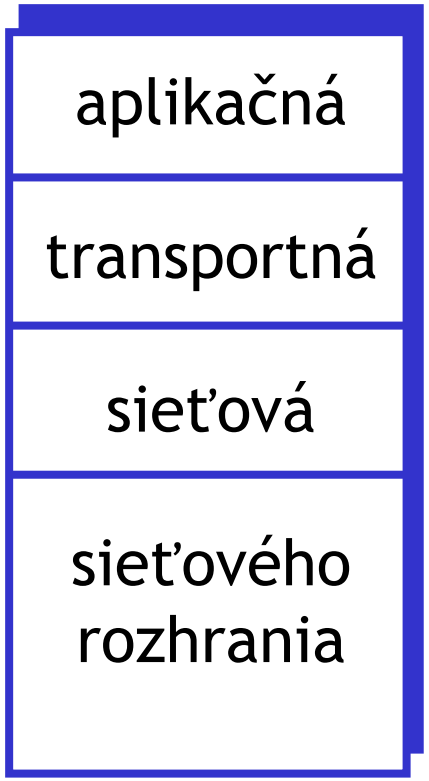
Implementácia internetu TCP/IP

- aplikačná (application): umožňuje fungovanie sieťových aplikácií – definuje tvar a poradie správ
  - prezentačná a relačná splynuli s aplikačnou
  - tieto služby musí aj tak mať implementované aplikácia, ak to potrebuje
  - a čo ak nepotrebuje?
  - HTTP, FTP, SMTP, POP, IMAP, XMPP, SSH, ...
- transportná (transport): prenáša dáta medzi dvoma procesmi na rôznych koncových zariadeniach
  - TCP, UDP
- sieťová (network): smeruje datagramy od odosielateľa k príjemcovi
  - IP, smerovacie protokoly
- sieťového rozhrania (network interface): splynutie funkcionality do technológií na prenos dát medzi susednými sieťovými prvkami a spôsobu prenášania binárnych dát
  - PPP, Ethernet



Transportná vrstva: čo už vieme

- transportná vrstva odosielateľa delí správy (prúd dát) aplikačnej vrstvy na časti, z ktorých pridaním hlavičky vznikajú segmenty
- transportná vrstva príjemcu extrahuje dáta zo segmentov a poskytuje ich aplikačnej vrstve
- hlavnou úlohou transportných protokolov je umožniť odosielanie dát medzi dvoma soketmi
- soket otvára proces alebo vlákno procesu, pričom musí určiť port a rozhranie (jeho IP adresu), cez ktoré bude komunikovať
- soket je jednoznačne určený:
  - v UDP IP adresou príjemcu a portom príjemcu
  - v TCP IP adresami príjemcu a odosielateľa a portami príjemcu a odosielateľa

UDP: User Datagram Protocol [RFC 768]

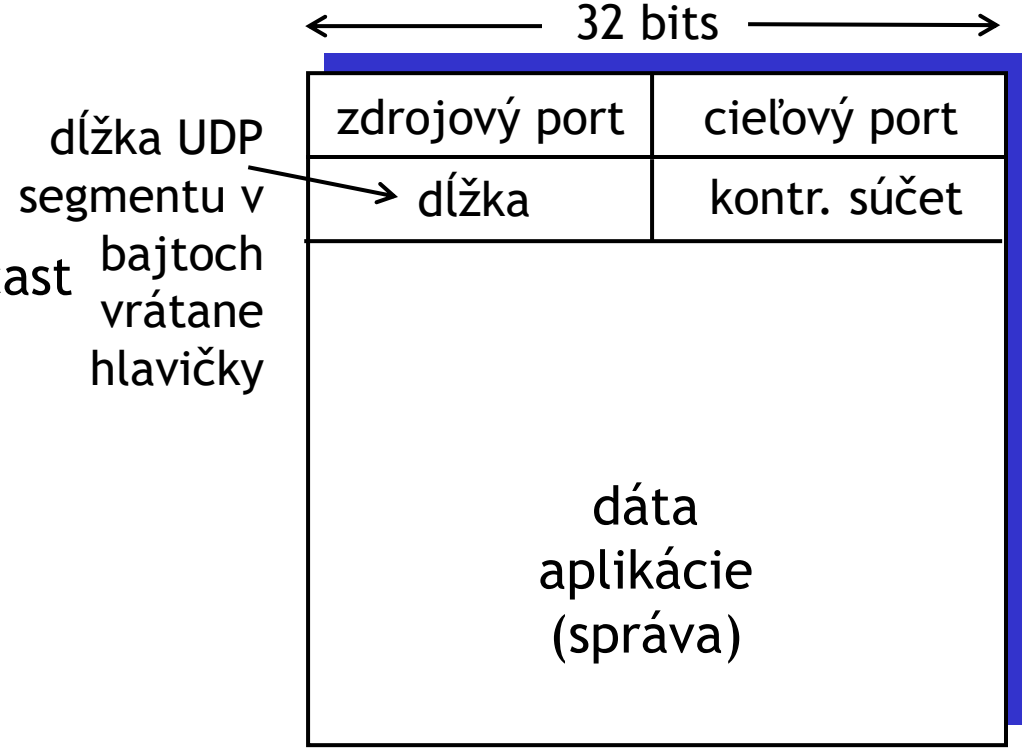
- UDP segmenty sa môžu:
  - stratiť
  - byť doručené v inom poradí, ako boli odoslané
- odosielanie „najväčším úsilím“ (best effort)
- *bezspojový protokol:*
  - žiadne nadväzovanie spojenia (handshaking) medzi odosielateľom a príjemcom
  - každý UDP segment je obslúžený nezávisle na ostatných

Načo je dobré UDP?

- žiadne inicializácie spojenia, ktoré môžu spôsobiť oneskorenie
- jednoduchý: nie je potrebné žiadne uchovávanie stavu odosielania/prijímania
- real-time prenos
- žiadna kontrola zahltenia: UDP odosiela dáta hneď, keď ich dostane z aplikačnej vrstvy
- odosielanie segmentu viacerým cieľovým staniciam (broadcast, multicast)

UDP: použitie

- Streamovanie multimédií
  - tolerujú stratu častí dát
  - závislé na rýchlosti odosielania
  - viacerým cieľom cez multicast
- DNS
- SNMP – simple network management protocol
- spoľahlivý prenos cez UDP: spoľahlivosť je riešená v aplikačnej vrstve
  - špecifické zotavovanie z chýb pre danú aplikáciu



formát UDP segmentu

Kontrolný súčet UDP segmentu

Cieľ: odhalenie "chýb" (napr. zmenené bity) v dôjdenom segmente

Odosielateľ:

- prechádza obsah segmentu ako množinu 16-bitových čísiel
- kontrolný súčet: sčítanie týchto čísiel a na záver vyrobenie inverzného čísla z výsledku sčítania (nuly vymení za jednotky a jednotky za nuly)
- odosielateľ pridá kontrolný súčet do UDP hlavičky

Príjemca:

- vypočíta súčet obsahu segmentu ako množiny 16-bitových čísiel
- nakoniec sčíta výsledok s kontrolným súčtom z UDP hlavičky.
  - Ak výsledkom nie je číslo so samými jednotkami, našli sme chybu a segment zahodíme
  - Ak áno, nenašli sme chybu
  - (ale chyba mohla aj tak nastať)

Príklad kontrolného súčtu

- Obsah segmentu tvoria červené čísla

Odosielateľ:

	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
súčet	1	1	0	0	1	0	1	0	1	1	0	0	1	0	1	0
kontrolný súčet	0	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1

Príjemca – spraví rovnaký súčet obsahu segmentu:

súčet	1	1	0	0	1	0	1	0	1	1	0	0	1	0	1	0
	0	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1
kontrolný súčet z hlavičky	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

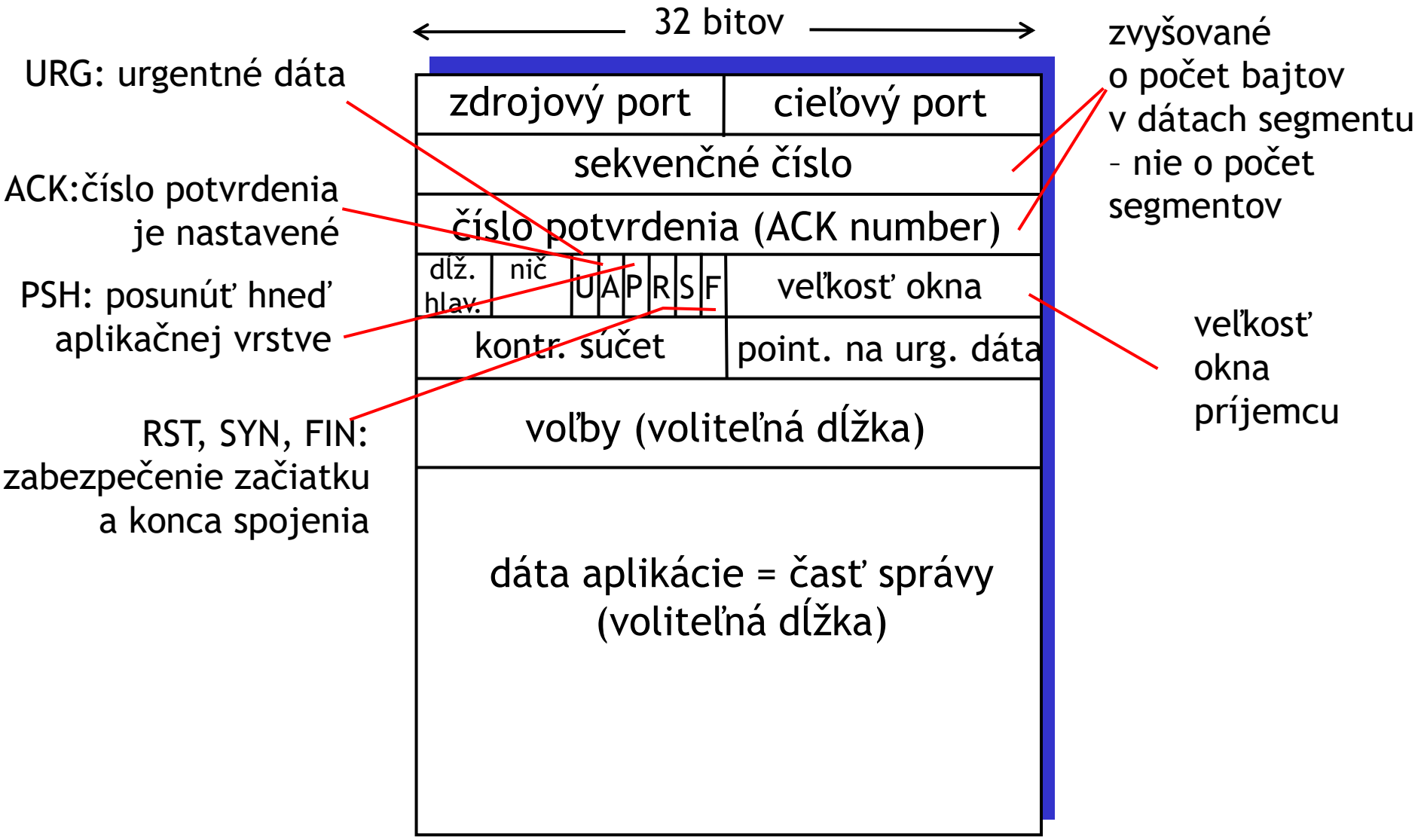
samé 1 - OK

TCP: vlastnosti

RFCs: 793, 1122, 1323, 2018, 2581

- point-to-point:
  - jeden odosielateľ, jeden príjemca
- potvrdzovaný prúd bajtov v správnom poradí
- pipelining:
  - kontrola toku dát a zahltenia ovplyvňujú veľkosť okna
- buffre odosielateľa a príjemcu
- full duplex:
  - obojsmerný tok dát v tom istom spojení
- so spojením:
  - handshaking (zahájenie spojenia cez kontrolné správy) pred odosielaním dát
- kontrola toku dát:
  - odosielateľ nezahltí príjemcu
- kontrola zahltenia siete:
  - odosielateľ nezahltí zariadenia na ceste k cieľu

Štruktúra TCP segmentu

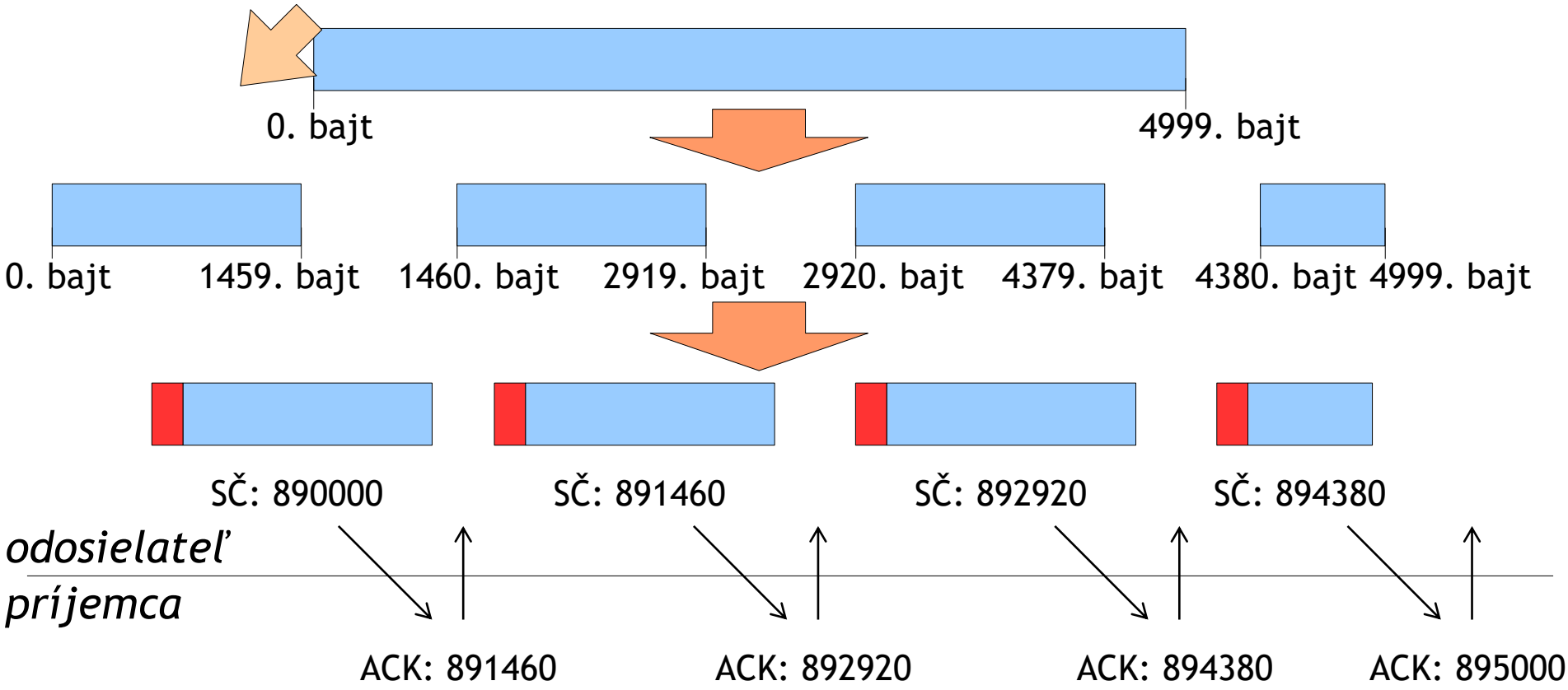


TCP sekvenčné čísla a potvrdenia

Vygenerované sekvenčné číslo = 889999, Prvé dátové SČ = 890000

odosielajúci proces

pošle "veľkú" správu transportnej vrstve veľkosti 5000 bajtov



TCP: Manažment napojenia

Napojenie (handshake):

