# Sieci komputerowe

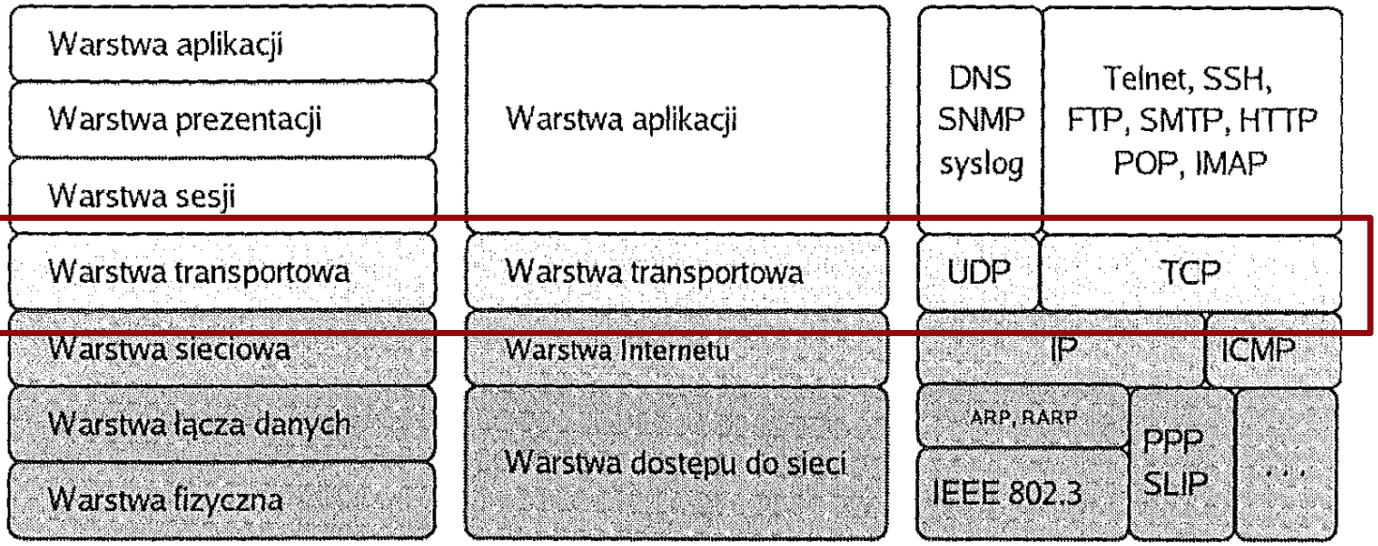
Wykład 5
13.11.2019

dr inż. Łukasz Graczykowski
lukasz.graczykowski@pw.edu.pl

*Semestr letni 2019/2020*

# Warstwa transportowa

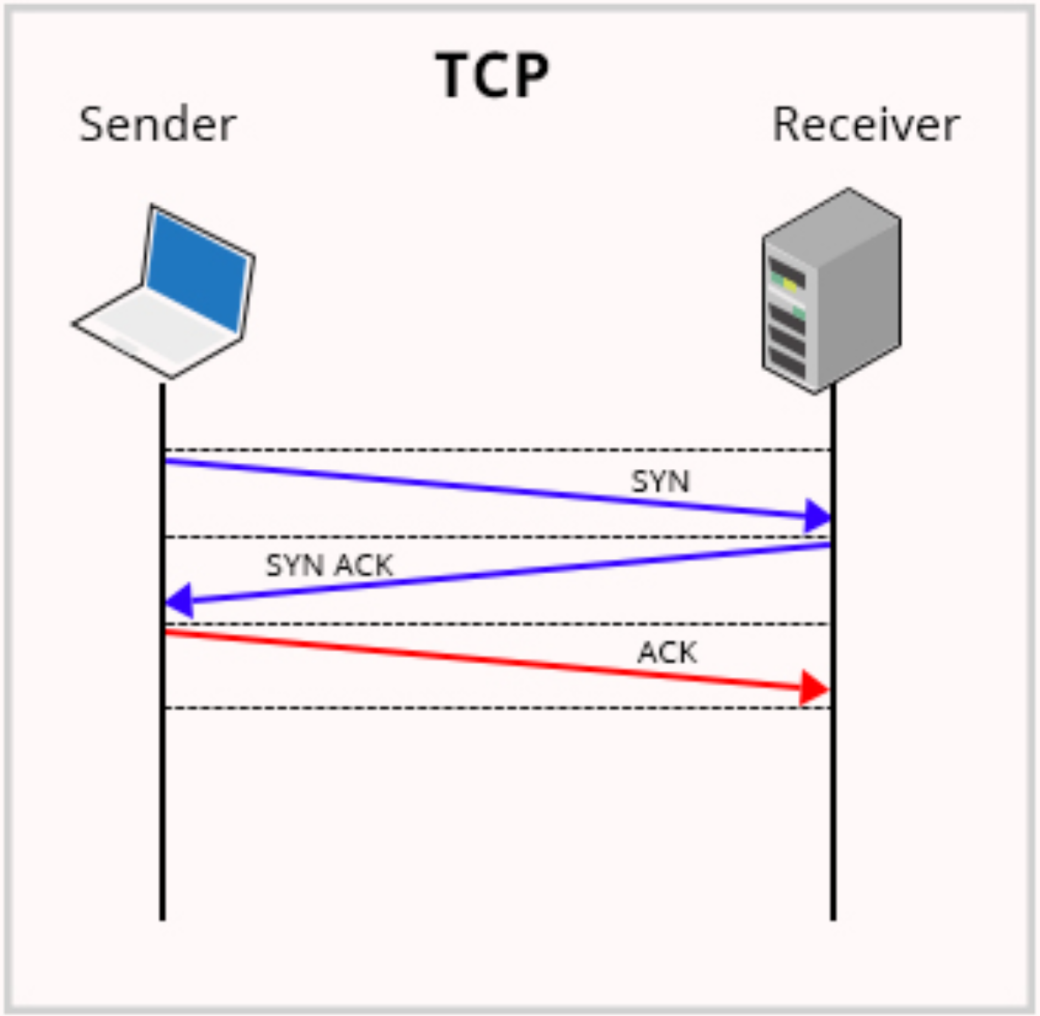
## Protokoły TCP i UDP dokończenie

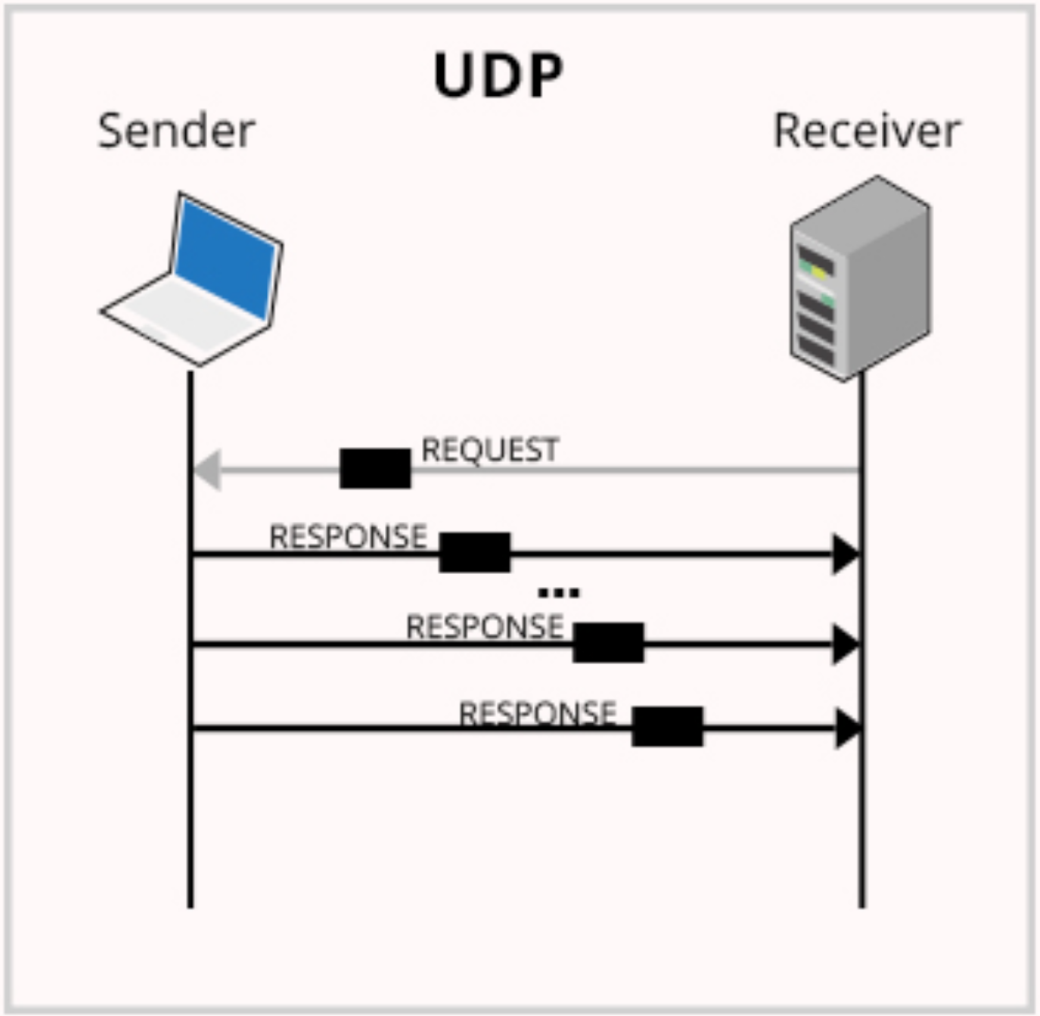



źródło: Helion

# Protokół UDP

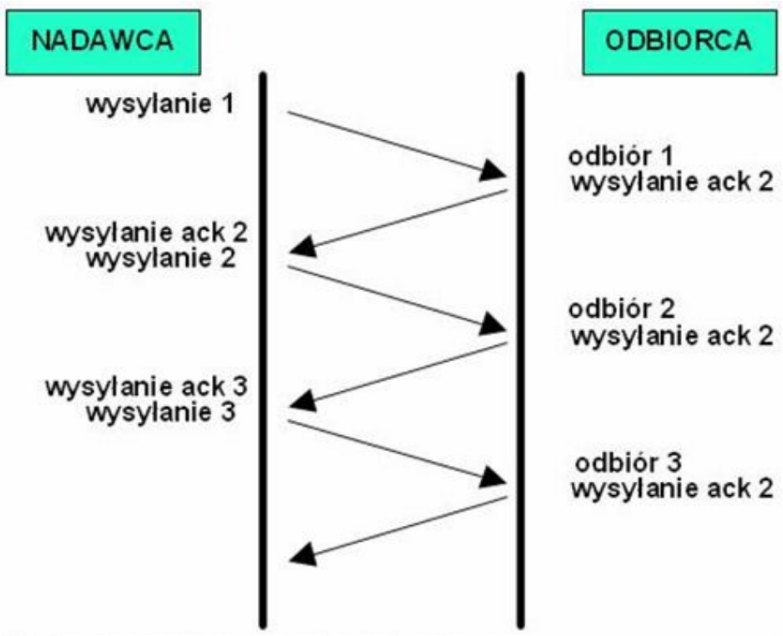
- Porównanie połączenia w TCP i “połączenia” w UDP

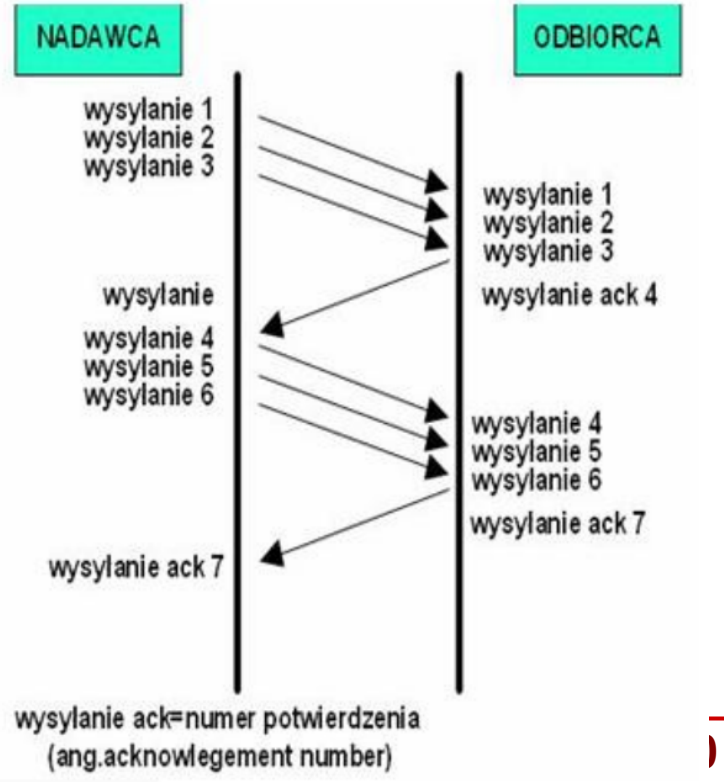


# Przesyłanie danych w TCP

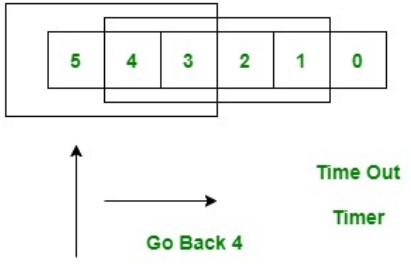
- Teoretycznie możemy sobie wyobrazić następujący scenariusz:
  - po każdym wysłaniu jednego segmentu danych odbiorca potwierdza otrzymanie danych
  - jest to jednak **mało efektywne**
- Rozwiązanie – **buforowanie** danych (przechowywanie danych po stronie klienta i serwera)

