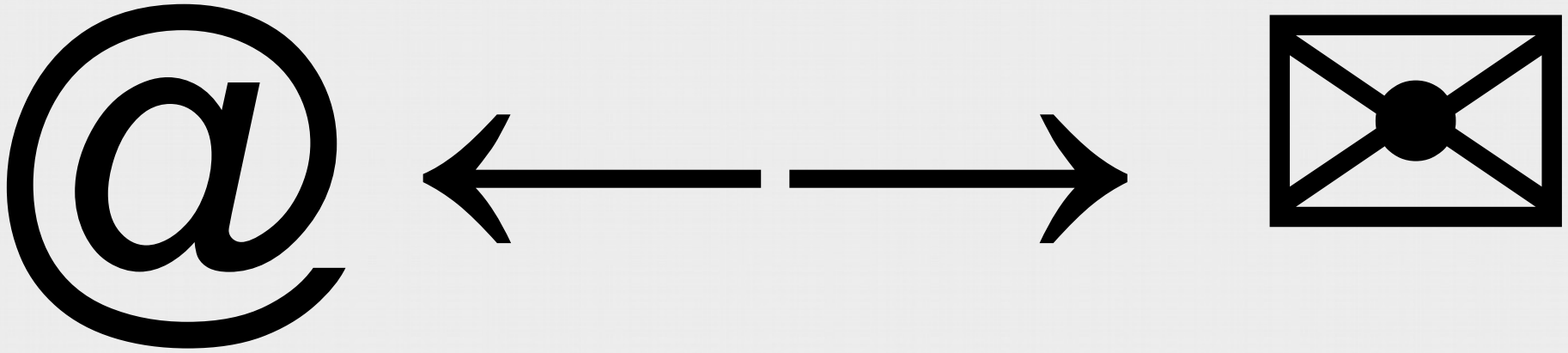
- A / AAAA– Address
  - Returns an IP address
- MX – Mail exchange
  - Maps a domain name to a list of message transfer agents
- NS – Name server
  - Delegates a DNS zone to use the given authoritative name servers
- PTR – Pointer
  - Pointer to a canonical name
  - Unlike a CNAME, DNS processing stops and just the name is returned

# DNS queries

- *dig @nameserver domain record-type +trace*
- dig PCH.RCP.pe ANY
  - pseudo-record – self explanatory
- dig @ns.example.com example.com AXFR
  - pseudo-record – authoritative transfer



# SMTP
## Simple Mail Transfer Protocol



# SMTP protocol

- 220 mail.example.org ESMTP Sendmail; Fri, 15 Jan 2016 16:27:08 +0000

- HELO relay.example.org
  - 250 mail.example.org Hello relay.example.org [192.168.2.3] (may be forged), pleased to meet you
- MAIL FROM: <alice@example.org>
  - 250 2.1.0 alice@example.org... Sender ok
- RCPT TO: <bob@example.com>
  - 250 2.1.5 bob@example.com... Recipient ok

# SMTP protocol

- DATA
  - 354 Enter mail, end with "." on a line by itself
- From: "Alice Alice" <alice@example.com>

  To: "Bob Bob" <bob@example.org>

  Date: Fri, 15 Jan 2016 16:27:03 +0000

  Subject: Test e-mail

  Testing.

  .

# SMTP protocol

- 250 2.0.0 vB3DJ2cP000123 Message accepted for delivery

- QUIT
  - 221 2.0.0 mail.example.org closing connection

# FTP

- 220 Hello, this is the Acme FTP server.

- USER username
  - 331 Password required to access user account username.
- PASS A6Va2MkOOL
  - 230 Logged in.
- CWD data
  - 250 "/home/username/data" is new working directory.

# FTP

- PORT 192,168,1,2,7,138
  - 200 PORT command successful.
- LIST
  - 150 Opening ASCII mode data connection for /bin/ls.
  - 226 Listing completed.

# FTP

- PORT 192,168,1,2,7,139
  - 200 PORT command successful.
- RETR information.txt
  - 150 Opening ASCII mode data connection for information.txt.
  - 226 Transfer completed.
- QUIT
  - 221 Goodbye.

# HTTP request

GET /page HTTP/1.1

Host: example.com

User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/50.0

Accept: text/html,application/xhtml+xml;q=0.9,*/*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Cookie: hell=o; data=1001090933

Connection: keep-alive

# HTTP response

HTTP/1.1 200 OK

Date: Thu, 01 Aug 2016 12:02:57 GMT

Server: Apache

Content-Length: 2667

Keep-Alive: timeout=3, max=20

Connection: Keep-Alive

Content-Type: text/html

<html>

# Encrypted protocols

- TLS (Transport Layer Security) widely used
- Allows to add encryption to:
  - telnet → ssh
  - http → https
  - smtp → smtps
  - etc.

# Back to wireshark

- Step-by-step analysis of opening a webpage

# That is all folks!

- For the intro that is.
- Have a superb Congress and see you around!

- Visit me at:
  - @KirilsSolovjovs
  - kirils.org