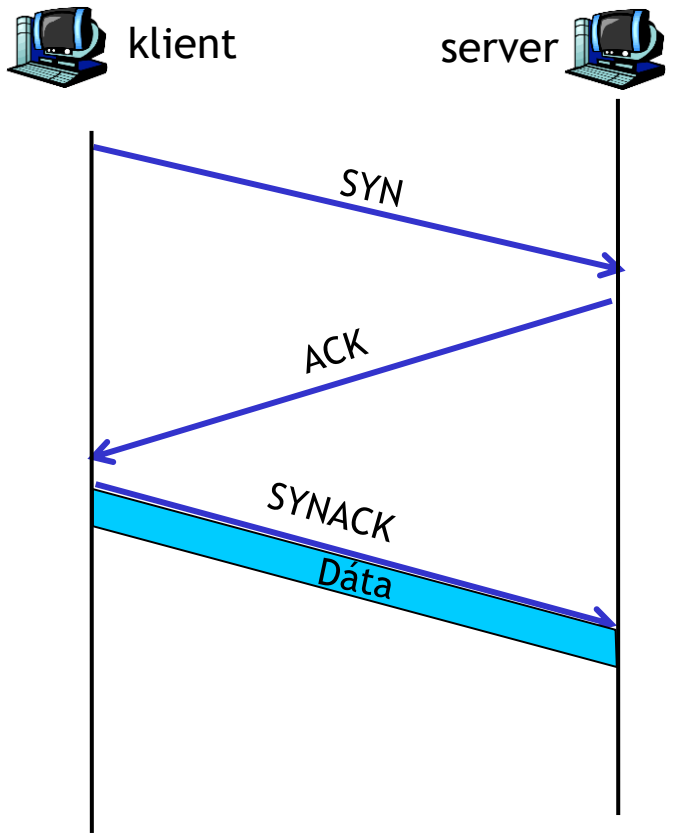
krok 1: klient pošle SYN segment na server

- vygeneruje svoje úvodné sekvenčné číslo
- žiadne dáta
- klient alokuje buffer

krok 2: server prijme SYN, odpovie SYNACK segmentom

- vygeneruje svoje úvodné sekvenčné číslo
- žiadne dáta
- server alokuje buffer

krok 3: klient prijme SYNACK, odpovie bežným ACK segmentom, ktorý už môže obsahovať dáta



Po vytvorení spojenia už je z pohľadu TCP jedno, ktorý z nich je klient a ktorý je server

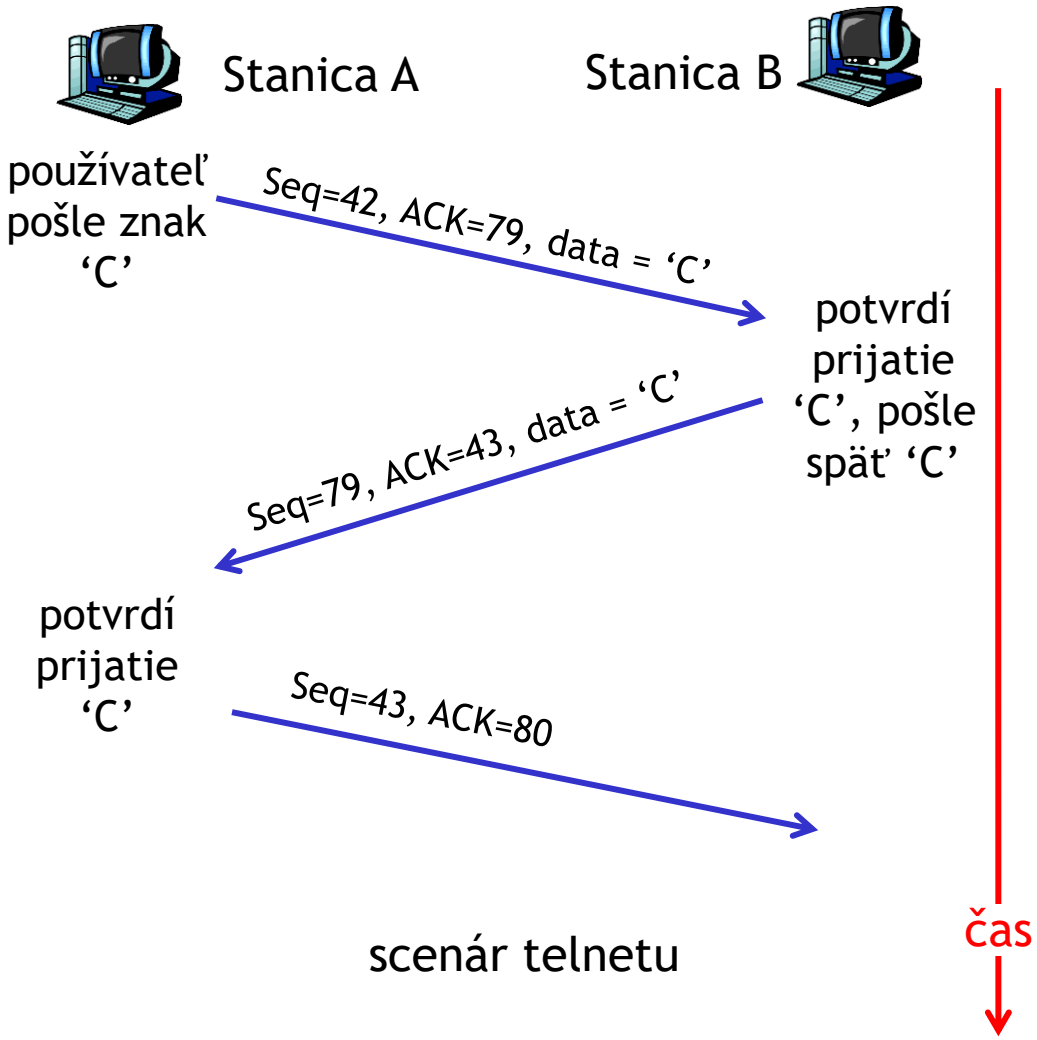
TCP sekvenčné čísla a potvrdenia

Sekvenčné čísla:

- poradové číslo prvého dátového bajtu v segmente

Čísla potvrdenia (ACK):

- sekvenčné číslo dátového bajtu, ktorý je očakávaný ako ďalší v poradí na prijatie
- kumulatívne potvrdenie



scenár telnetu

TCP: Manažment odpojenia

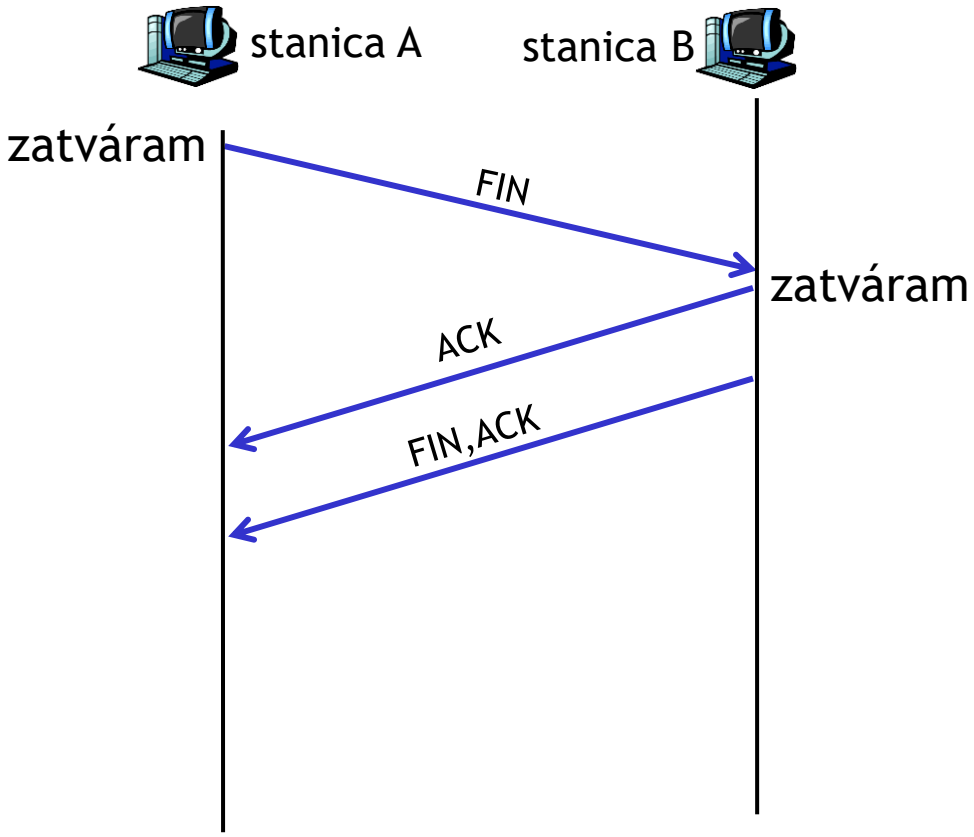
Uzavretie spojenia:

jedna zo staníc chce ukončiť spojenie (pôvodný klient alebo server): `socket.close();`

krok 1: stanica A pošle FIN (alebo FINACK) segment stanici B

krok 2: stanica B prijme FIN, odpovie ACK segmentom.

krok 3: stanica B ak už nemá ďalšie dáta, pošle FINACK segment stanici A.

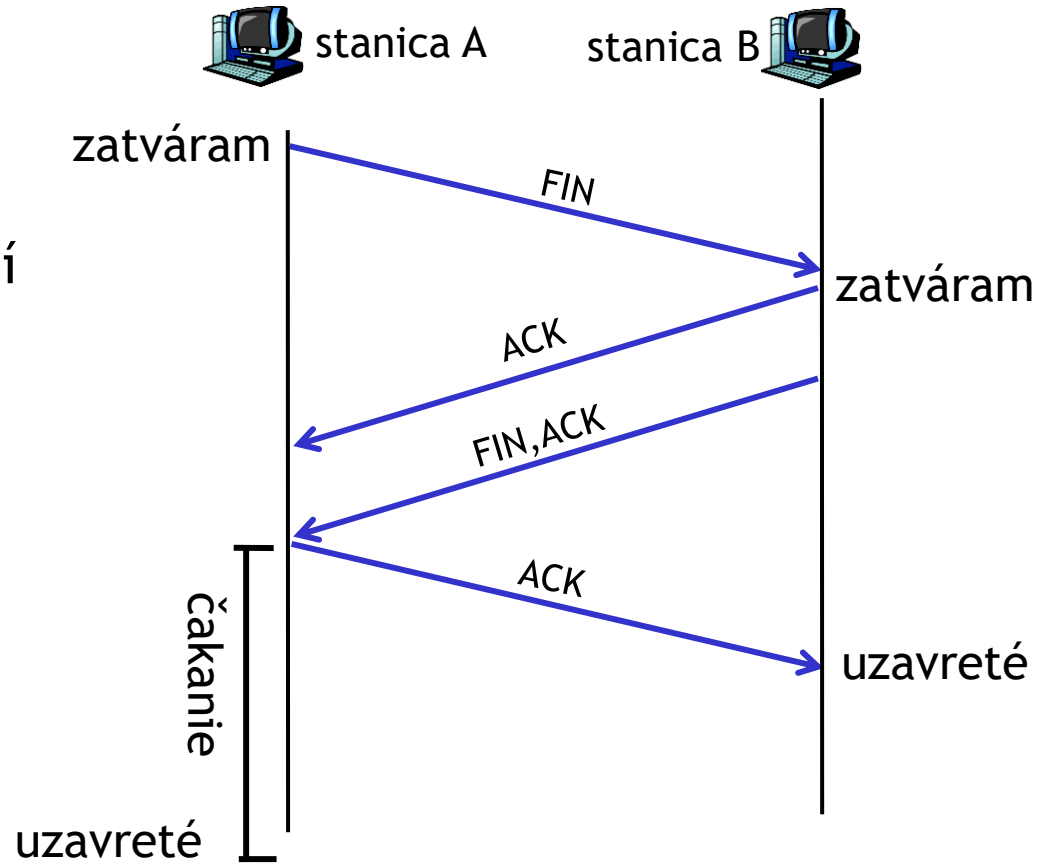


TCP: Manažment odpojenia

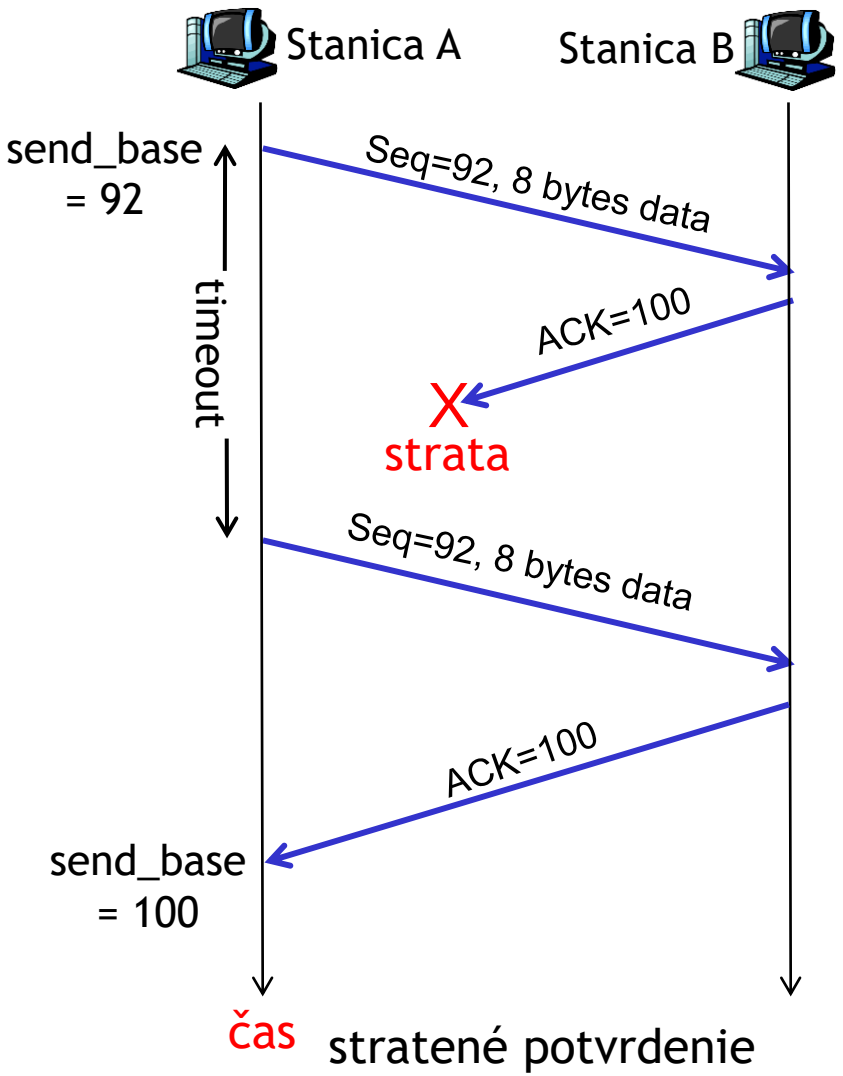
krok 4: stanica A prijme FINACK, odpovie ACK segmentom.