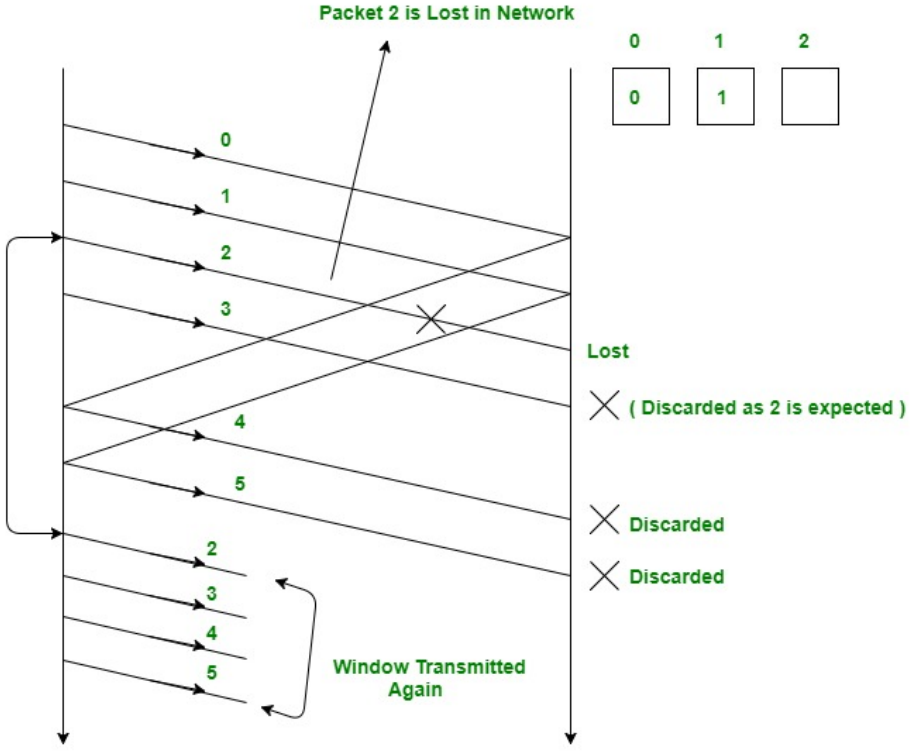


# Przesyłanie danych w TCP

- Mechanizm **przesuwnego okna** (*sliding window*), wysyłanie okna zawierającego konkretne segmenty
- Wysyłamy segmenty odpowiadające rozmiarowi okna bez potwierdzania każdego z nich przed wysłaniem kolejnego
- Powtarzamy wysłanie okna, gdy nie któryś segment nie dojdzie w założonym czasie (**timeout**)
- **Kontrola przepływu**:
  - zmienny rozmiar okna
  - blokowanie nadawcy by nie przeciążyć bufora



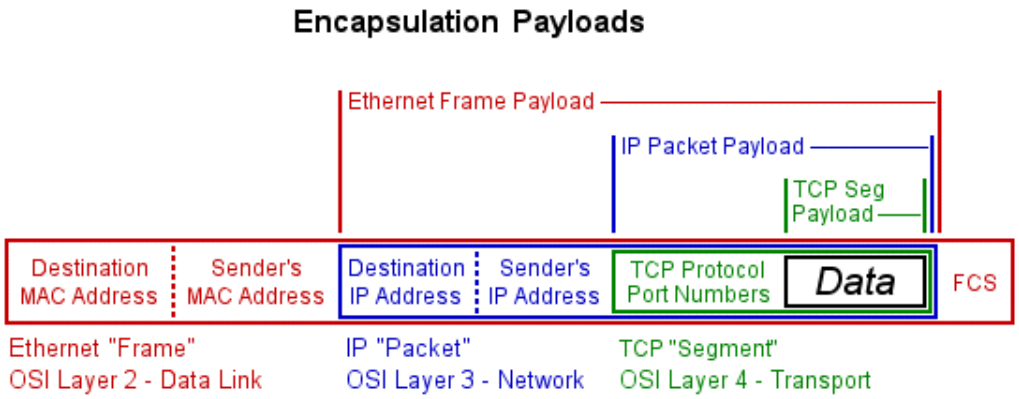



# Porównanie UDP i TCP

| Cecha | UDP | TCP |
|---|---|---|
| Opis | Prosty protokół dużych przepustowości (przeniesienie funkcjonalności na warstwy wyższe) | W pełni funkcjonalny, niezawodny protokół komunikacyjny z mechanizmami obsługi błędów warstwy sieciowej |
| Ustanawianie połączenia | bezpołączeniowy | Połączeniowy, faza nawiązania połączenia |
| Interfejs danych dla aplikacji | Zorientowany na wiadomości | Zorientowany strumieniowo |
| Wiarygodność i potwierdzenia | Zawodny, bez potwierdzeń | Niezawodny, wymaga potwierdzeń dostarczenia datagramów |
| Retransmisje | Nie obsługiwane (przeniesione do warstw wyższych) | Obsługiwane automatycznie |
| Kontrola przepływu | Brak | Okno przesuwne zmiennych rozmiarów, mechanizmy zapobiegania przeciążeniom |
| Narzut | Bardzo mały | Mały |
| Prędkość transmisji | Bardzo duża | Duża |
| Typ danych (wielkość, rozmiar) | od małych do średnich | Od małych do bardzo dużych |

# Enkapsulacja danych

- Przykład enkapsulacji danych
- ramka Ethernet → datagramIPv4 → segment TCP

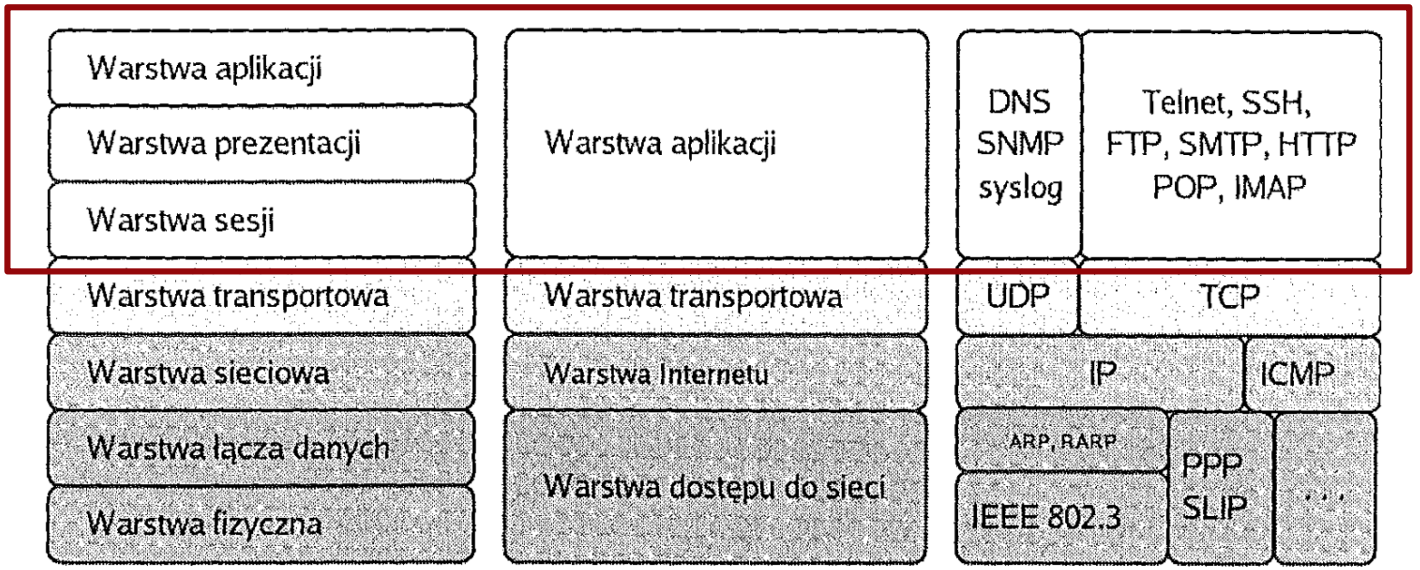



TCP segment in IPv4 packet in Ethernet frame

| Ethernet | Octets | | | | |
|---|---|---|---|---|---|
| Preamble | 7 | | | | |
| Start of frame delimiter | 1 | | | | |
| MAC destination | 6 | | | | |
| MAC source | 6 | | | | |
| 802.1Q tag (opt.) | 4 | | | | |
| Ethertype or length | 2 | IPv4 | Bits | | |
| Payload | 46 -1500 | Version | 4 | | |
| | | Header Length | 4 | | |
| | | Differentiated Services Code Point | 6 | | |
| | | Explicit Congestion Notification | 2 | | |
| | | Total Length | 16 | | |
| | | Identification | 16 | | |
| | | Flags | 3 | | |
| | | Fragment Offset | 13 | | |
| | | Time to Live | 8 | | |
| | | Protocol | 8 | | |
| | | Header Checksum | 16 | | |
| | | Source IP Address | 32 | | |
| | | Destination IP Address | 32 | | |
| | | Options ( if Header Length > 5 ) | ? | | |
| | | Payload | 1440-1480 Bytes | TCP | Bits |
| | | | | Source Port | 16 |
| | | | | Destination Port | 16 |
| | | | | Sequence number | 32 |
| | | | | Acknowledgment number | 32 |
| | | | | Data offset | 4 |
| | | | | Reserved | 4 |
| | | | | Flag | 8 |
| | | | | Window Size | 16 |
| | | | | Checksum | 16 |
| | | | | Urgent pointer | 16 |
| | | | | Options (if Data Offset > 5) | varies |
| | | | | padding | 8 |
| | | | | Payload | Payload |
| CRC | 4 | | | | |
| Interframe gap | 12 | | | | |