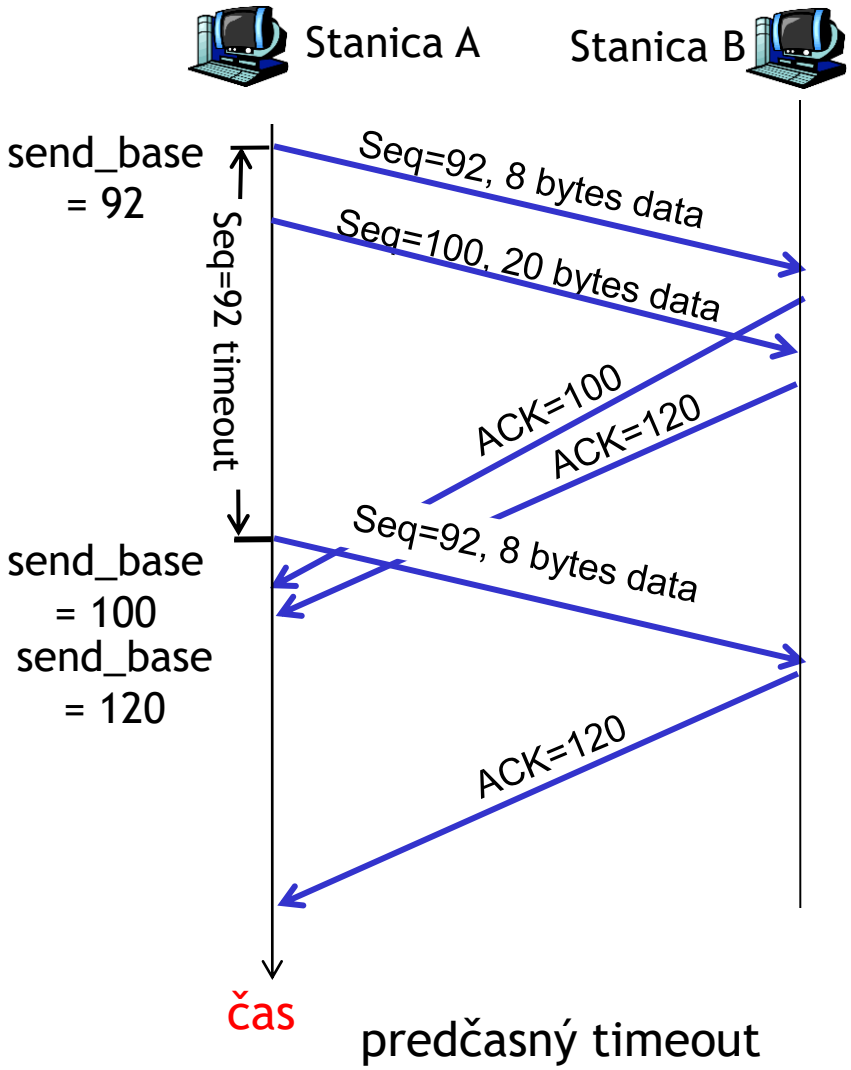
- začne čakať, či nepríde ďalší FINACK (ak sa jeho posledný ACK stratil)
- obvykle sa čaká 30 s, 1 min alebo 2 min

krok 5: stanica B prijme ACK. Uzatvorenie spojenia.

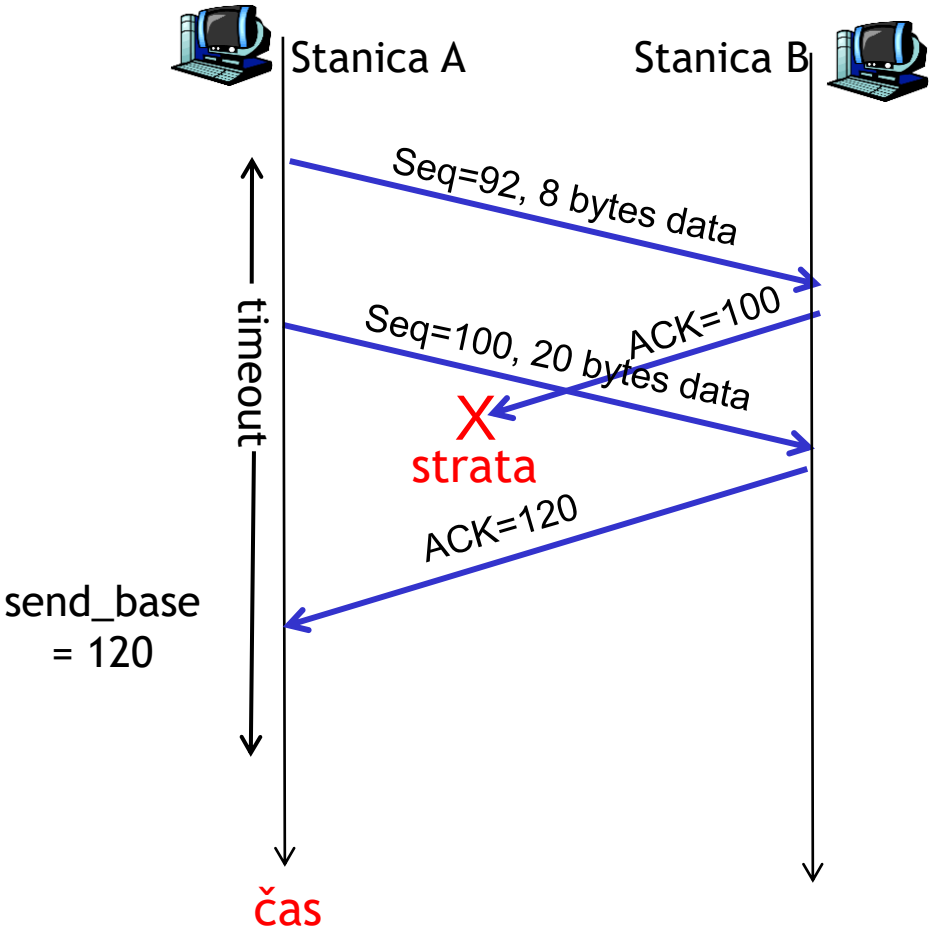


TCP: príklad komunikácie



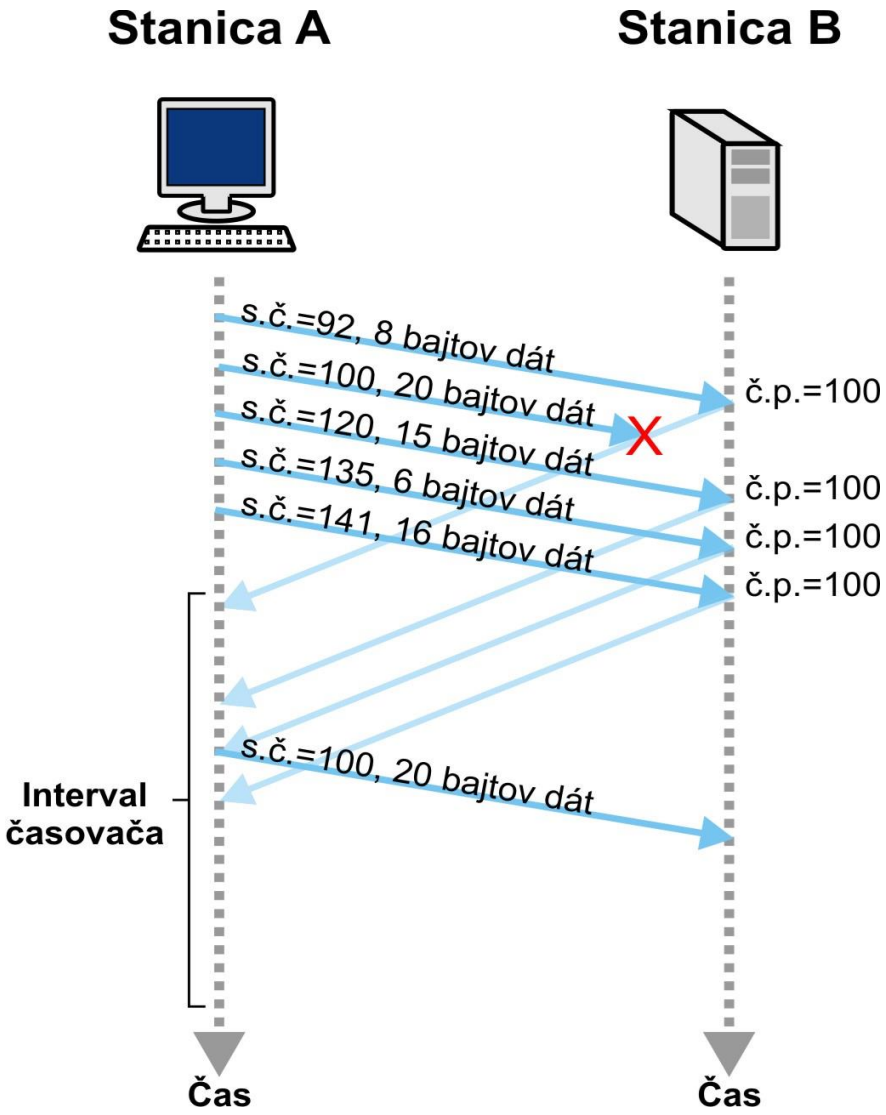


TCP: príklad komunikácie



kumulatívne potvrdenie

TCP: viacnásobné potvrdenie



Udalosti odosielateľa:

došli dáta aplikačnej vrstvy:

- Vytvor segment so sekvenčným číslom *nextSeqNum*

❖nastav *nextSeqNum* na svoje sekvenčné číslo plus veľkosť dát v svojom segmente

- Ak bolo okno odosielateľa prázdne, zapni časovač
- expiračný interval: *TimeOutInterval*

timeout:

- prepošli najstarší nepotvrdený segment
- reštartuj časovač

prišlo potvrdenie:

- ak sa týka doteraz nepotvrdených segmentov (číslo potvrdenia ≥ send_base)

❖nastav segment s dôjdeným číslom potvrdenia a všetky od neho staršie ako potvrdené a odstráň ich z okna odosielateľa

❖ak sú ešte nepotvrdené segmenty, zapni časovač

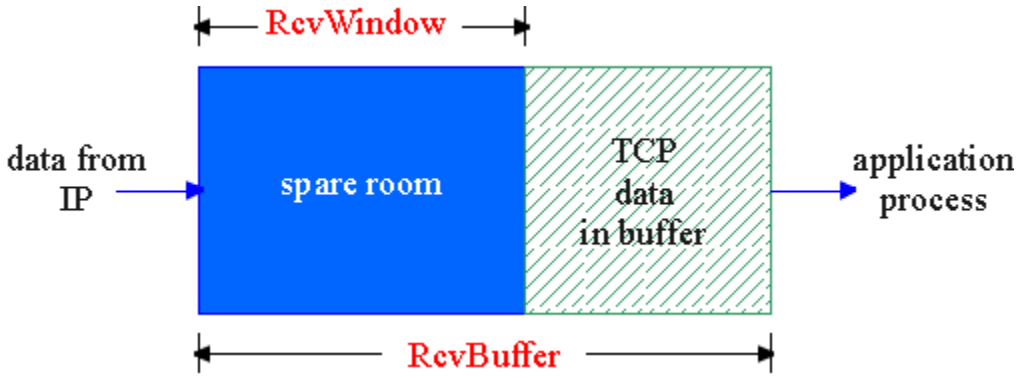
- ak prišli 3 potvrdenia s rovnakým číslom, prepošli najstarší nepotvrdený segment znova

Rýchle preposlanie

- Timeout nastáva až po relatívne dlhom čase:
  - dlhá prestávka pred preposlaním strateného paketu
- Zistenie stratených segmentov cez viacnásobné potvrdenie
  - Odosielateľ odošle mnoho dát, pokiaľ sa nezistí strata
  - Ak sa segment stratí, začne prichádzať veľa rovnakých potvrdení
- Ak odosielateľ dostane 3 potvrdenia s rovnakým číslom, predpokladá, že dáta sa stratili:
  - rýchle preposlanie: pošlem dáta znova bez toho, aby som čakal na timeout a reštartujem časovač

TCP: Kontrola toku dát

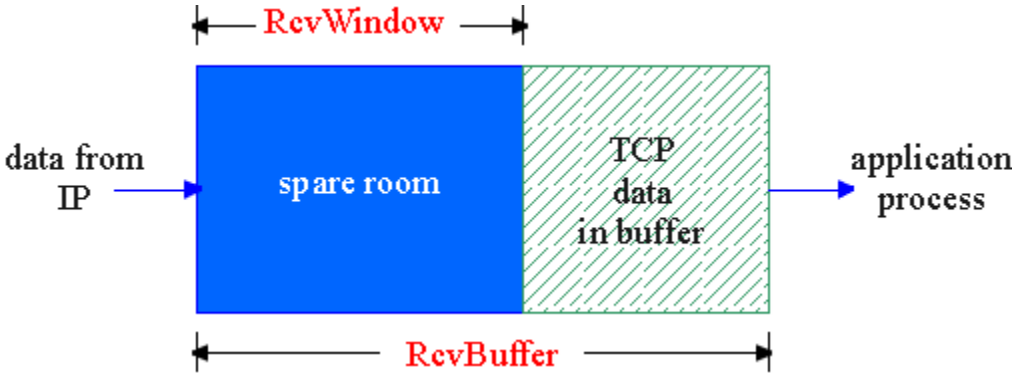
- príjemca má svoj buffer = okno príjemcu + nespracované dáta:



- proces aplikácie nemusí stíhať čítať z buffra

- odosielateľ nechce zahltiť príjemcu rýchlym posielaním správ
- snažíme sa odosielať takou rýchlosťou, akou to stíha príjemca spracúvať