# Warstwa aplikacji

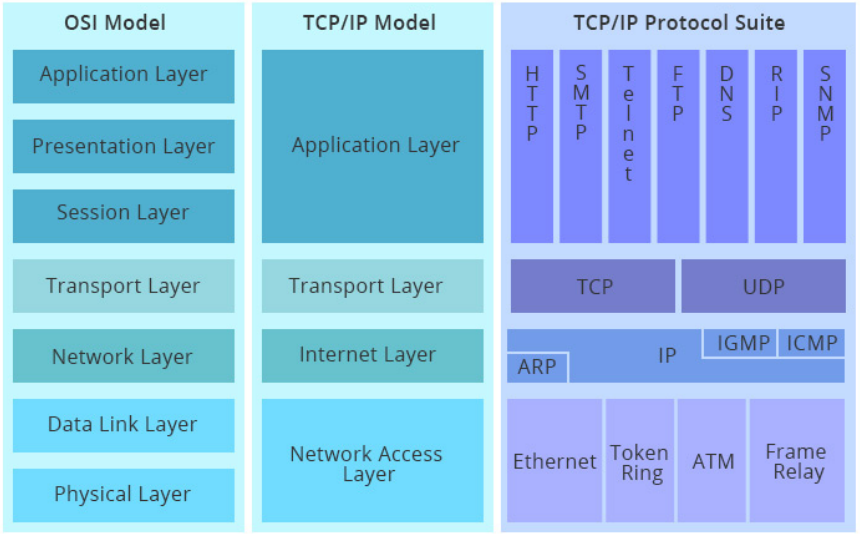
# Usługi sieciowe

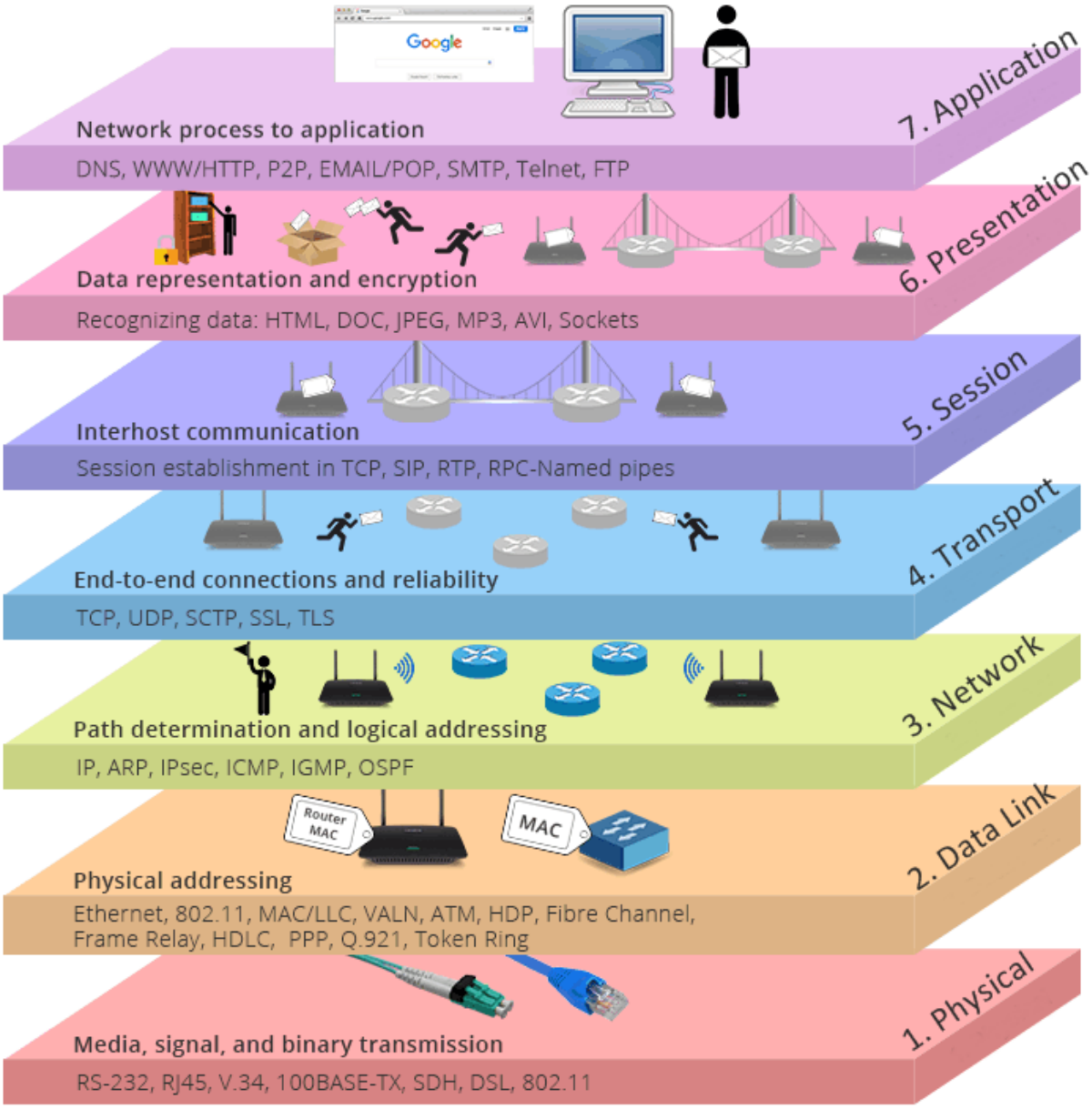



źródło: Helion

# Warstwa aplikacji

- W modelu ISO/OSI wyróżniamy w zasadzie trzy górne warstwy:
  - warstwa sesji
  - warstwa prezentacji
  - warstwa aplikacji
- W praktyce to aplikacje TCP(UDP)/IP realizują zadania ostatnich 3 warstw



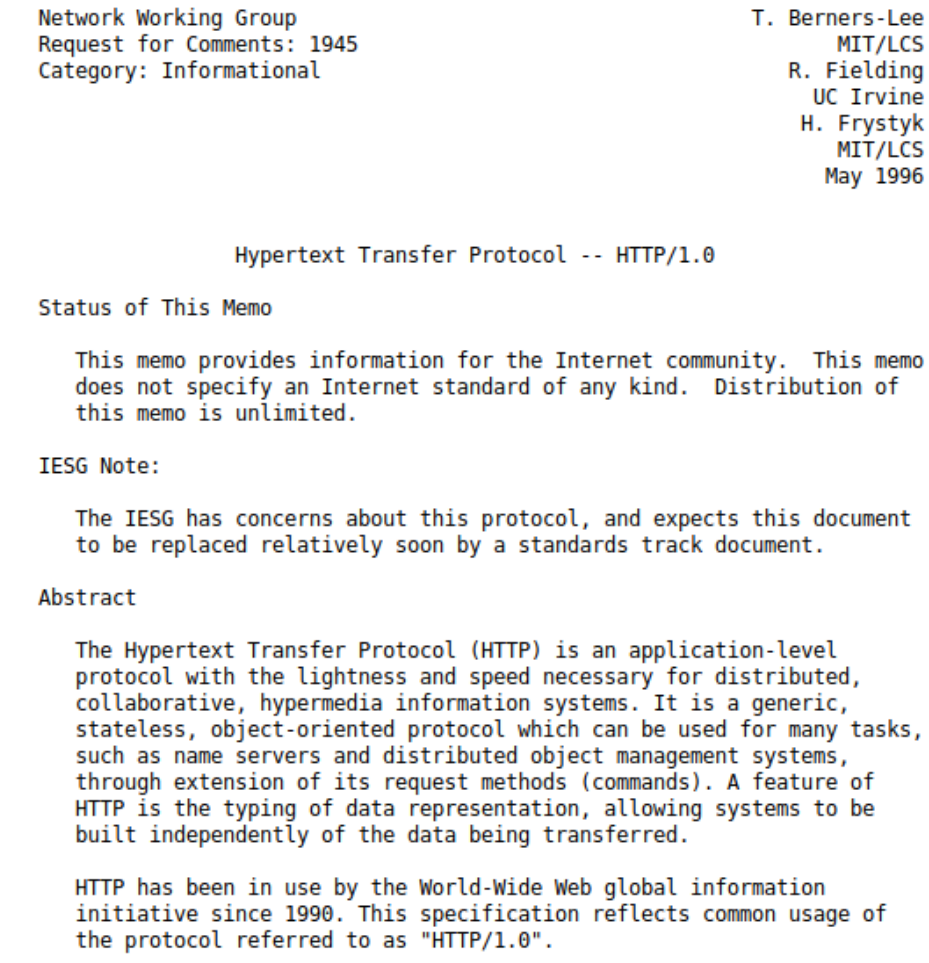



# Warstwa aplikacji

- W ostatniej warstwie działają protokoły komunikacji, które używane są przez finalne aplikacje komunikujące się z użytkownikiem
- Część z nich już poznaliśmu (np. DNS czy DHCP)
- Protokoły zapewniają możliwość korzystania z różnych usług dostępnych w sieci, np.:
  - WWW/HTTP – przeglądanie stron internetowych
  - FTP – serwery plików
  - SMTP – wysyłanie/odbieranie e-mail'i
  - POP oraz IMAP – odczytywanie e-maili z serwera przez aplikacje klienckie
  - SSH – bezpieczne logowanie
  - LDAP – usługi katalogowe (np. wspólne usługi drukarek, logowanie, itp.)
  - … itp./itd.

# Standaryzacja – Request for Comments

- Protokoły definiowane są przez **RFC** (*Request for Comments*)
- Utrzymywane są przez **IETF** (*Internet Engineering Task Force*)
- https://www.ietf.org/standards/rfcs/



Network Working Group
Request for Comments: 1945
Category: Informational

T. Berners-Lee
MIT/LCS
R. Fielding
UC Irvine
H. Frystyk
MIT/LCS
May 1996

Hypertext Transfer Protocol -- HTTP/1.0

Status of This Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

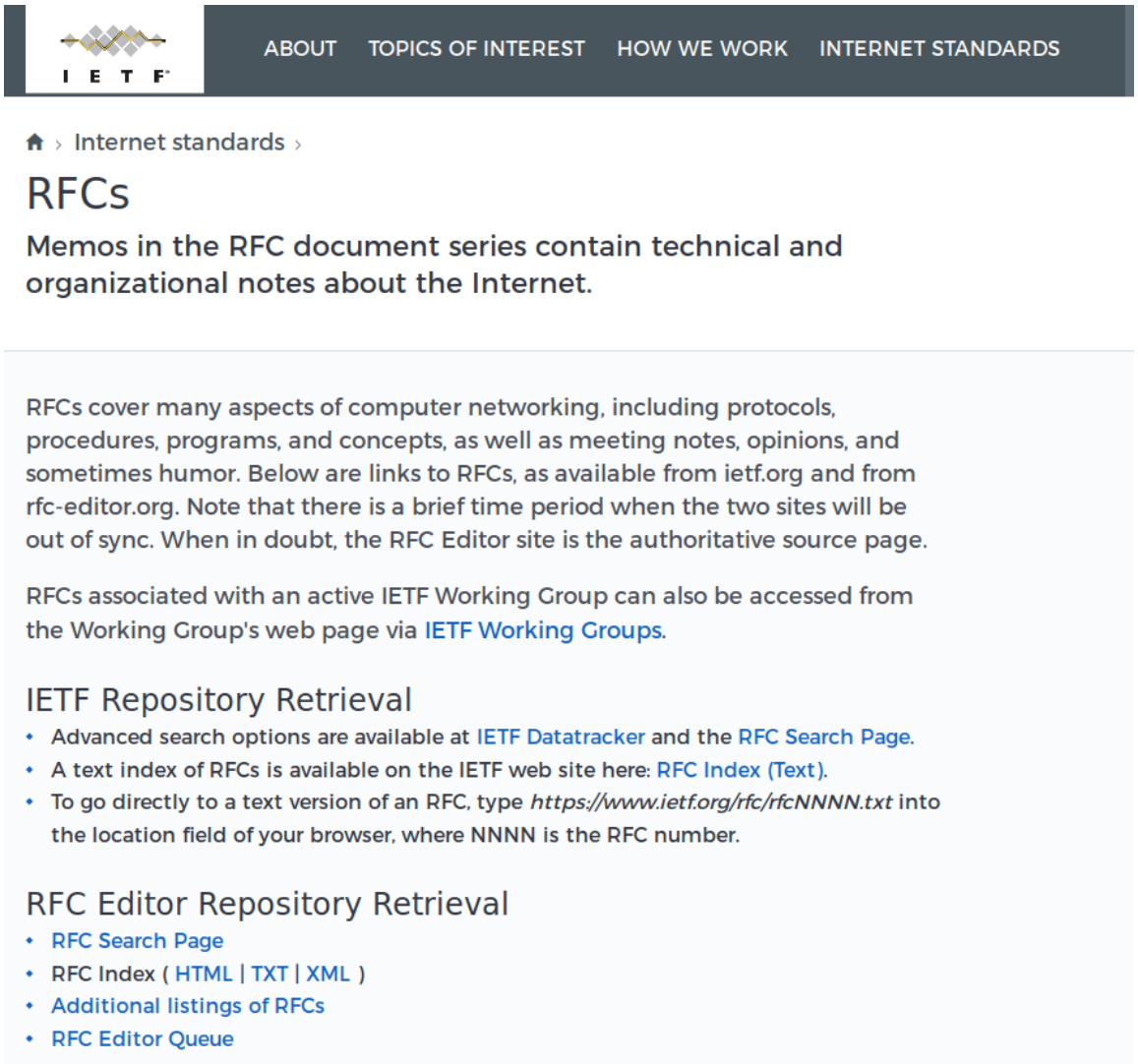
IESG Note:

The IESG has concerns about this protocol, and expects this document to be replaced relatively soon by a standards track document.

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands). A feature of HTTP is the typing of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. This specification reflects common usage of the protocol referred to as "HTTP/1.0".



IETF ABOUT TOPICS OF INTEREST HOW WE WORK INTERNET STANDARDS

Internet standards

RFCs

Memos in the RFC document series contain technical and organizational notes about the Internet.

RFCs cover many aspects of computer networking, including protocols, procedures, programs, and concepts, as well as meeting notes, opinions, and sometimes humor. Below are links to RFCs, as available from ietf.org and from rfc-editor.org. Note that there is a brief time period when the two sites will be out of sync. When in doubt, the RFC Editor site is the authoritative source page.

RFCs associated with an active IETF Working Group can also be accessed from the Working Group's web page via IETF Working Groups.

IETF Repository Retrieval

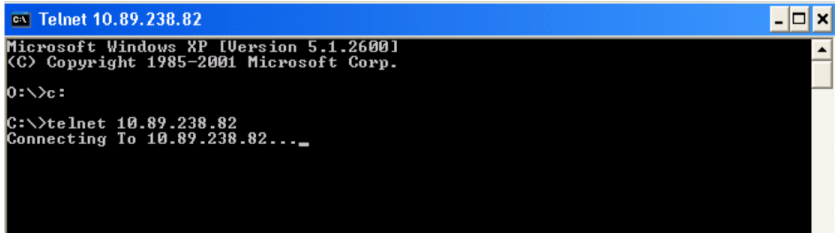
- Advanced search options are available at IETF Datatracker and the RFC Search Page.
- A text index of RFCs is available on the IETF web site here: RFC Index (Text).
- To go directly to a text version of an RFC, type *https://www.ietf.org/rfc/rfcNNNN.txt* into the location field of your browser, where NNNN is the RFC number.