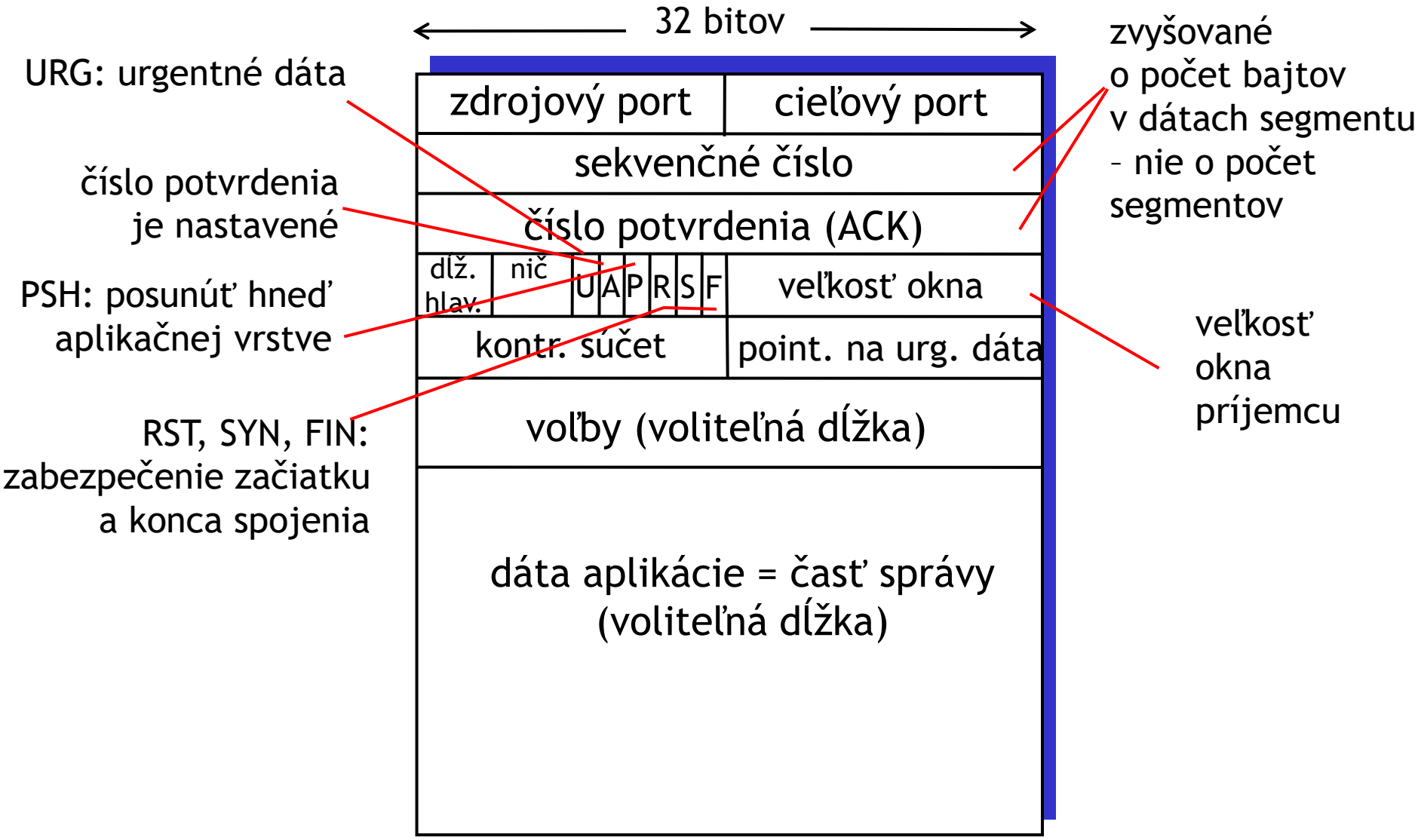
TCP: Kontrola toku dát



- voľné miesto v buffri príjemcu
- = **`RcvWindow`**
- = **`RcvBuffer`** - dáta nespracované aplikáciou

- Príjemca informuje odosielateľa o veľkosti **`RcvWindow`** v hlavičke segmentu
- Odosielateľ nastavuje veľkosť svojho okna podľa hodnoty **`RcvWindow`**
  - iba toľko nepotvrdených segmentov môže mať odoslaných
  - garantujeme, že príjemcu nezahltíme

Štruktúra TCP segmentu



Užitočné voľby v TCP hlavičke

- Hlavička obsahuje miesto pre ďalšie voľby
- Štandardná veľkosť MSS je 536 bajtov. Bežne sa však používa 1460 bajtov.
- Počas handshaku vieme nastaviť premennú MSS na vyššiu alebo nižšiu
- Maximálne číslo v TCP hlavičke pre veľkosť okna príjemcu je 65535 bajtov
- pri veľkosti MSS 1460 B máme iba okno iba do 44 MSS !
- Počas handshaku vieme nastaviť premennú WSCALE, ktorou vykonáme bitový posun čísla veľkosti okna v hlavičke.
- Určíme tak násobok dvojky, ktorým sa bude násobiť hodnota veľkosti okna v hlavičke počas celej komunikácie ➜ okno môže mať pokojne až 1 GB

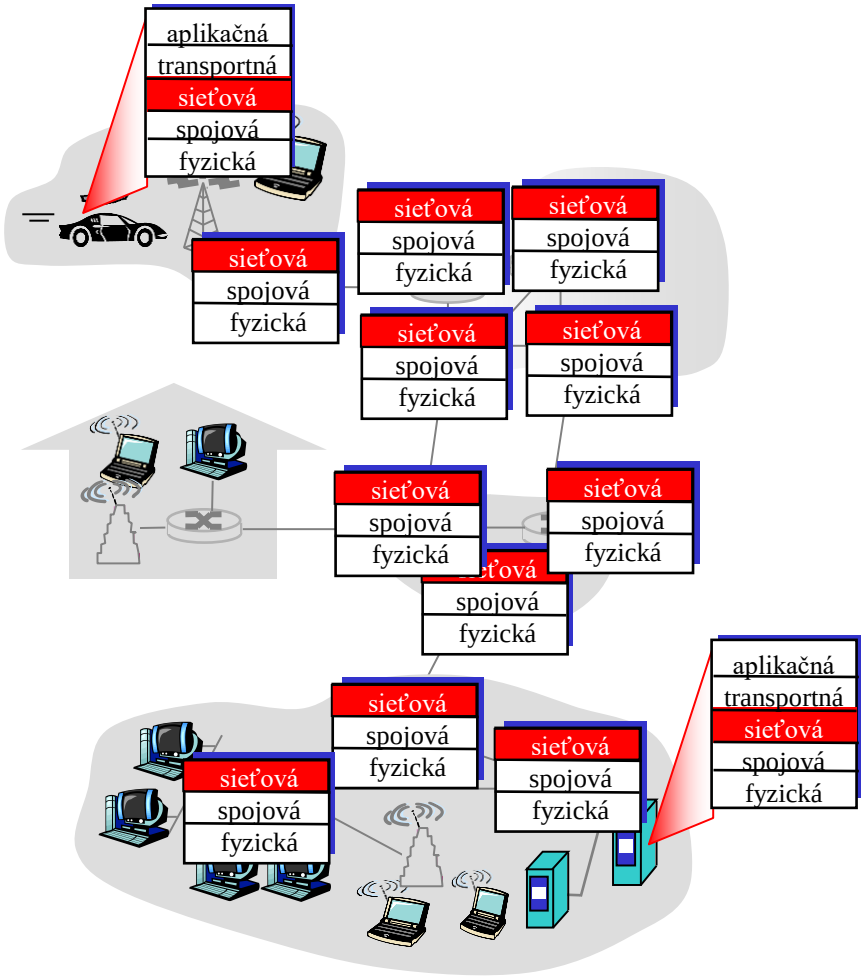
Kapitola 4: Sieťová vrstva

Náš cieľ:

- fungovanie smerovača (routra)
- sieť založená na datagramoch vs. okruhmi riadená sieť
- IP adresy, masky
- aplikačný protokol DHCP
- NAT
- ICMP
- IPv6

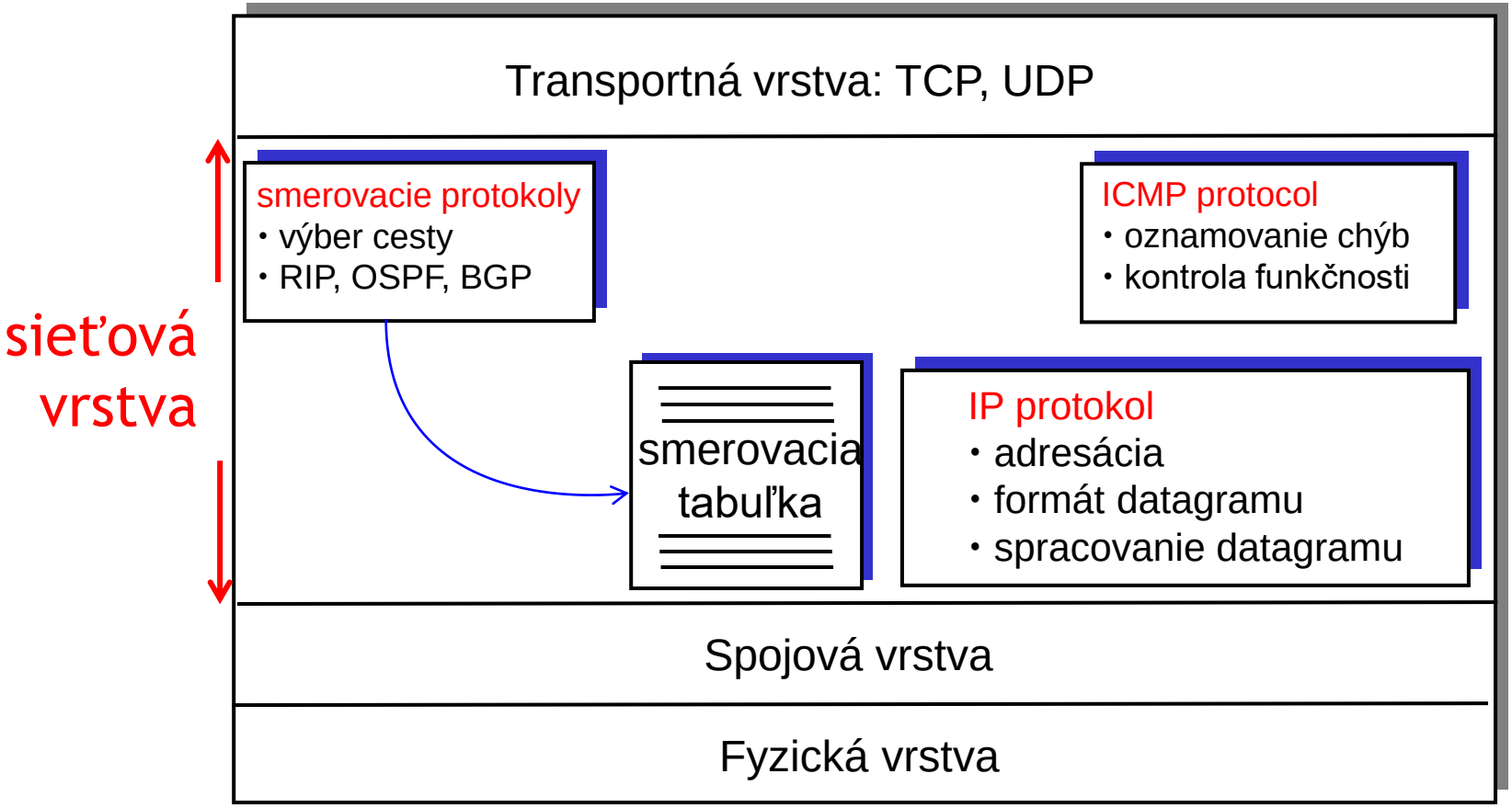
Sieťová vrstva

- zabezpečuje dopravenie segmentov transportnej vrstvy do cieľovej stanice
- u odosielateľa sa obalí segment do datagramu (dodá sa na začiatok hlavička datagramu)
- na strane príjemcu sa doručený datagram rozbalí a segment sa odovzdá transportnej vrstve
- sieťová vrstva je implementovaná v každej koncovej stanici a v každom smerovači (routri)
- smerovač číta hlavičky všetkých IP datagramov, ktoré cez neho prechádzajú