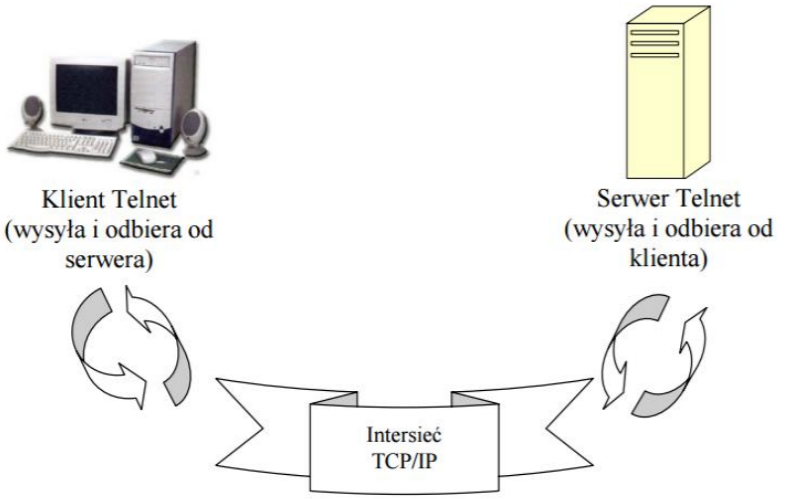
RFC Editor Repository Retrieval

- RFC Search Page
- RFC Index ( HTML | TXT | XML )
- Additional listings of RFCs
- RFC Editor Queue

# Telnet

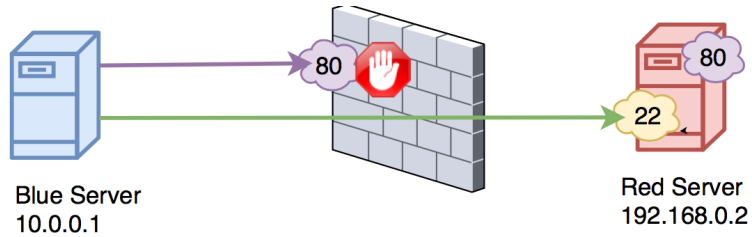
- **Telnet** (*Network Terminal Protocol*) jest protokołem zadalnego dostępu do komputera poprzez terminal, RFC 854
- Działa za pomocą protokołu TCP, *well known port* to 23
- Obsługuje jedynie terminale numeryczne w trybie tekstowym (brak GUI czy myszy)
- Działa w trybie klient-serwer → na komputerze, na który chcemy się zalogować musi działać serwe, który obsługuje tę usługę
- Wymaga posiadania konta na serwerze
- Często wykorzystywany np. do konfiguracji switczy czy routerów
- Wada – brak szyfrowania
- Obecnie zastępowany przez SSH

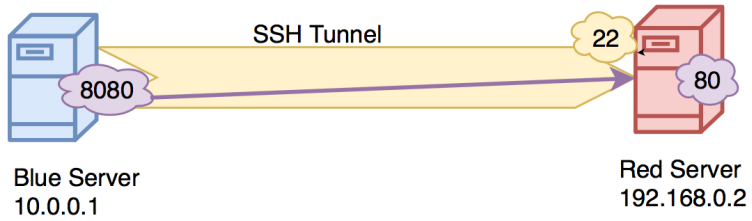


# SSH

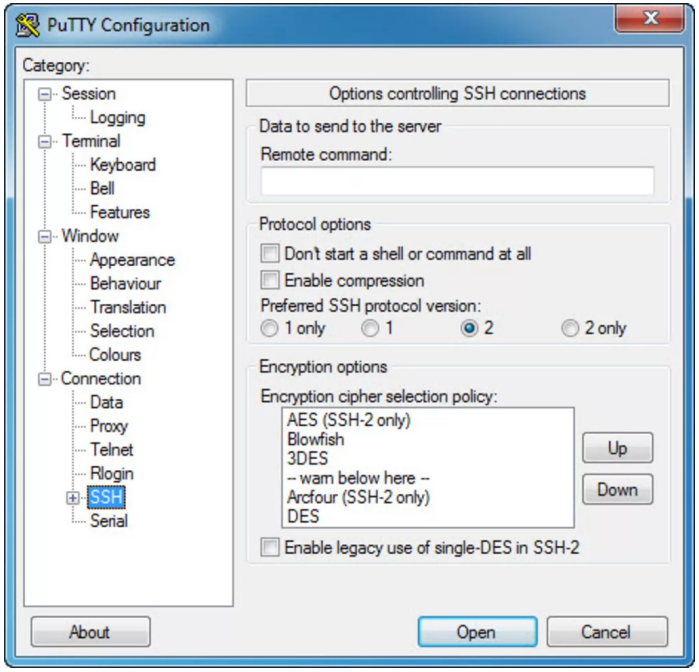
- **SSH** (*Secure Shell Login*) jest protokołem zadalnego dostępu do komputera, jak telnet, ale z wykorzystaniem szyfrowania, RFC 4253
- Działa za pomocą protokołu TCP, *well known port* to 22
- Szyfrowanie asymetryczne (hasła, klucze)
  - szyfrowane zarówno hasło jak i klucze
- Dużo więcej możliwości:
  - terminal (ssh)
  - przesyłanie plików (scp, sftp)
  - zdalna kopia (rsync)
  - tunelowanie (np. aplikacje GUI, X11)

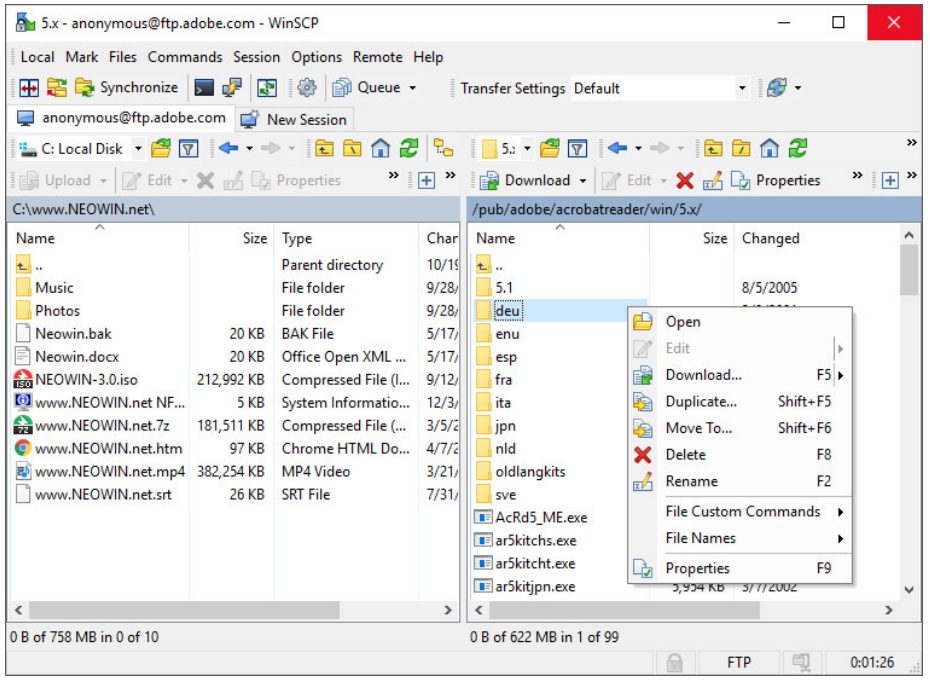



# SSH

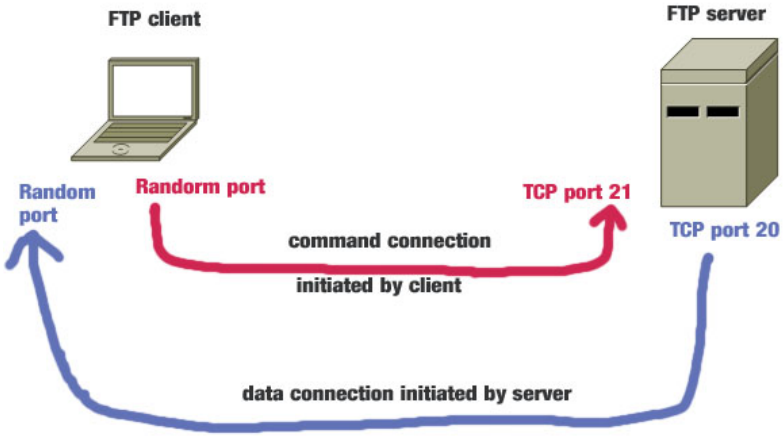
- SSH ma dwie implentacje:
  - zamknięta: ssh.com
  - otwarta – OpenSSH – obecnie używana
- Istnieją również dwie wersje (SSHv1, SSHv2)
- Aplikacje klienckie:
  - PuTTy
  - WinSCP
  - OpenSSH
  - … inne



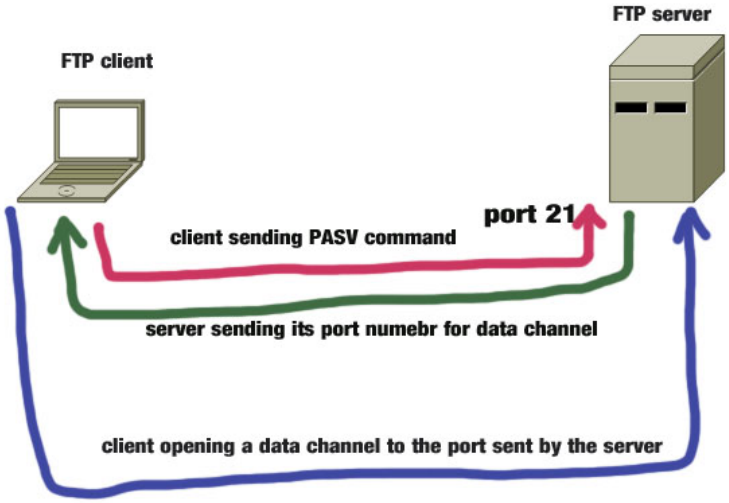



# FTP

- **FTP** (*File Transfer Protocol*) jest protokołem transferu plików, RFC.959
- Działa za pomocą protokołu TCP, *well known ports* to 20 oraz 21
- Może działać w dwóch trybach:
  - tryb aktywny:
    - port 21 – polecenia (klient → serwer)
    - port 20 – transfer danych (serwer → klient)
  - tryb pasywny:
    - port 21 – polecenia (klient → serwer)
    - port powyżej 1024 (ustalany przez klienta) – transfer danych (serwer → klient)
- Tryb dostępu:
  - anonimowy – bez hasłą uwierzytelniającego
  - autoryzowany – w oparciu o login i hasło
- Isntieją warianty z szyfrowaniem



FTP mode: Active



FTP mode: Passive

# SMTP

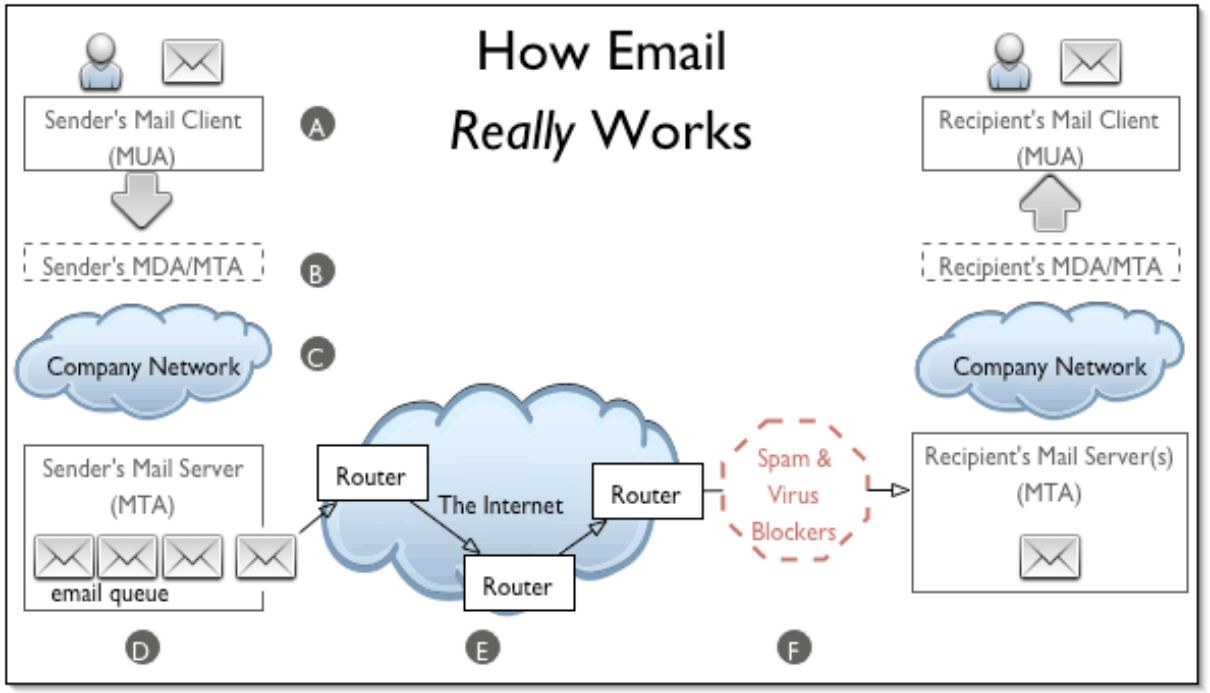
- **SMTP** (*Simple Mail Transfer Protocol*) protokół służący do przekazywania poczty elektronicznej w Internecie, RFC 821 oraz RFC 532
- *Well known port* to 25 (czasami 587) - protokół TCP
- Komunikacja serwer-serwer – wykorzystuje serwery do przesyłania wiadomości
- Protokoły klienckie (np. Outlook, Thunderbird) używają go tylko do wysyłania poczty
- Jak to działa?

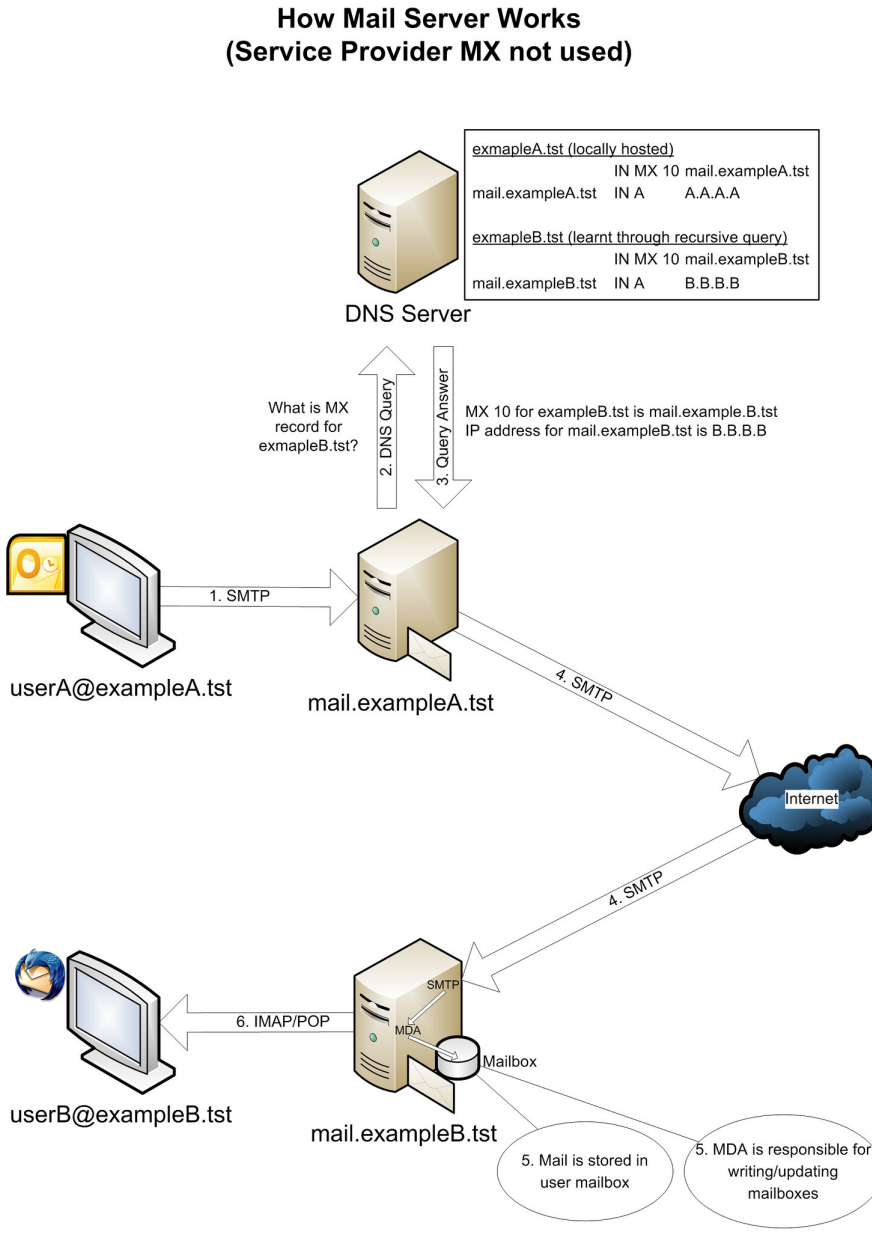
# SMTP

- Pewnie wyobrażacie sobie to tak:

# SMTP

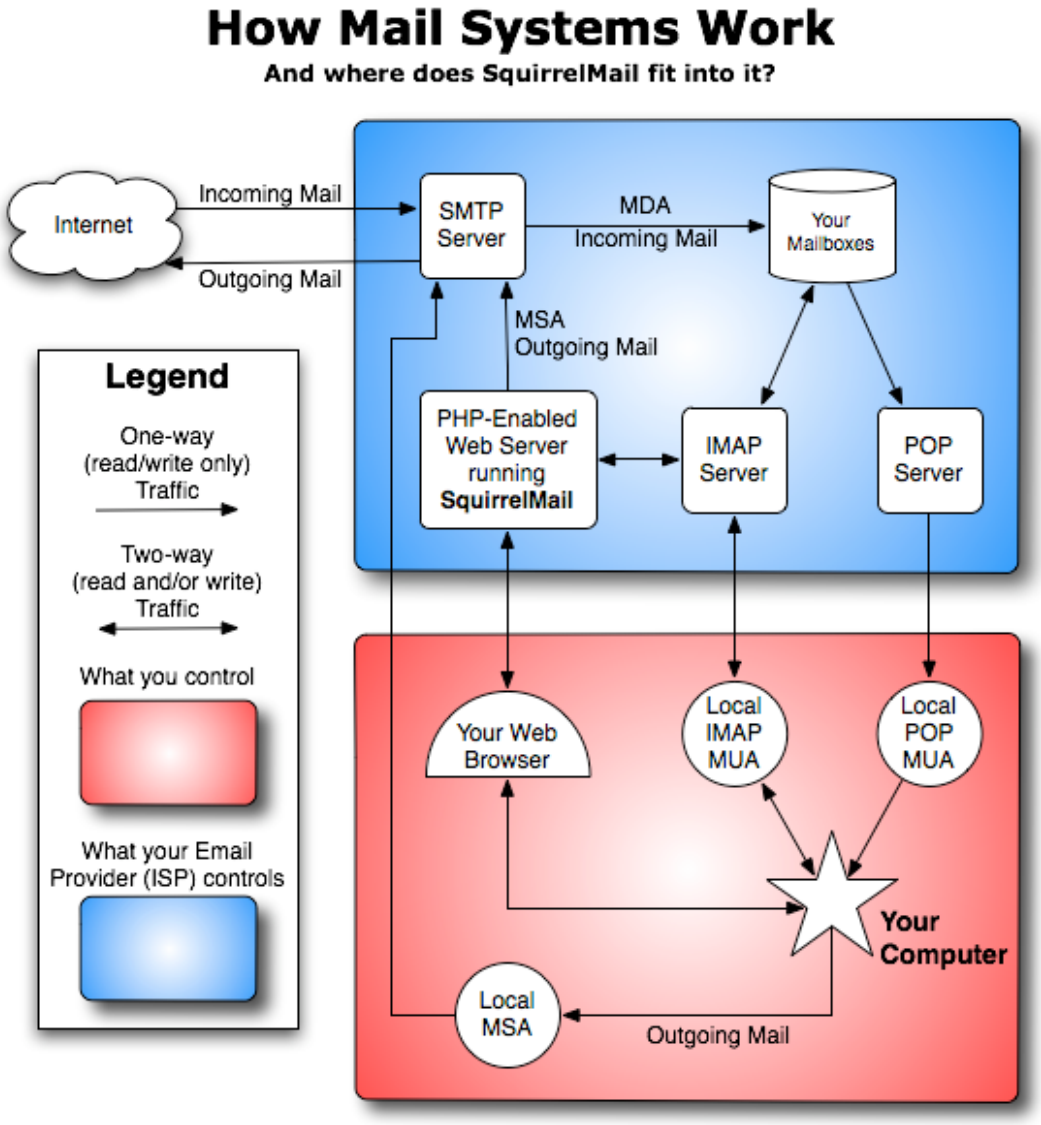
- A w praktyce wygląda to tak:

# SMTP

- A to co my kontrolujemy wygląda tak:



# SMTP

- A w praktyce wygląda to tak:
  - użytkownik (*Mail User Agent* – **MUA**)
  - połączenie MUA z *Mail Submission Agent* (**MSA**) – port 25 lub 587
  - MSA przesyła dane do *Mail Transfer Agent* (**MTA**, zwykle ta sama maszyna)
  - MTA uzyskuje poprzez DNS (rekord **MX** – *Mail Exchange*) adres domeny docelowej – serwera poczty odbiorcy lub pośrednika (**MDA** – *Mail Delivery Agent*)
  - MTA przesyła dane do MDA, ten ewentualnie do kolejnego pośredniczącego MDA, aż do celu
  - MUA odbiorcy poprzez inny protokół (POP lub IMAP) łączy się z MDA i pobiera e-mail

# SMTP

- **Podstawowe komendy protokołu SMTP**
  - HELO – C->S, identyfikacja klienta na serwerze i inicjalizacja komunikacji
  - EHLO – jak HELO, gdy klient chce użyć ESMTP
  - MAIL – specyfikacja adresu nadawcy
  - RCPT – specyfikacja adresu odbiorcy
  - DATA – początek treści wiadomości (na zakończenie kropka w nowej linii)
  - RSET – przerwanie wysyłania maila i wyczyszczenie informacji o nadawcy/odbiorcach
  - NOOP – żądanie odpowiedzi OK (PING)
  - QUIT – C->S, żądanie zakończenia połączenia
  - VRFY – żądanie zweryfikowania istnienia klienta na danym serwerze
- **Przykładowa komunikacja**
  - W domenie example.com
  - Pomiędzy bob@
  - Do alice@, CC dla theboss@