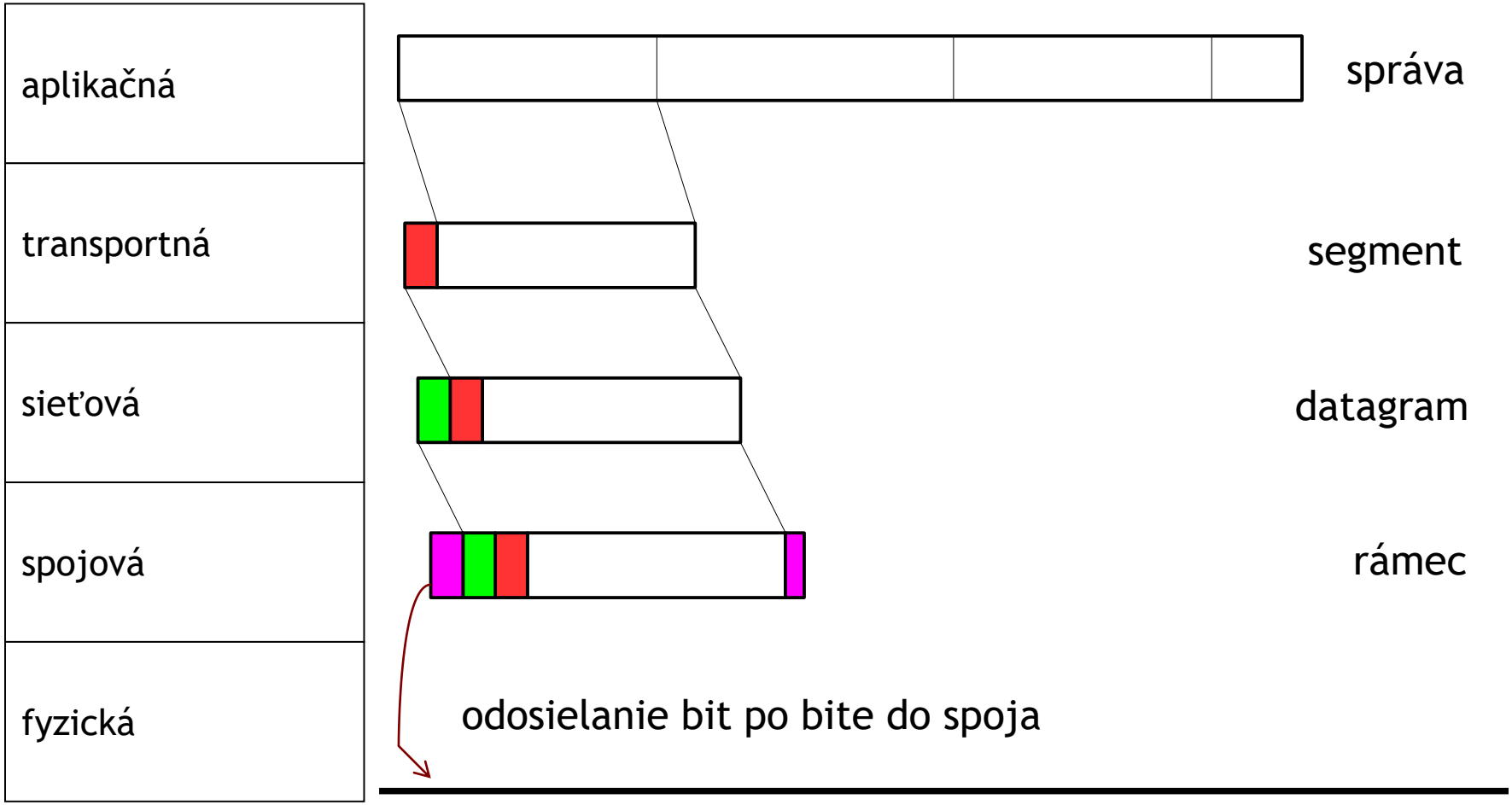


Sieťová vrstva internetu

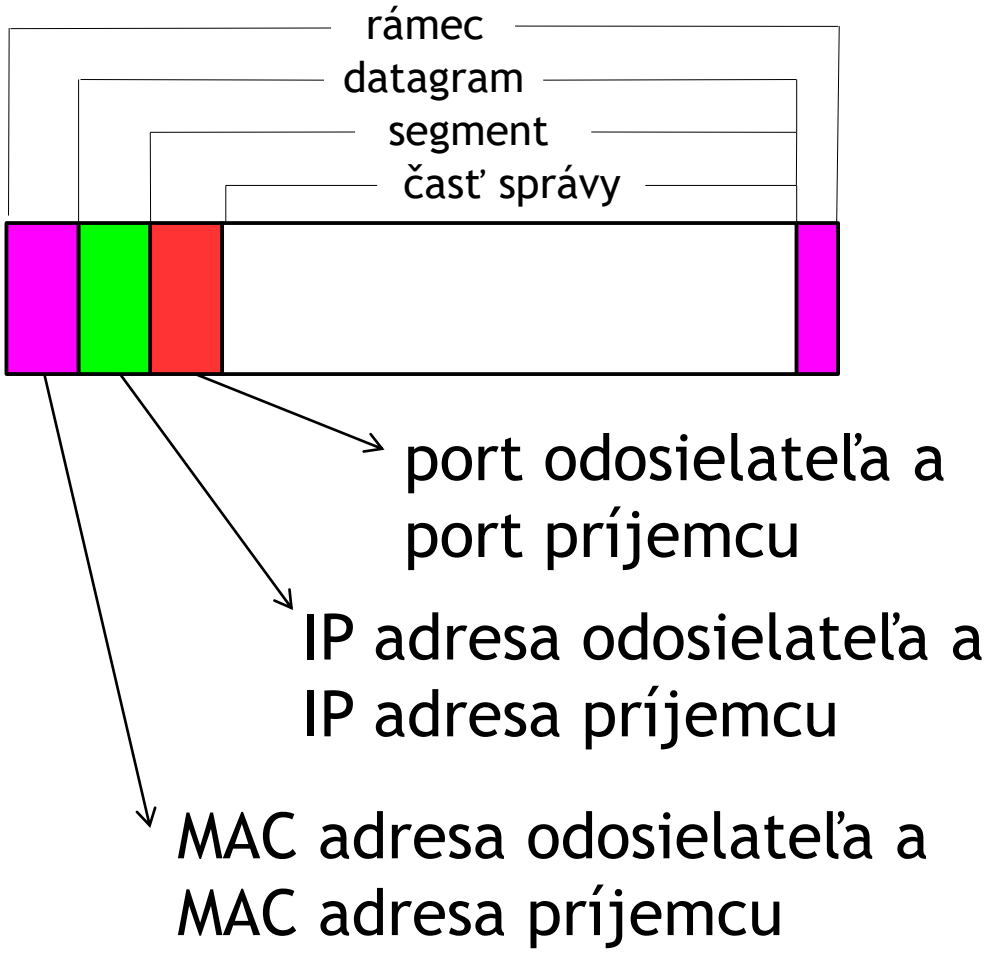
funkcie sieťovej vrstvy (stanice, smerovače):



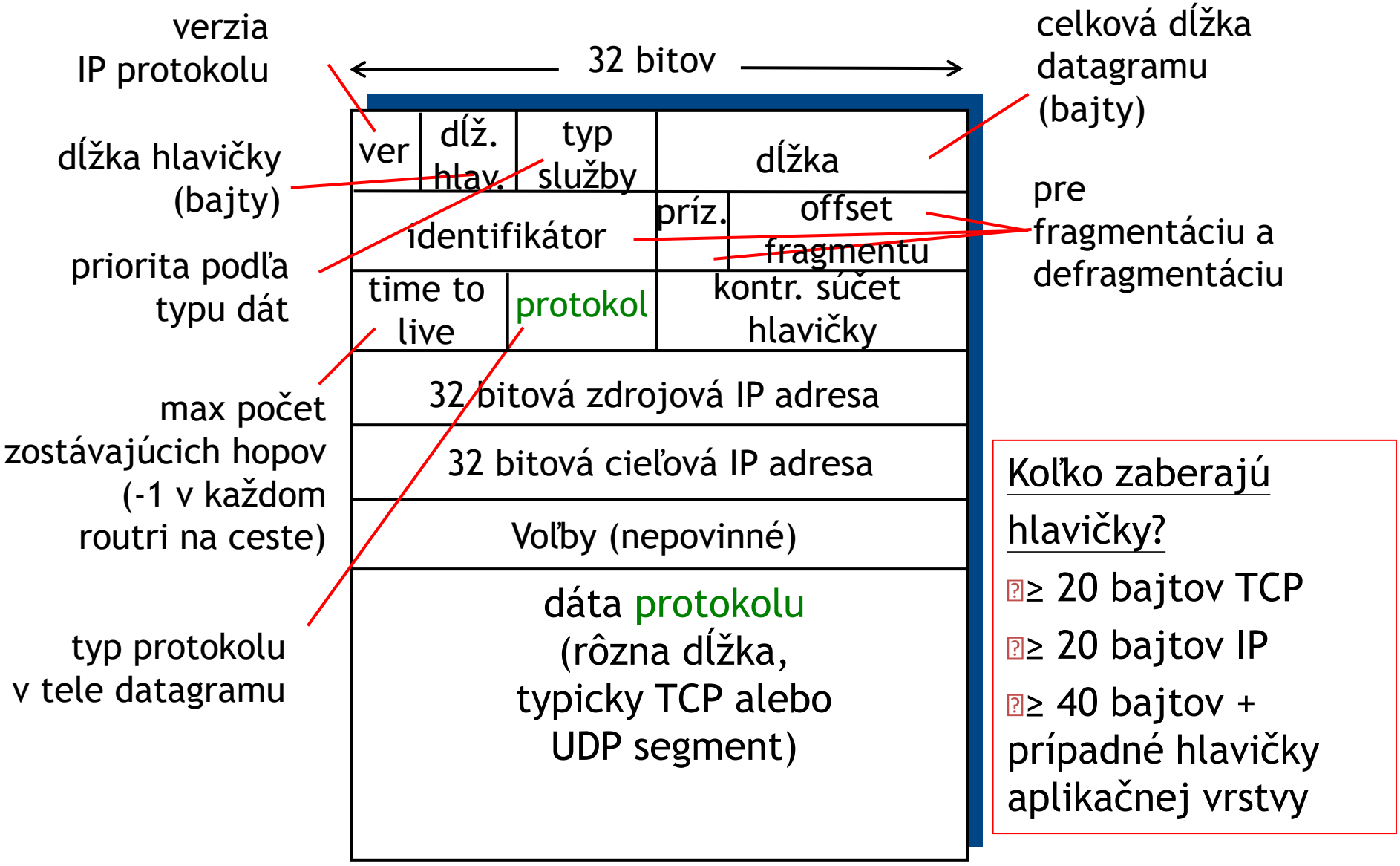
Tvorba paketu



Adresovacie údaje v pakete



Formát IPv4 datagramu

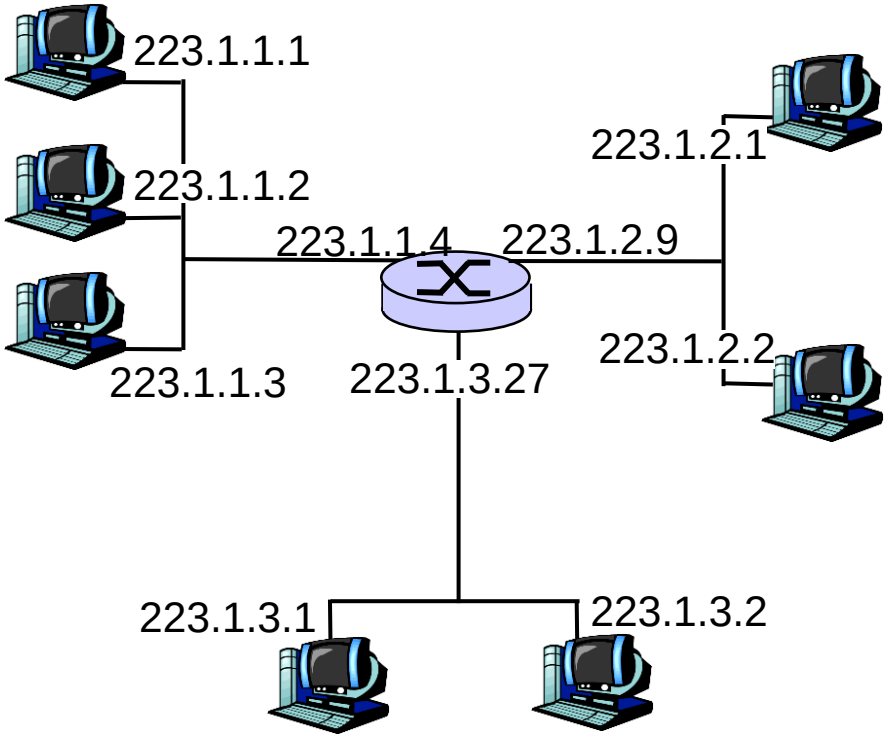


Koľko zaberajú hlavičky?

- ≥ 20 bajtov TCP
- ≥ 20 bajtov IP
- ≥ 40 bajtov + prípadné hlavičky aplikačnej vrstvy

IP adresácia: úvod

- IPv4 adresa: 32-bitový identifikátor koncovej stanice alebo rozhrania smerovača (routra)
- *rozhranie:* logická časť smerovača alebo stanice typicky priradená jednému fyzickému pripojeniu
  - smerovače majú obvykle mnoho rozhraní
  - koncová stanica má typicky jedno aktívne rozhranie
  - s každým rozhraním je asociovaná jedna IP adresa

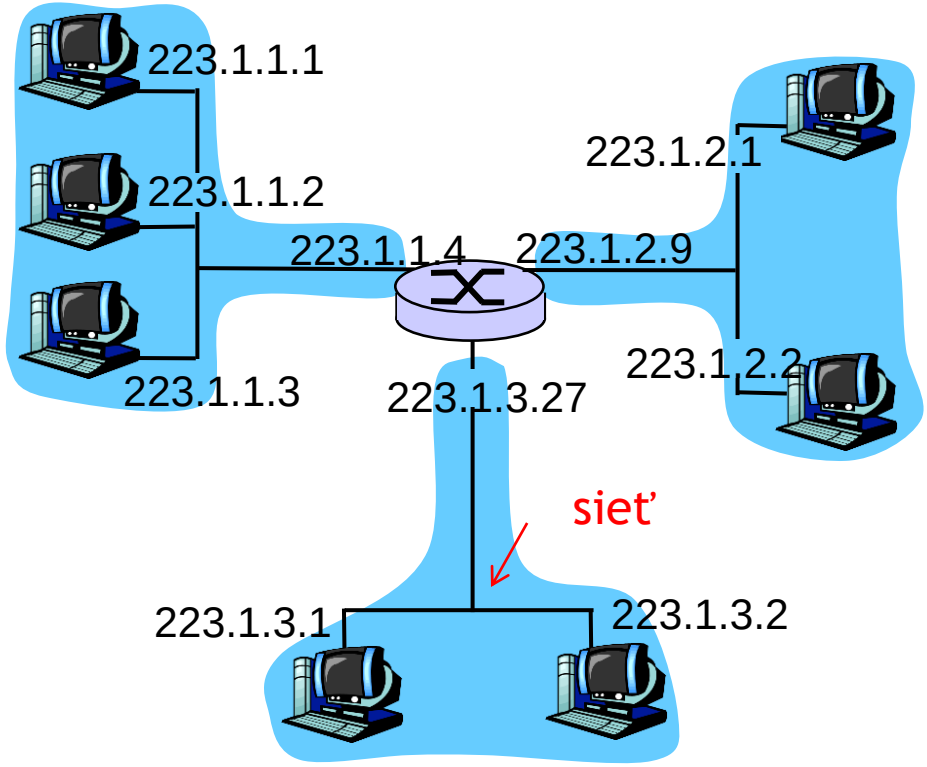


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Siete

- IP adresa:
  - každé rozhranie má svoju
- Sieť
  - rozhrania s rovnakou adresou siete (sieťová časť adresy je rovnaká)
  - zariadenia sú medzi sebou fyzicky prepojené bez smerovača