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.com
S: 250 smtp.example.com, I am glad to meet you
C: MAIL FROM:<bob@example.com>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.com>
C: To: Alice Example <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 January 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 header fields and 4 lines
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
{The server closes the connection}
```

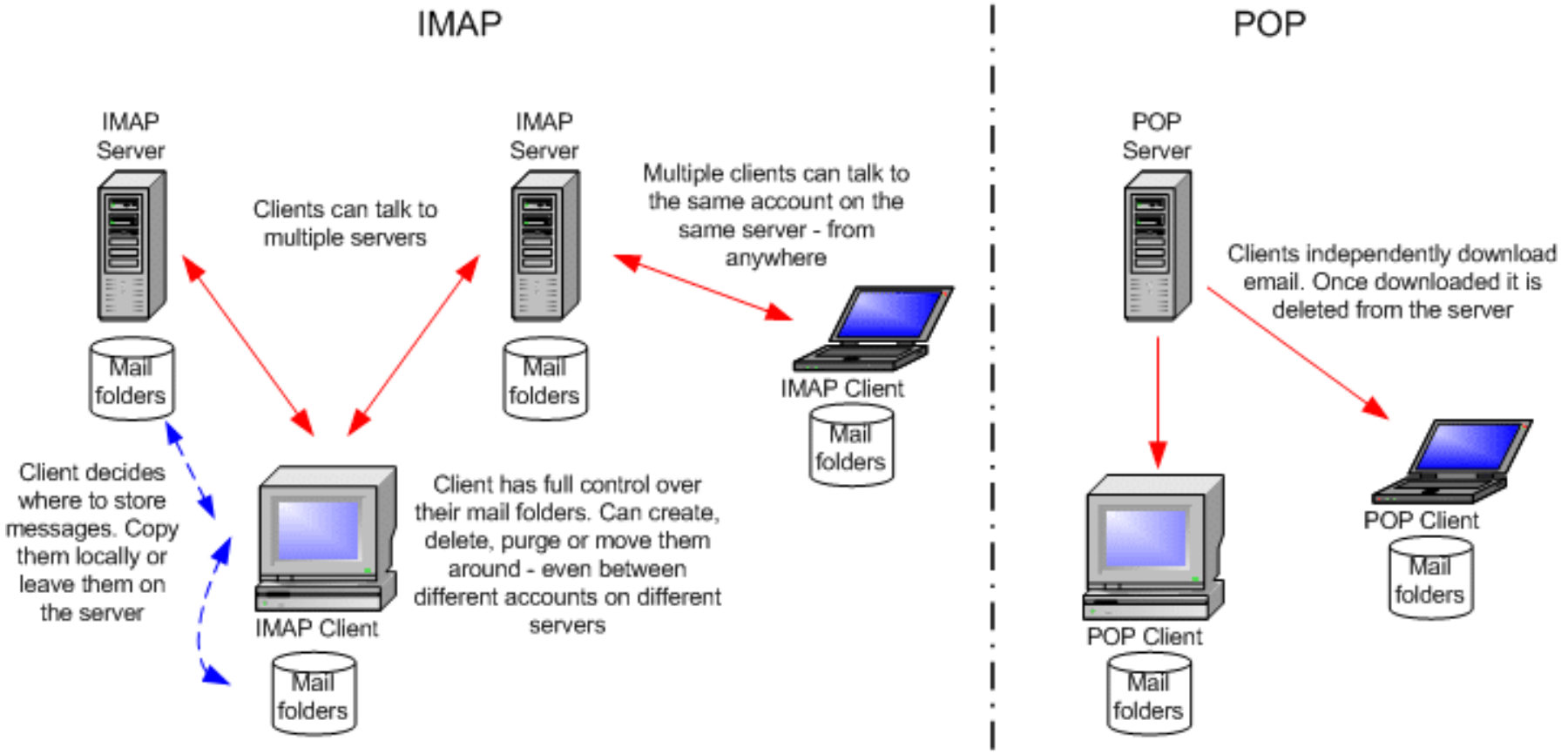
# POP3

- **POP** (*Post Office Protocol*) jest protokołem klienckim do odbierania wiadomości e-mail z serwerów pocztowych, RFC 1939
  - aktualnie wersja trzecia – POP3
  - obecnie IMAP bardziej popularny
- Protokół działa w protokole TCP, port 110
- Tryb działania klient-serwer
  - klient łączy się z serwerem
  - pobranie wszystkich wiadomości z serwera
  - zapisanie wiadomości u klienta
  - usunięcie pobranych wiadomości na serwerze (z możliwością ich zachowania na serwerze)
  - serwer nasłuchuje na porcie 110 czy klient się podłączył

# IMAP

- **IMAP** (*Internet Message Access Protocol*) podobnie jak POP3 służy do pobierania wiadomości z serwera pocztowego, RFC 3501
  - w przeciwieństwie do POP3 – zostawiamy wiadomości na serwerze (wiele klientów)
  - również wykorzystuje TCP, port 143 (w szyfrowaniu port 993)
- Zalety w stosunku do POP3:
  - równoległe podłączanie wielu klientów
  - dostęp do części wiadomości bez pobierania całej
  - flagi stanu (odebrana, przeczytana, itp.)
  - tworzenie folderów i grupowanie wiadomości
  - przeszukiwanie wiadomości na serwerze bez ściągania

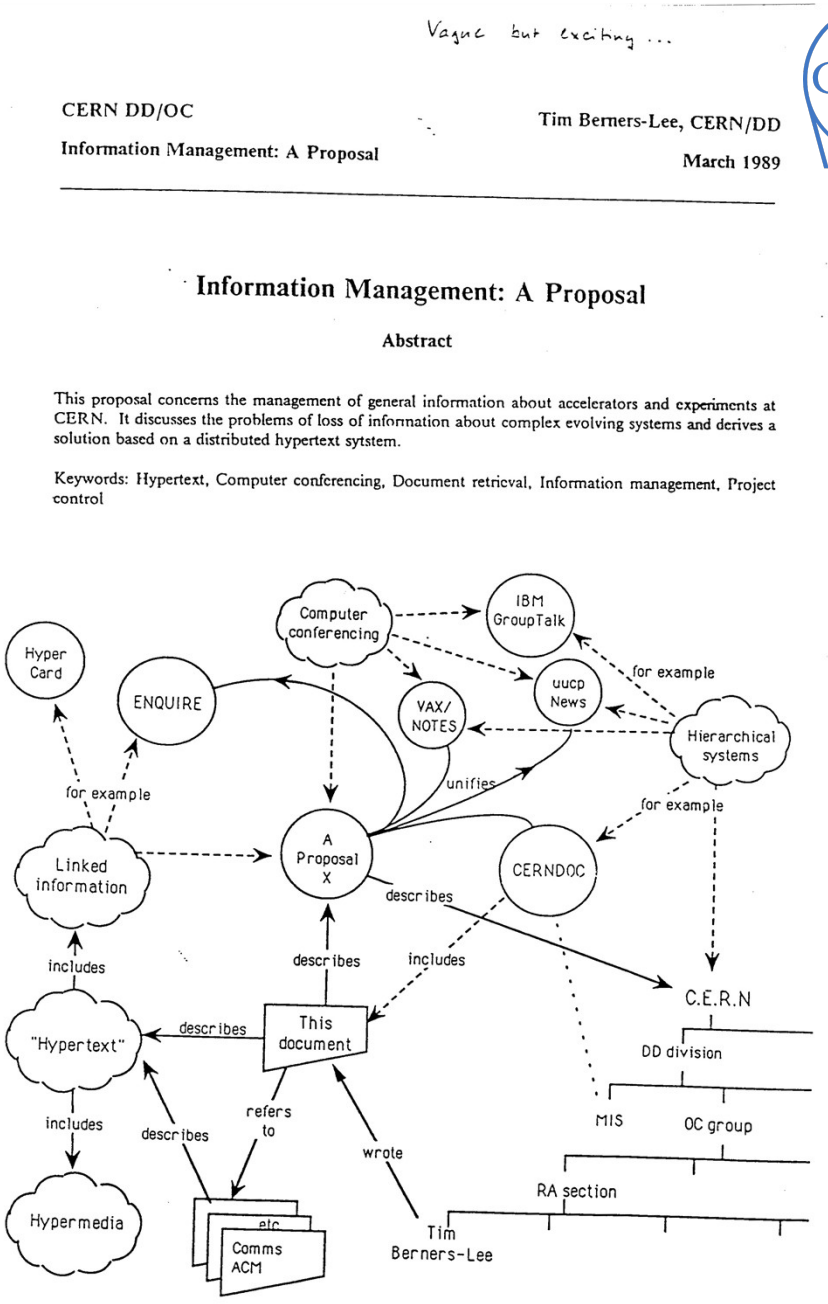
# IMAP


IMAP
IMAP Server
IMAP Server
Clients can talk to multiple servers
Multiple clients can talk to the same account on the same server - from anywhere
Mail folders
Mail folders
IMAP Client
Mail folders
Client decides where to store messages. Copy them locally or leave them on the server
IMAP Client
Mail folders
Client has full control over their mail folders. Can create, delete, purge or move them around - even between different accounts on different servers
POP
POP Server
Clients independently download email. Once downloaded it is deleted from the server
POP Client
Mail folders
POP Client
Mail folders

# HTTP

- **HTTP** (*Hypertext Transfer Protocol*) jest protokołem do przesyłania dokumentów hipertekstowych, RFC 2616
- Powtórka z Wykładu 1...

# Krótka historia Internetu (3)



Vague but exciting ...

CERN DD/OC

Tim Berners-Lee, CERN/DD

Information Management: A Proposal

March 1989

## Information Management: A Proposal

**Abstract**

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext sytstem.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control



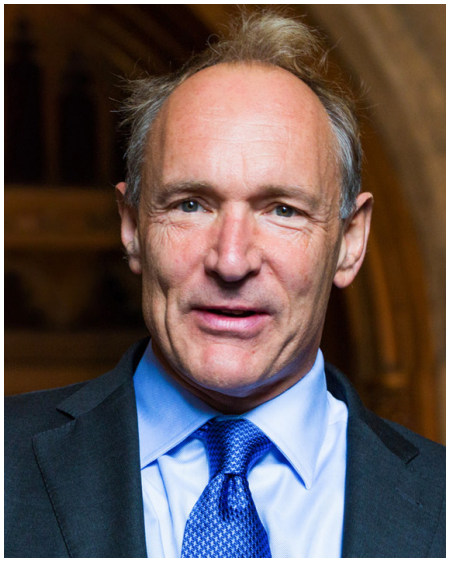
https://cds.cern.ch/record/369245/files/dd-89-001.pdf

1989 – Timothy Berners-Lee, pisze dokument “Information Management: A Proposal”, w którym:

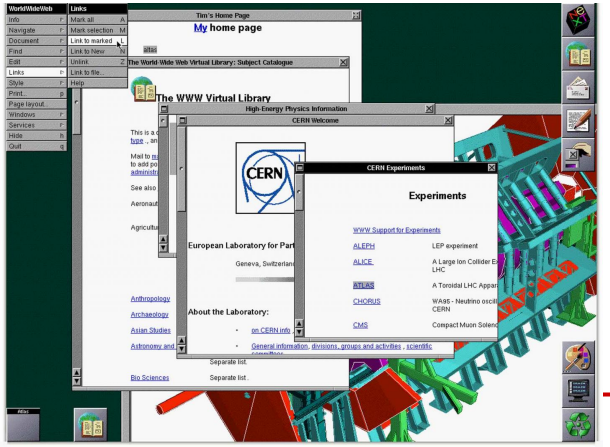
- wprowadza HTML, HTTP, URL, które składają się na WWW (**World Wide Web**)
- tworzy pierwszy serwer WWW
- Pierwsza strona WWW:

  http://info.cern.ch/hypertext/WWW/TheProject.html

Dla większości ludzi **WWW jest synonimem Internetu**



źródło: cern.ch



# HTTP

- **HTTP** (*Hypertext Transfer Protocol*) jest protokołem do przesyłania dokumentów hipertekstowych, RFC 2616
  - dokumenty hipertekstowe to takie zawierające linki do innych dokumentów
  - działa w systemie klient-serwer, jest bestanowy (brak monitoringu sesji)
  - port 80, protokół TCP
- Komunikacja z serwerem za pomocą metod

# HTTP

- **Metody HTTP**
  - GET – pobranie z serwera do klienta zasobu wskazanego przez URI (Uniform Resource Identifier)
  - POST – żądanie odebrania przez serwer danych (np. formularza, komentarza, rekordu bazy danych) przesłanych od klienta w ramach zapytania identyfikowanych poprzez URI. (nie jest idempotentne)
  - HEAD – pobranie informacji o danym zasobie z serwera (ale bez jego zawartości)
  - OPTIONS - informacje o metodach obsługiwanych w ramach danego zasobu
  - PUT – żądanie odebrania i utworzenia/zaktualizowania danych wysłanych od klienta do serwera (jest idempotentne)
  - DELETE – żądanie usunięcia zasobu z serwera
  - TRACE – odesłanie przez serwer odebranego zapytania w celu przetestowania ewentualnych narzutów/modyfikacji serwerów pośredniczących
  - CONNECT – konwersja żądania na tunel TCP/IP w celu stosowania komunikacji szyfrowanej
  - PATCH – częściowa modyfikacja zasobu
- Idempotencja – wiele identycznych wywołań takiego samego żądania ma identyczny skutek co pojedyncze wywołanie tego żądania. (dotyczy stanu serwera po obsłudze żądania)

# HTTP


User clicks

Browser

```
GET http://weather.example.org/oaxaca HTTP/1.1
Host: weather.example.org
Accept: application/xhtml+xml
```

sends request

Server handles request

Web server for weather.example.org

sends response

```
HTTP/1.1 200 OK
Content-Length: 45178
Content-Type: application/xhtml+xml; charset=utf-8

<!DOCTYPE html PUBLIC "...
<html xmlns="http://www...
 <head>
  <title>5 day forecast . . .
```

Server


Browser interprets representation, displays presentation

```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

Request Line
Request Headers
Request Message Header
A blank line separates header & body
Request Message Body

```
HTTP/1.1 200 OK
Date: Sun, 08 Feb xxxx 01:11:12 GMT
Server: Apache/1.3.29 (Win32)
Last-Modified: Sat, 07 Feb xxxx
ETag: "0-23-4024c3a5"
Accept-Ranges: bytes
Content-Length: 35
Connection: close
Content-Type: text/html

<h1>My Home page</h1>
```

Status Line
Response Headers
Response Message Header
A blank line separates header & body
Response Message Body

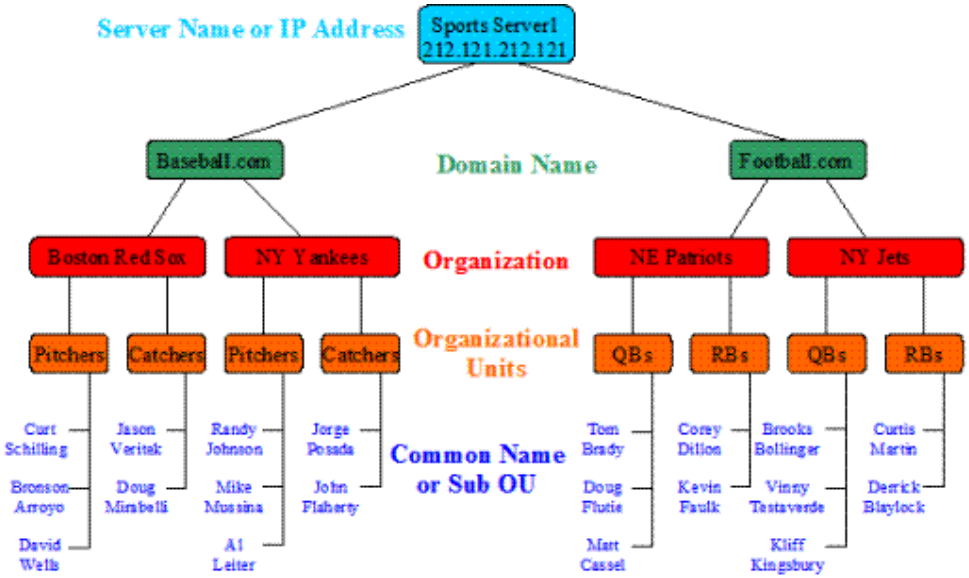
# HTTP

| Linia żądania | Znaczenie |
|---|---|
| GET / HTTP/1.1 | prośba o zwrócenie dokumentu o URI / zgodnie z protokołem HTTP 1.1 |
| Host: example.com | wymagany w HTTP 1.1 nagłówek Host służący do rozpoznania hosta, jeśli serwer na jednym IP obsługuje kilka VirtualHostów |
| User-Agent: Mozilla/5.0 (X11; U; Linux i686; pl; rv:1.8.1.7) Gecko/20070914 Firefox/2.0.0.7 | nazwa aplikacji klienckiej |
| Accept: text/xml,application/xml,application/xhtml+xml ,text/html;q=0.9,text/plain;q=0.8 | akceptowane (bądź nieakceptowane dla q=0) przez klienta typy plików |
| Accept-Language: pl,en-us;q=0.7,en;q=0.3 | preferowany język strony – nagłówek przydatny przy Language negotiation |
| Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7 | preferowane kodowanie znaków |
| Keep-Alive: 300 | czas, jaki klient chce zarezerwować do następnego zapytania w przypadku połączenia Keep-Alive |
| Connection: keep-alive | chęć nawiązania połączenia stałego Keep-Alive z serwerem HTTP/1.0 |
| | znak powrotu karetki i nowej linii (CRLF) |

| Linia odpowiedzi | Znaczenie |
|---|---|
| HTTP/1.1 200 OK | kod odpowiedzi HTTP - zaakceptowanie i zwrócenie zawartości |
| Date: Thu, 20 Dec 2001 12:04:30 GMT | czas serwera |
| Server: Apache/2.0.50 (Unix) DAV/2 | opis aplikacji serwera |
| Set-Cookie: PSID=d6dd02e9957fb162d2385c a6f2829a73; path=/ | nakazanie klientowi zapisania ciasteczka |
| Cache-Control: no-store, no-cache, must-revalidate | no-store zabrania przechowywania dokumentu na dysku, must-revalidate nakazuje bezwzględnie sprawdzić świeżość dokumentu za każdym razem |
| Keep-Alive: timeout=15, max=100<br>Connection: Keep-Alive | akceptacja połączenia Keep-Alive dla klientów HTTP/1.0 |
| Transfer-Encoding: chunked | typ kodowania zawartości stosowanej przez serwer |
| Content-Type: application/xhtml+xml; charset=utf-8 | typ MIME i strona kodowa zwróconego dokumentu |
| | znak powrotu karetki i nowej linii (CRLF) |
| <html><head><title>An Example page</title></head><br><body> Hello World</body></html> | tutaj zawartość dokumentu (wiele linii) |

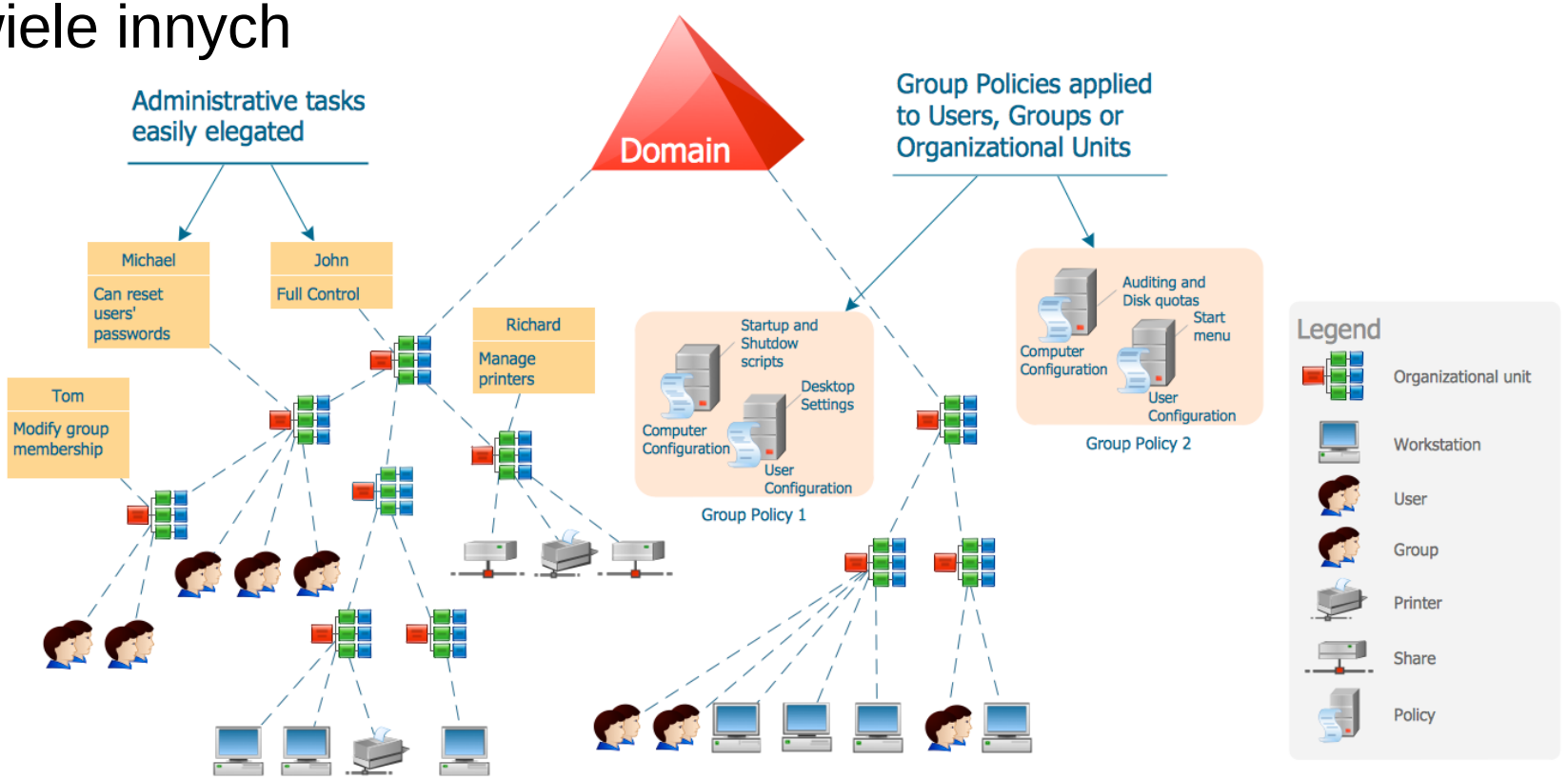
# LDAP

- **LDAP** (*Leightweight Directory Access Protocol*) jest protokołem umożliwiającym korzystanie z usług katalogowych, RFC 4511
  - **Usługa katalogowa** (*Directory service*) – to baza danych zawierająca: użytkowników, aplikacje, urządzenia i inne zasoby sieciowe. Pomaga kojarzyć (tworzy relacje) między ludźmi i aplikacjami/urządzeniami (przykładowo – można się dostać do drukarki z poziomu wielu użytkowników)
  - LDAP pracuje na protokole TCP, port 389
  - dane w drzewiastej strukturze katalogów

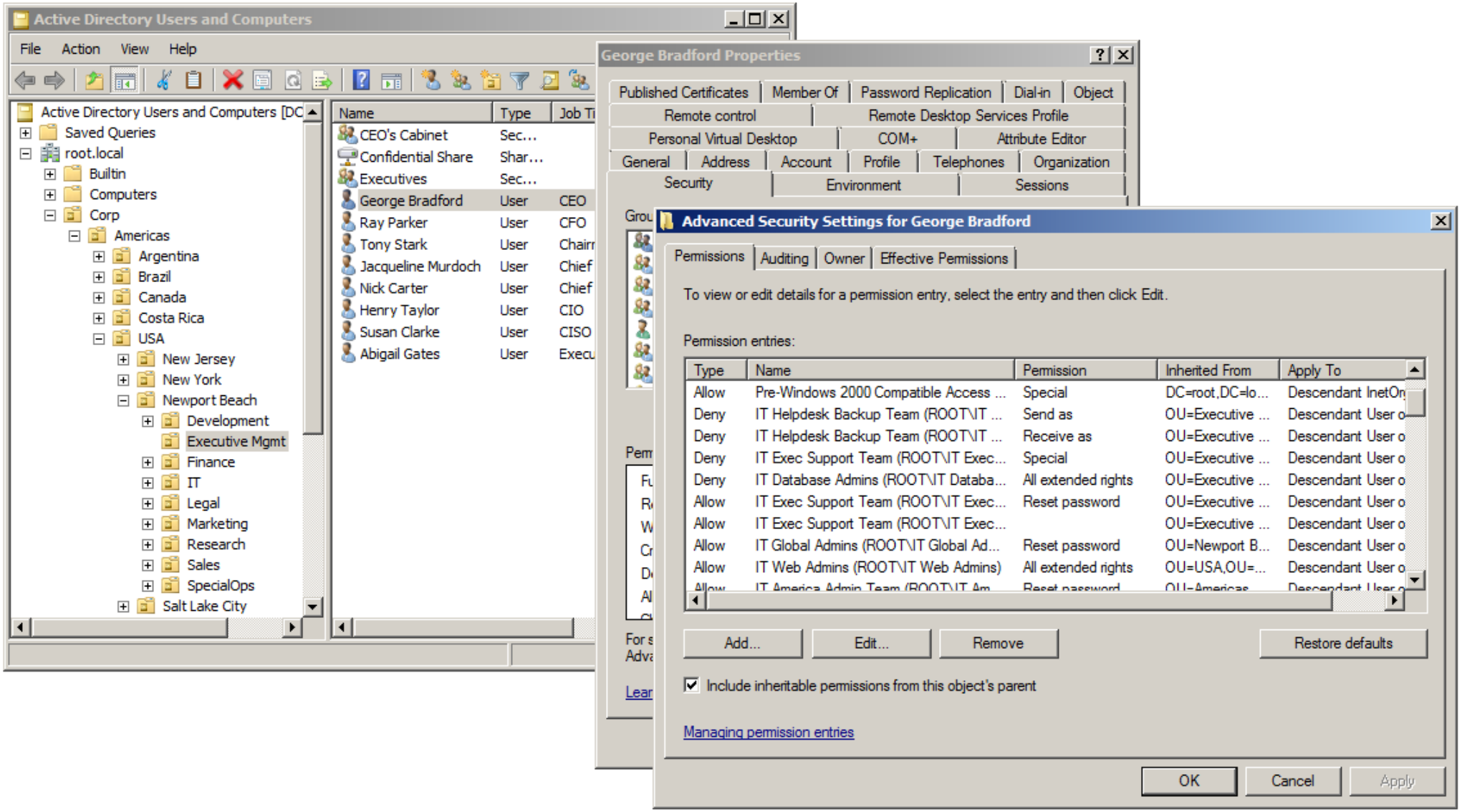


# LDAP

- LDAP zapewnia logiczny, precyzyjny i hierarchiczny sposób opisu zasobów danej organizacji
- Przykładowe systemy:
  - Active Directory (Microsoft)
  - OpenLDAP
  - … wiele innych

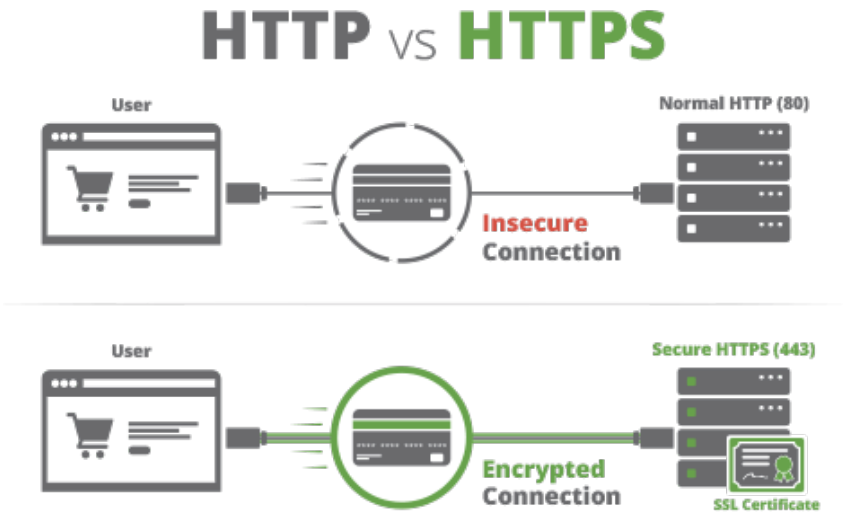


# LDAP


Active Directory Users and Computers
File Action View Help
Active Directory Users and Computers [DC
Saved Queries
root.local
Builtin
Computers
Corp
Americas
Argentina
Brazil
Canada
Costa Rica
USA
New Jersey
New York
Newport Beach
Development
Executive Mgmt
Finance
IT
Legal
Marketing
Research
Sales
SpecialOps
Salt Lake City
Name Type Job Ti
CEO's Cabinet Sec...
Confidential Share Shar...
Executives Sec...
George Bradford User CEO
Ray Parker User CFO
Tony Stark User Chairr
Jacqueline Murdoch User Chief
Nick Carter User Chief
Henry Taylor User CIO
Susan Clarke User CISO
Abigail Gates User Execu
George Bradford Properties
Published Certificates
Member Of
Password Replication
Dial-in
Object
Remote control
Remote Desktop Services Profile
Personal Virtual Desktop
COM+
Attribute Editor
General
Address
Account
Profile
Telephones
Organization
Security
Environment
Sessions
Advanced Security Settings for George Bradford
Permissions
Auditing
Owner
Effective Permissions
To view or edit details for a permission entry, select the entry and then click Edit.
Permission entries:
Type Name Permission Inherited From Apply To
Allow Pre-Windows 2000 Compatible Access ... Special DC=root,DC=lo... Descendant InetOr
Deny IT Helpdesk Backup Team (ROOT\IT ... Send as OU=Executive ... Descendant User o
Deny IT Helpdesk Backup Team (ROOT\IT ... Receive as OU=Executive ... Descendant User o
Deny IT Exec Support Team (ROOT\IT Exec... Special OU=Executive ... Descendant User o
Deny IT Database Admins (ROOT\IT Databa... All extended rights OU=Executive ... Descendant User o
Allow IT Exec Support Team (ROOT\IT Exec... Reset password OU=Executive ... Descendant User o
Allow IT Exec Support Team (ROOT\IT Exec... OU=Executive ... Descendant User o
Allow IT Global Admins (ROOT\IT Global Ad... Reset password OU=Newport B... Descendant User o
Allow IT Web Admins (ROOT\IT Web Admins) All extended rights OU=USA,OU=... Descendant User o
Add...
Edit...
Remove
Restore defaults
Include inheritable permissions from this object's parent
Managing permission entries
OK
Cancel
Apply