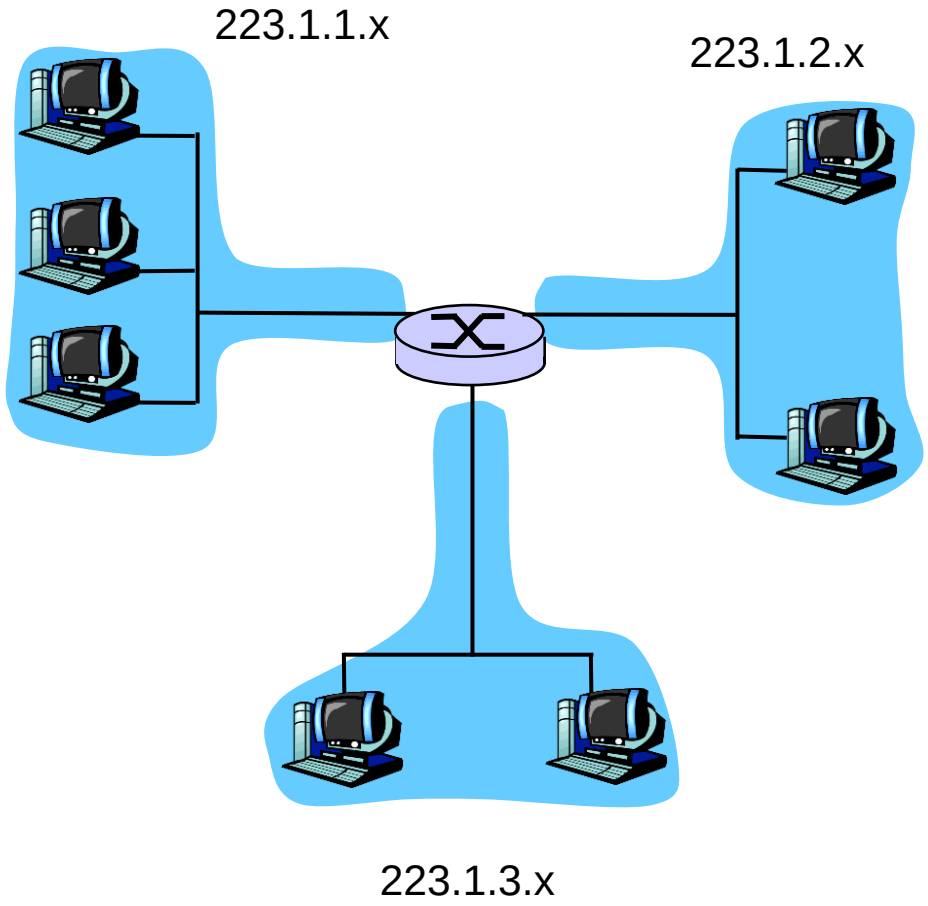


3 siete prepojené smerovačom

Siete

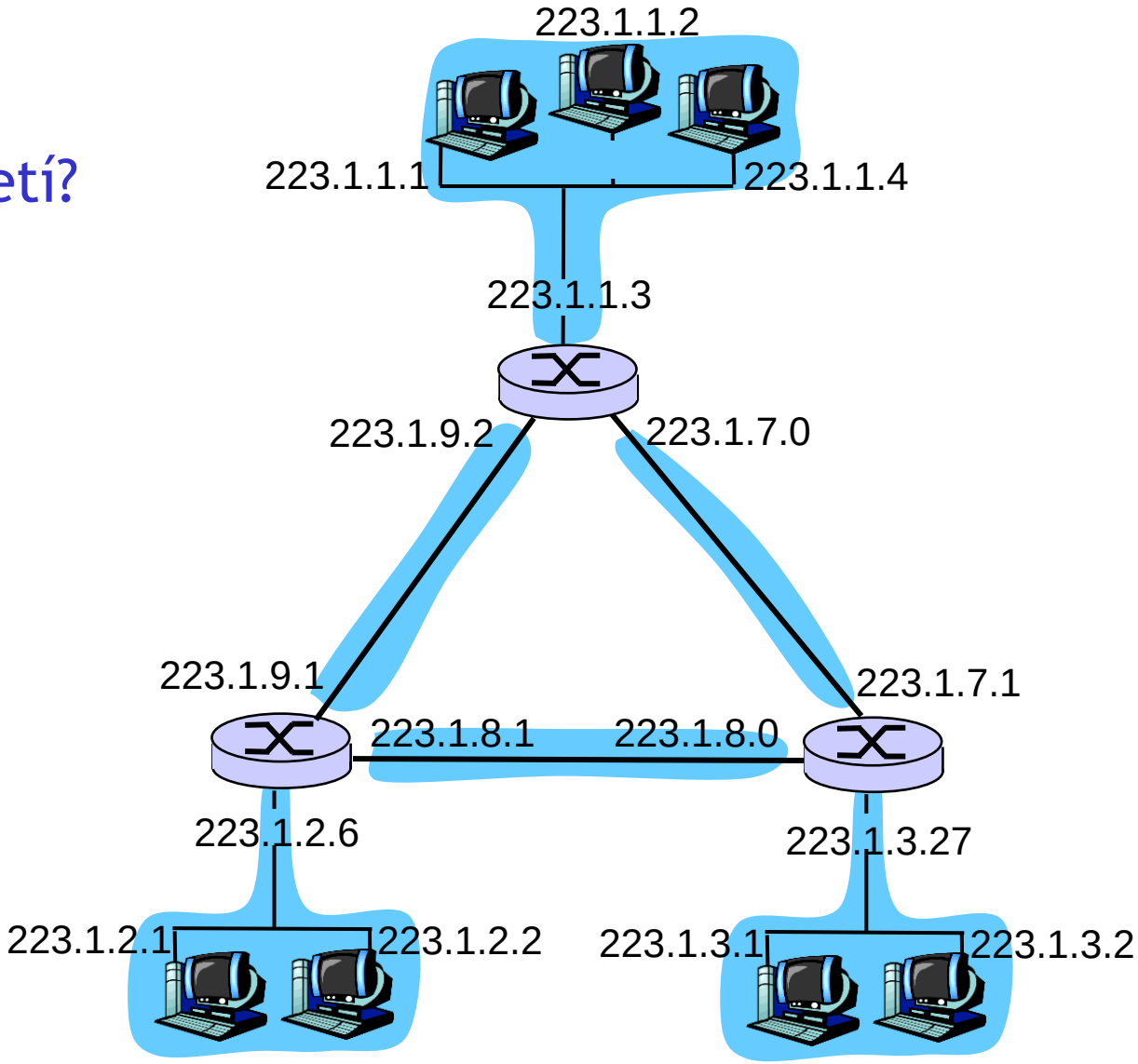
pomôcka:

- na zistenie toho, čo všetko je v jednej sieti, odpoj všetky rozhrania smerovačov
- ostanú “ostrovy” prepojených (a navzájom v pohode komunikujúcich) staníc
- každý “ostrov” predstavuje sieť



Siete

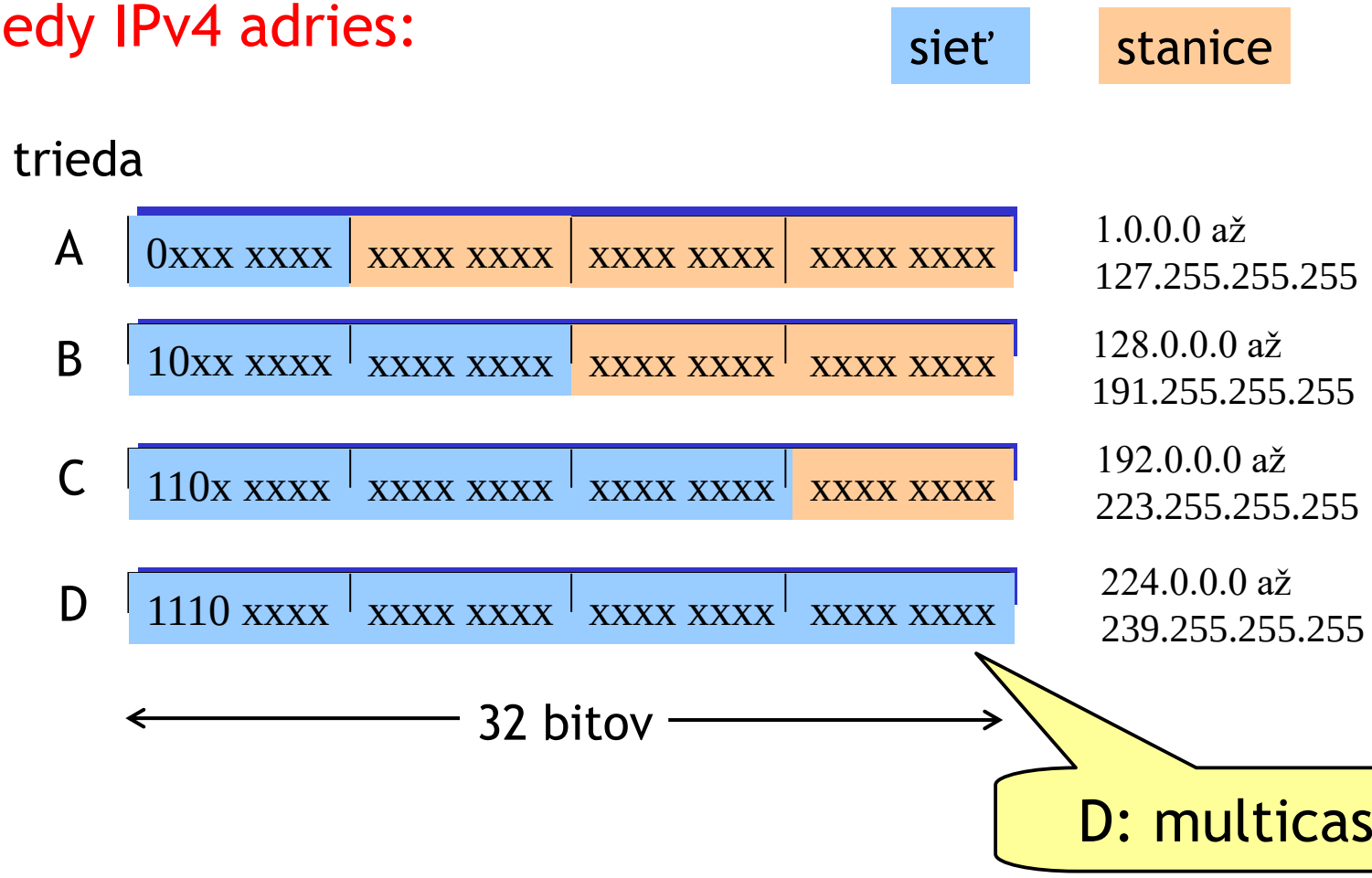
Koľko tu máme sietí?



IPv4 adresy

Pôvodné delenie IPv4 adries (organizácia IANA):

triedy IPv4 adries:



Jemnejšia IPv4 adresácia: CIDR

CIDR: Classless Inter-Domain Routing

- ľubovoľne dlhá úvodná časť IP adresy môže tvoriť sieťovú časť
- úplná adresa: a.b.c.d/x, kde x je počet bitov, ktoré predstavujú sieťovú časť

adresa: 11001000 00010111 00010000 00000101

200.23.16.5/23

maska: 11111111 11111111 11111110 00000000

255.255.254.0

Špeciálne IPv4 adresy

- xxx...x000...0/m – adresa siete

  m

- xxx...x111...1/m – broadcast (obežník) siete

  m

- 0.0.0.0/32 – moja adresa v lokálnej sieti
- 255.255.255.255 – broadcast lokálnej siete
- 127.0.0.0/8 – loopback, (127.0.0.1 = localhost)
- 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 – privátne siete

Výpočet adresy siete

adresa stanice: 200.23.16.5/23

11001000 00010111 00010000 00000101

logický AND

maska: 11111111 11111111 11111110 00000000

---

adresa siete: 11001000 00010111 00010000 00000000

200.23.16.0/23

Výpočet broadcastovej adresy siete

	200.23.16.5/23			
adresa stanice:	11001000	00010111	00010000	00000101
	logický OR			
invertovaná maska:	00000000	00000000	00000001	11111111
broadcast siete:	11001000	00010111	00010001	11111111
	200.23.17.255/23			

Výpočet broadcastovej adresy siete

	200.23.16.5/23			
adresa stanice:	11001000	00010111	00010000	00000101
	logický OR			
invertovaná maska:	00000000	00000000	00000001	11111111
broadcast siete:	11001000	00010111	00010001	11111111
	200.23.17.255/23			

Pozor! 200.23.17.0/23 je adresa stanice, nie siete

Vyskúšajte si

- Windows: ipconfig /all
  - MAC adresa (spojová vrstva), IP adresa, maska, default gateway, default DNS server, (DHCP server a dĺžka platnosti)
- Unix: ifconfig
  - MAC adresa, IP adresa, broadcastová adresa, maska
  - MTU = maximal transfer unit

Smerovacia tabuľka

4 miliardy možných cieľov

cieľ	maska	brána	rozhranie
200.23.24.0	255.255.255.0 (24)	0.0.0.0	1
200.23.16.0	255.255.248.0 (21)	0.0.0.0	3
200.23.24.0	255.255.248.0 (21)	0.0.0.0	2
0.0.0.0	0.0.0.0 (0)	200.23.1.1	3

cieľ od	cieľ do	rozh.
11001000 00010111 00011000 00000000	11001000 00010111 00011000 11111111	1
11001000 00010111 00010000 00000000	11001000 00010111 00010111 11111111	3
11001000 00010111 00011000 00000000	11001000 00010111 00011111 11111111	2
00000000 00000000 00000000 00000000	11111111 11111111 11111111 11111111	3

Kam pôjdu?:

11001000 00010111 00010110 10100001

11001000 00010111 00011000 10101010

Smerovacia tabuľka

4 miliardy možných cieľov

cieľ	maska	brána	rozhranie
200.23.24.0	255.255.255.0 (24)	0.0.0.0	
200.23.16.0	255.255.248.0 (21)	0.0.0.0	
200.23.24.0	255.255.248.0 (21)	0.0.0.0	
0.0.0.0	0.0.0.0 (0)	200.23.1.1	

najdlhší prefix

cieľ od	cieľ do	rozh.
11001000 00010111 00011000 00000000	11001000 00010111 00011000 11111111	1
11001000 00010111 00010000 00000000	11001000 00010111 00010111 11111111	3
11001000 00010111 00011000 00000000	11001000 00010111 00011111 11111111	2
00000000 00000000 00000000 00000000	11111111 11111111 11111111 11111111	3

Kam pôjdu?:

11001000 00010111 00010110 10100001

11001000 00010111 00011000 10101010

Smerovacia tabuľka

- smerovaciu tabuľku má aj každá stanica
- vyskúšajte si:
  - Unix: route -n alebo netstat -r
  - Windows: route PRINT alebo netstat -r

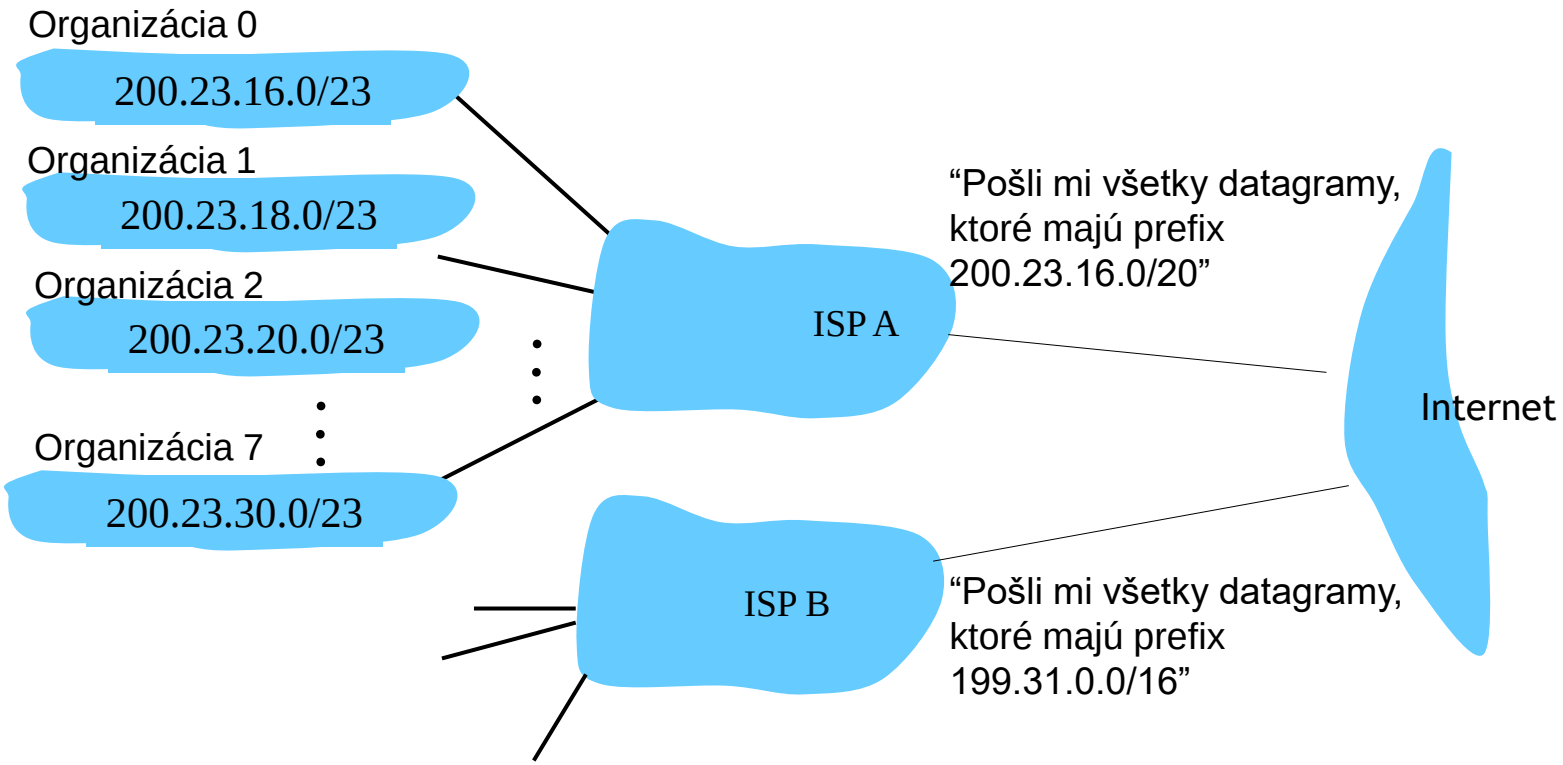
IP adresy: kto mi ich dá?

- ISP má pridelenú sieť s relatívne malým počtom jednotiek v maske
- vyrába podsiete zákazníkov, napríklad:

sieť ISP	11001000 00010111 00010000	00000000	200.23.16.0/20
Organizácia 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organizácia 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organizácia 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organizácia 7	11001000 00010111 00011110	00000000	200.23.30.0/23

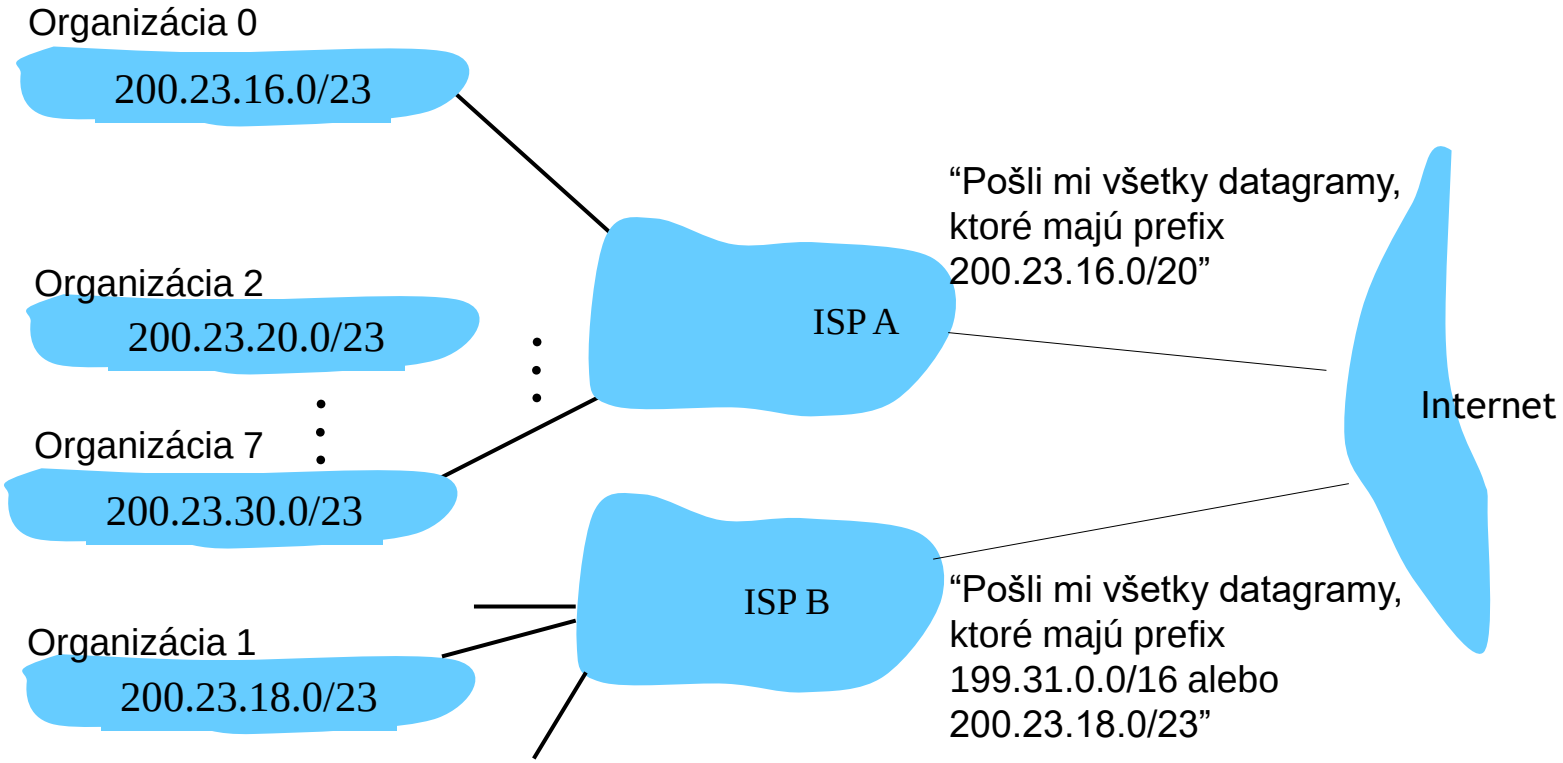
Adresovanie prefixom

- stačí menej záznamov v smerovacej tabuľke



Adresovanie prefixom

- keď Organizácia 1 zmení providera, nemusí meniť IP adresy
- využijeme princíp dlhšieho prefixu



Prideľovanie IP adries providerom

- organizácia IANA
- v Eurázii RIPE

- vyskúšajte si:
  - whois 158.197.0.0
  - http://www.db.ripe.net/whois

Delíme vlastnú sieť na podsiete

- Máme pridelenú sieť 200.23.16.0/23, na koľko nezávislých podsietí ju vieme rozdeliť?

počet 1 v maske	počet sietí	max. počet staníc v 1 sieti	max. počet pripojených staníc
23	1	510	510
24	2	254	508
25	4	126	504
26	8	62	496
27	16	30	480
28	32	14	448
29	64	6	384
30	128	2	256

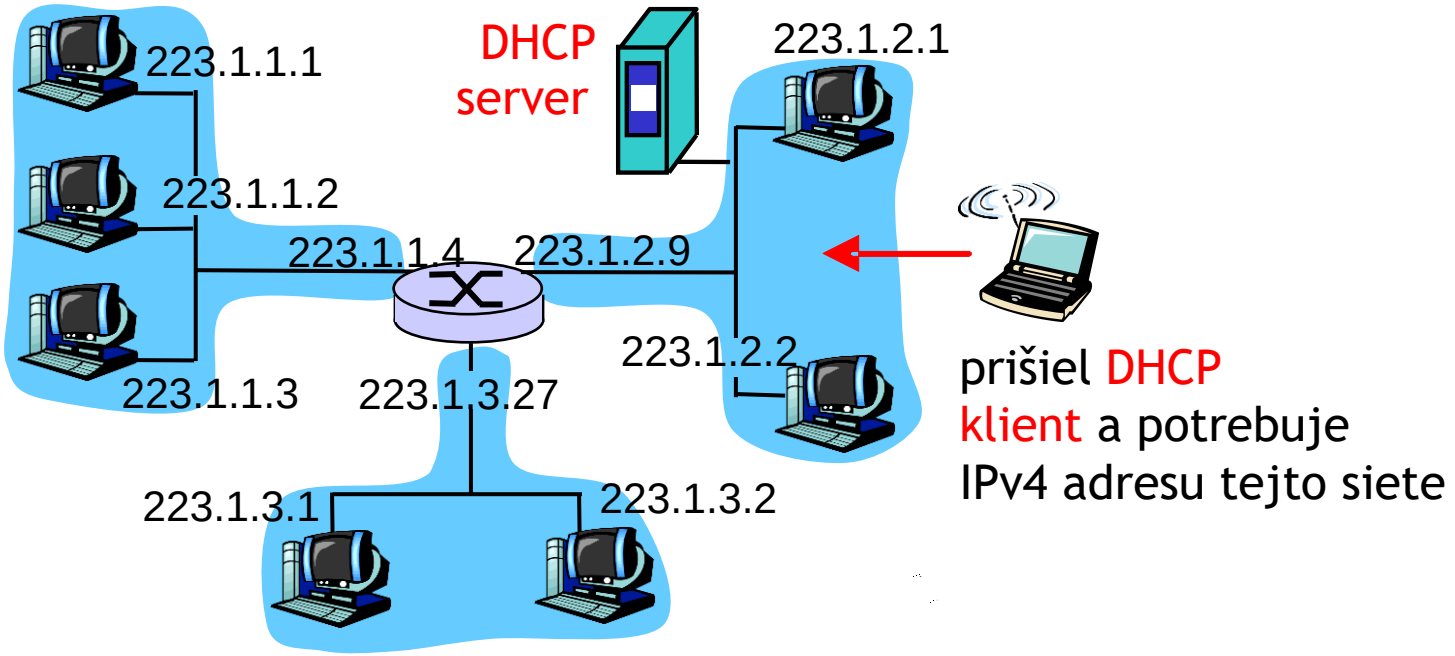
Ako nastaviť IPv4 adresu?

- nastavenie ručne (staticky) administrátorom
  - Windowsy: control-panel -> network -> configuration -> tcp/ip -> properties
  - UNIXy: ifconfig alebo NetworkManager alebo zmena konfiguračných súborov, ...
- DHCP: Dynamic Host Configuration Protocol: dynamické prideľovanie adries DHCP serverom
  - "plug-and-play"

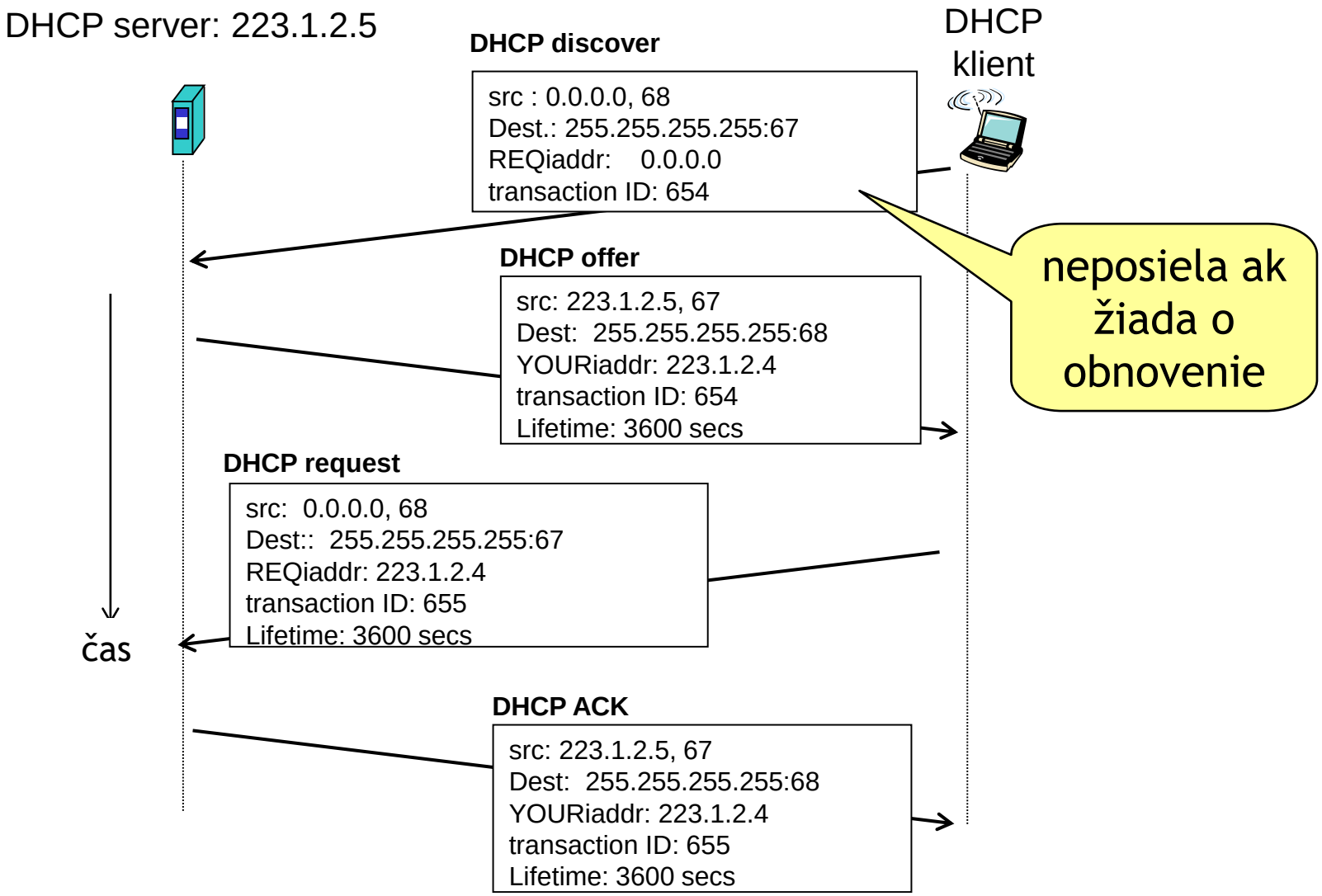
DHCP: Dynamic Host Configuration Protocol

- Cieľ: umožniť stanici získať *dynamicky* IPv4 adresu z DHCP servera v sieti okamžite po tom, ako sa stanica fyzicky pripojí do siete
  - Stanica môže požiadať o obnovenie prenajatej (lease) adresy, ktorú používala pred tým, alebo ktorú práve používa a končí jej platnosť
  - Jedna IPv4 adresa môže byť prideľovaná viacerým staniciam, pokiaľ sa prvá odpojila a prišla iná
    - viac používateľov pre malý počet voľných IPv4 adries
  - Ideálne pre mobilných používateľov
- Aplikačný protokol nad transportným protokolom UDP

DHCP scenár


DHCP
server
223.1.2.1
223.1.1.1
223.1.1.2
223.1.1.4
223.1.2.9
223.1.2.2
223.1.1.3
223.1.3.27
223.1.3.1
223.1.3.2
prišiel DHCP
klient a potrebuje
IPv4 adresu tejto siete