# SSL/TLS

- **SSL** (*Secure Socket Layer*), obecnie nazywane **TLS** (*Transport Layer Security*) jest protokołem do bezpiecznej komunikacji przy udziale certyfikatów, RFC 6101
  - działa w warstwie sesji modelu OSI
  - wykorzystywany jest przez protokoły wyższego poziomu (aplikacji) – np. HTTPS
  - wykorzystuje szyfrowanie (klucze publiczne i prywatne) danych




# SSL/TLS

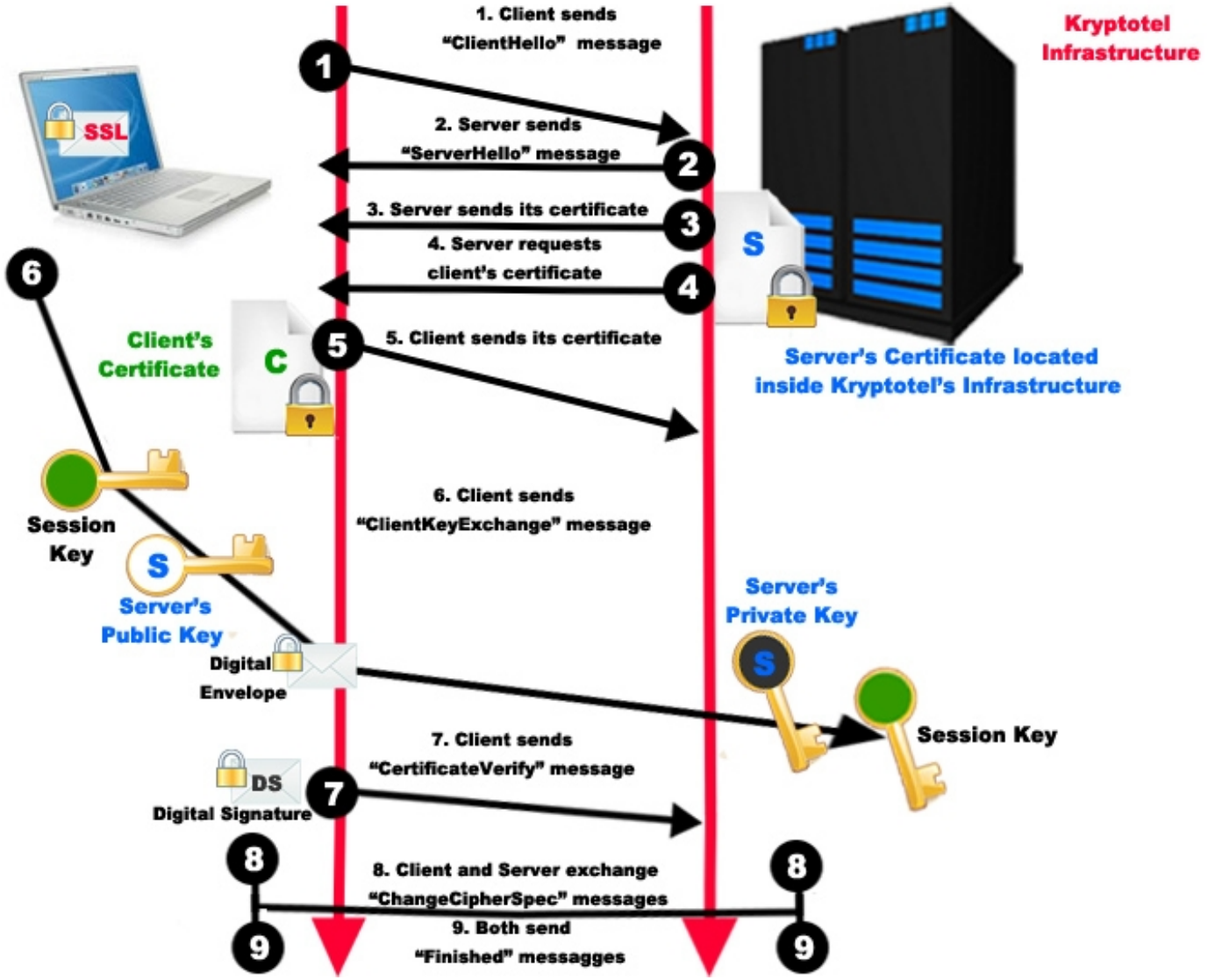
- Potwierdzanie połączeń odbywa się poprzez **certyfikaty SSL** przedstawiane przez serwer (format X.509)
- Certyfikaty są podpisywane przez specjalne urzędy certyfikacji (**CA** – *Certificate Authority*). Jeśli certyfikat nie posiada autoryzacji CA, zostanie nam wyświetlony stosowny komunikat
- Certyfikaty są wydwane przez lokalne organizacje (np. rządowe agendy)
  - przykładowo w Polsce instytucją certyfiukącą podpis kwalifikowany (to też certyfikat SSL!) wystawców jest **Narodowe Centrum Certyfikacji** przy NBP - https://www.nccert.pl/

# SSL/TLS

- Jak to działa?
- Protokół SSL działa dwuetapowo
  - Pierwszy etap – Hello - następuje uzgadnianie pomiędzy klientem a serwerem parametrów kryptograficznych
  - Następnie serwer wysyła swój certyfikat pozwalający klientowi na sprawdzenie swojej tożsamości.
  - Kolejnym etapem jest wysłanie przez serwer swojego klucza publicznego.
  - Dalej – serwer zawiadamia, że klient może przejść do następnej fazy zestawiania połączenia
  - Tą fazą jest mianowicie generowanie przez obie strony klucza sesji używanego do faktycznej wymiany danych. Klucz jest generowany na podstawie liczb losowych (klienta i serwera) ustalanych w pierwszej fazie – Hello
  - Wygenerowany tajny klucz jest kluczem algorytmu symetrycznego
  - Po tych procesach, zasadnicza wymiana danych następuje już przy użyciu symetrycznego szyfrowania
- Takie rozwiązanie jest obecnie uznawane za standard bezpiecznych połączeń internetowych, a protokół SSL/TLS jest obecnie instalowany we wszystkich przeglądarkach internetowych. Umożliwia on również zabezpieczenie protokołów warstw aplikacji takich jak np. poczta elektroniczna czy strony internetowe (banki)

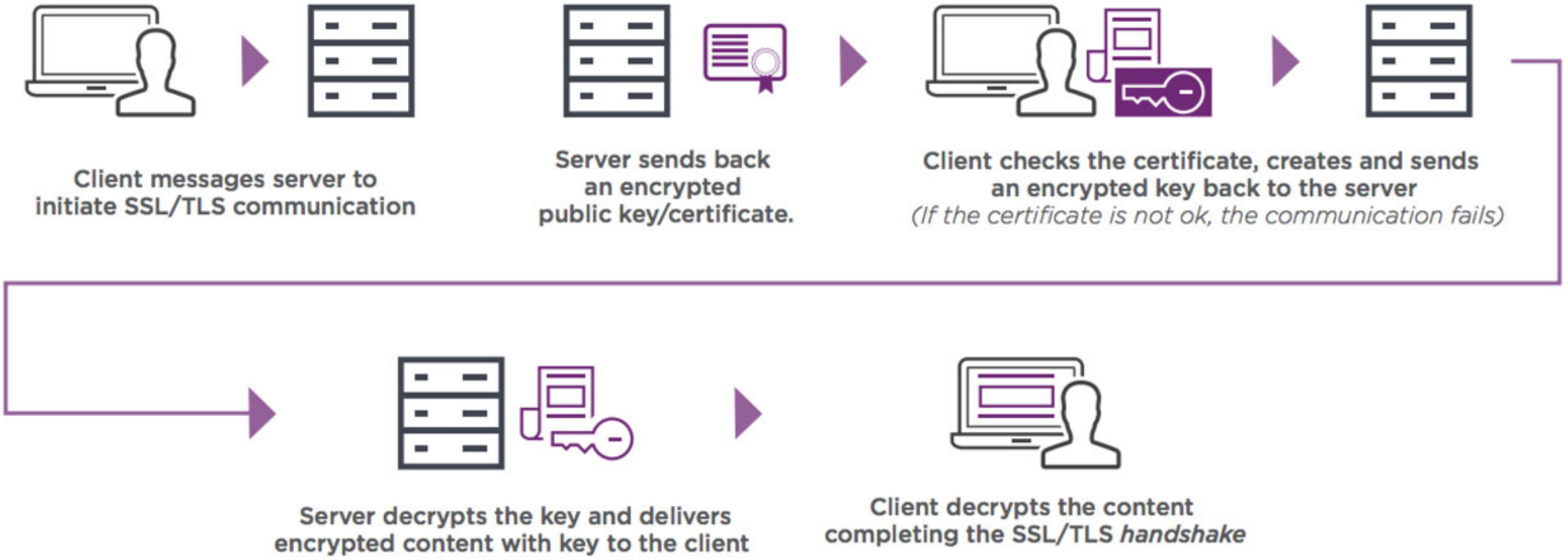
# SSL/TLS

- Jak to działa?



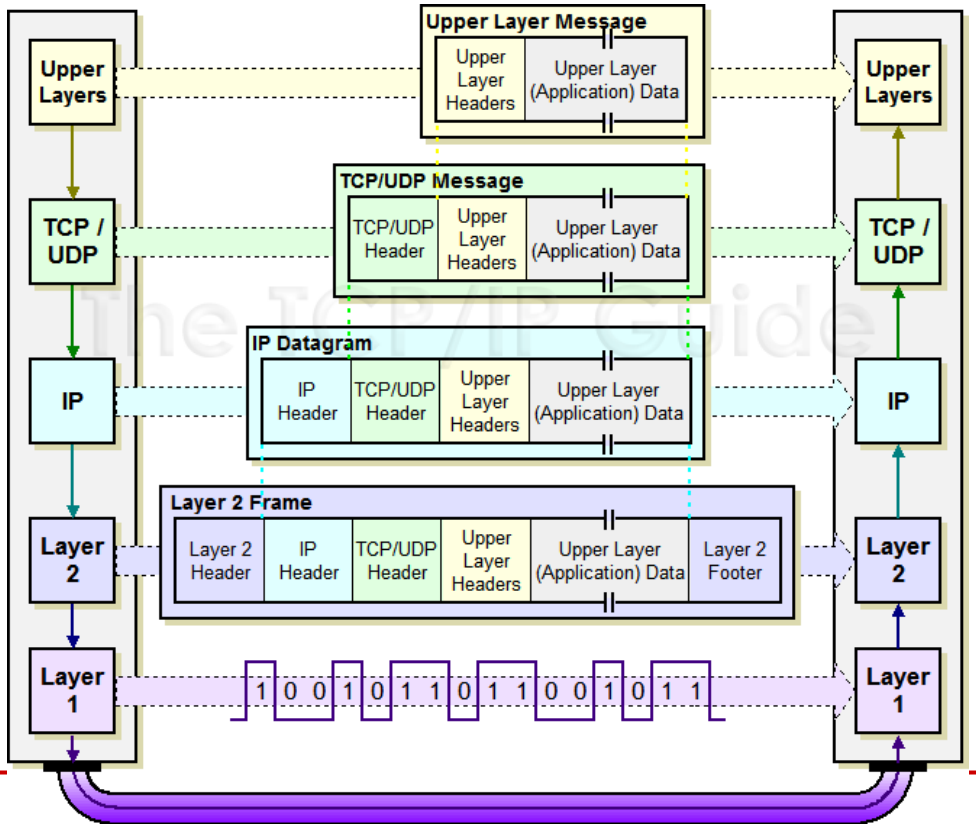
# SSL/TLS

- Jak to działa?



# Inne protokoły

- Protokołów jest oczywiście bardzo dużo
- Pisząc program komputerowy albo korzystamy z już istniejącego (np. nasz klient SSH), albo tworzymy swój własny (np. jakaś gra komputerowa)
- Dane są oczywiście ubierane w nagłówki protokołu (np. HTTP) kapsułkowane do niższych warstw modelu ISO/OSI




WYDZIAŁ FIZYKI
POLITECHNIKA WARSZAWSKA
POLITECHNIKA WARSZAWSKA
WARSAW UNIVERSITY OF TECHNOLOGY

# KONIEC