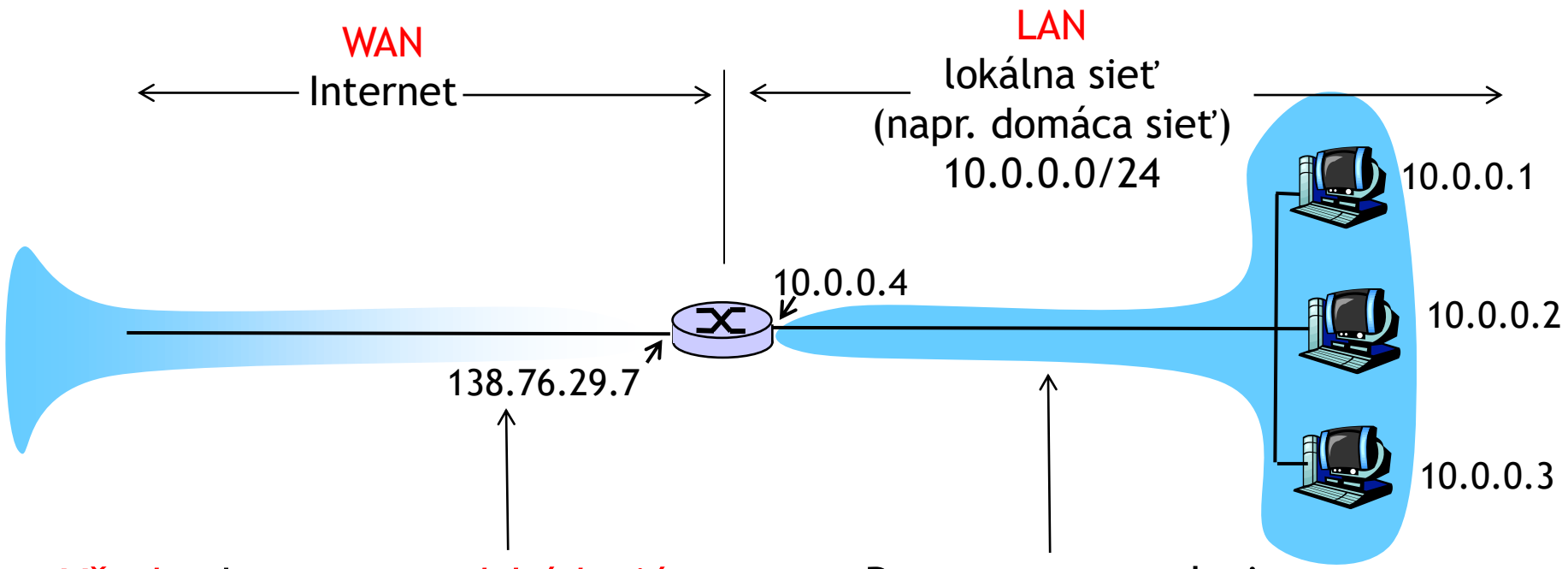
DHCP komunikácia



DHCP server poskytne

- IPv4 adresu
- masku
- default router
- default DNS servery
- dĺžku platnosti (lease time)

NAT: Network Address Translation



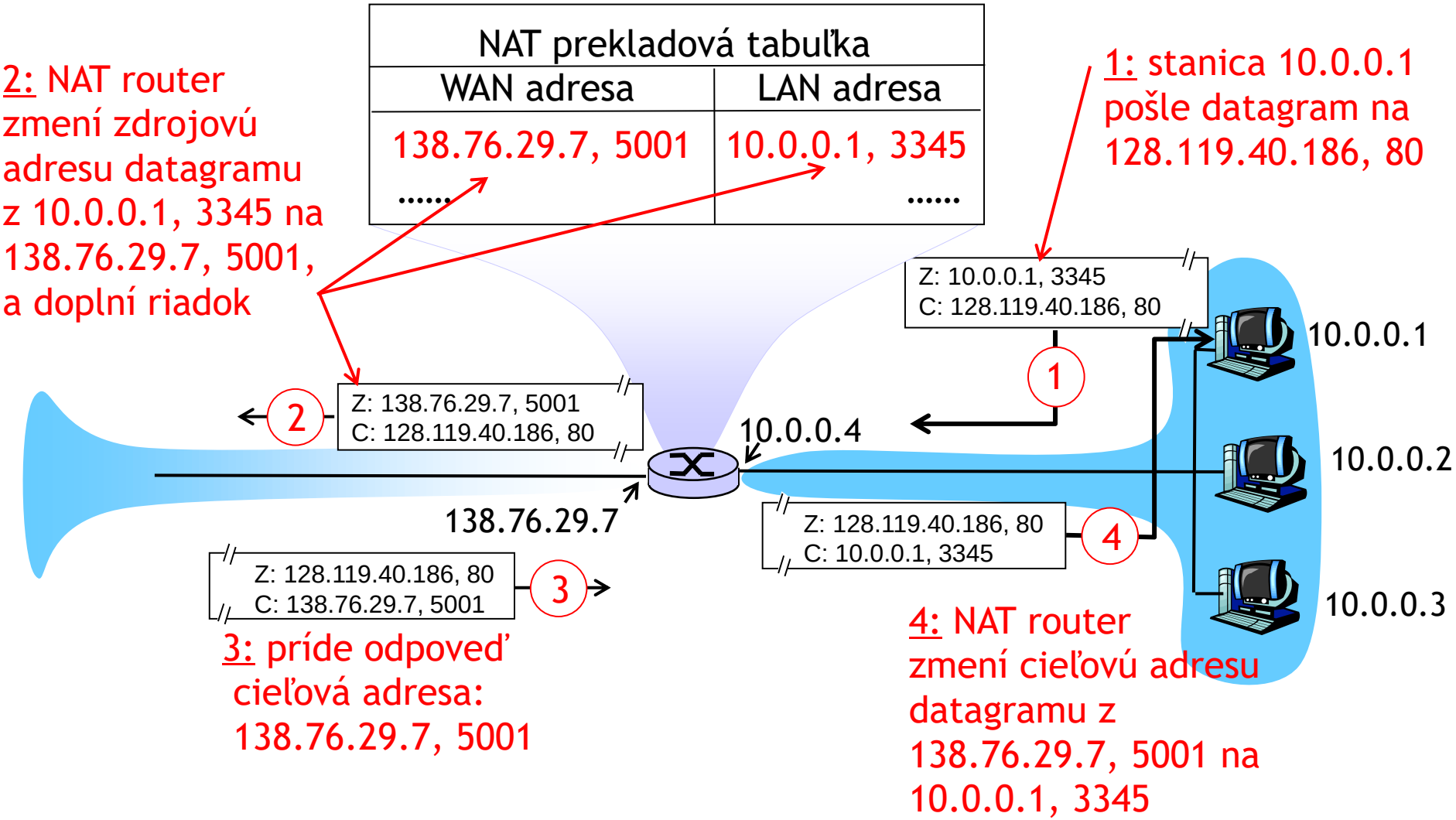
Všetky datagramy *odchádzajúce* z lokálnej siete majú rovnakú zdrojovú WAN IPv4 adresu: 138.76.29.7, rôzne môžu byť zdrojové porty

Datagramy so zdrojovou a zároveň cieľovou IPv4 adresou z vnútra siete 10.0.0.0/24 fungujú tak, ako obvykle

NAT: Network Address Translation

- Motivácia: lokálna sieť sa javí navonok (z internetu) ako jeden počítač s jedinou IPv4 adresou
  - stačí, ak nám provider (napr. Antik) pridelí jednu IPv4 adresu a môžeme pripájať viac počítačov
  - môžeme meniť IPv4 adresy vo vnútri lokálnej sieti bez účasti providera, aj klientov pre naše servery
  - môžem zmeniť ISP bez zmeny vnútornej adresácie v lokálnej sieti
  - zariadenia vo vnútornej sieti nie sú explicitne adresovateľné a viditeľné z vonku – nevieme ich priamo napadnúť (vylepšenie bezpečnosti).

NAT: postup

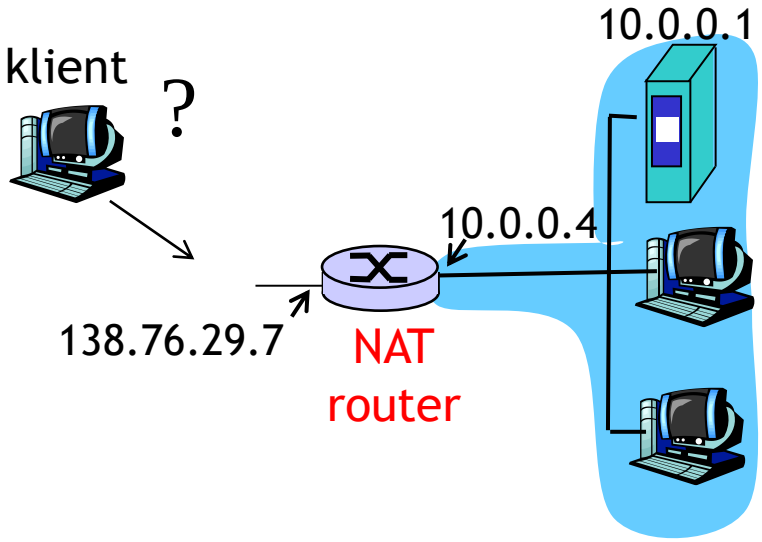

NAT prekladová tabuľka
WAN adresa
LAN adresa
138.76.29.7, 5001
10.0.0.1, 3345
.....
.....
2: NAT router zmení zdrojovú adresu datagramu z 10.0.0.1, 3345 na 138.76.29.7, 5001, a doplní riadok
1: stanica 10.0.0.1 pošle datagram na 128.119.40.186, 80
Z: 10.0.0.1, 3345
C: 128.119.40.186, 80
1
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
138.76.29.7
2
Z: 138.76.29.7, 5001
C: 128.119.40.186, 80
Z: 128.119.40.186, 80
C: 10.0.0.1, 3345
4
Z: 128.119.40.186, 80
C: 138.76.29.7, 5001
3
3: príde odpoveď
cieľová adresa:
138.76.29.7, 5001
4: NAT router zmení cieľovú adresu datagramu z 138.76.29.7, 5001 na 10.0.0.1, 3345

NAT: Network Address Translation

- čísla portov môžu byť 0 - 65535
  - S jedinou IPv4 adresou vieme robiť naraz vyše 65000 spojení!
- NAT je kontroverzný:
  - obvykle ako doplnková služba routrov, ktoré by inak mali rozbaľovať iba po tretiu vrstvu
  - dva počítače, každý za iným NATom, nevedia za normálnych okolností priamo komunikovať
  - možnosť, že klient je za NATom, musí byť braná do úvahy pri tvorbe sieťových aplikácií, napr. pri P2P
  - zánik by mal priniesť protokol IPv6

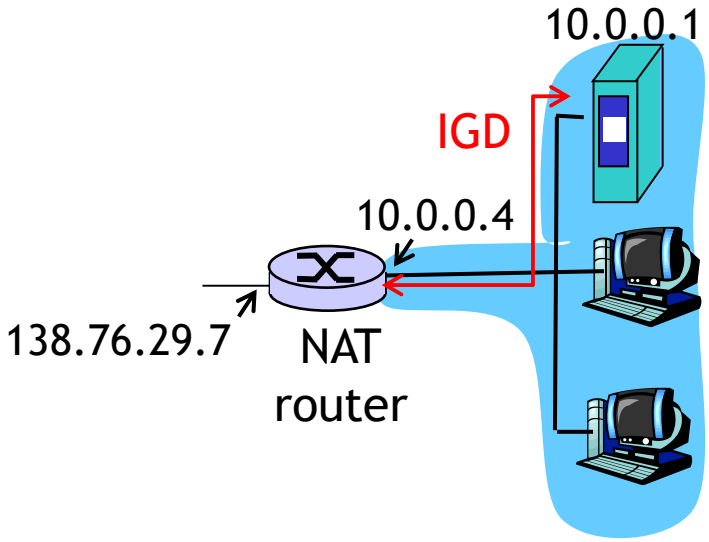
NAT traversal problem

- klient sa chce napojiť na server na adrese 10.0.0.1
  - 10.0.0.1 je lokálna adresa neexistujúca na internete (klient ju nemôže použiť ako cieľovú adresu)
  - jediný viditeľný je WAN port NAT routra: 138.76.29.7
- riešenie 1: statická konfigurácia NATu na preposielanie všetkých požiadaviek na určený port priamo na správny server
  - napr. (138.76.29.7, port 2500) preposielame na (10.0.0.1, port 25000)



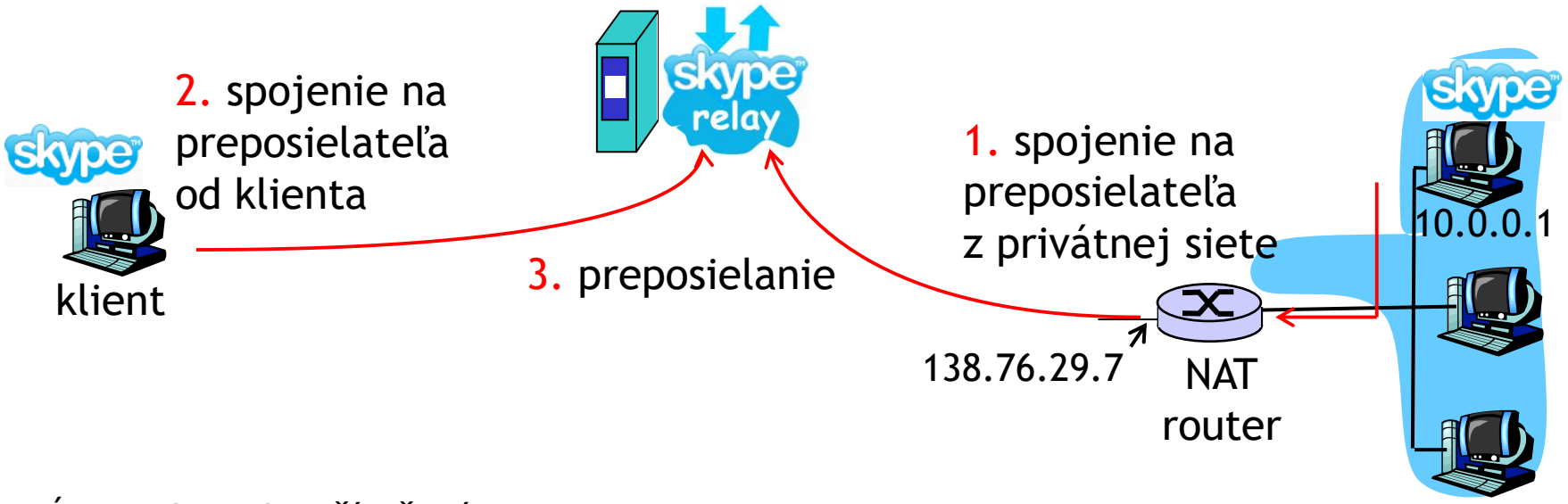
NAT traversal problem

- riešenie 2: Univerzálny Plug and Play (UPnP) Internet Gateway Device (IGD) protokol. Umožňuje stanici v lokálnej sieti:
  - zistiť verejnú IP adresu NAT routra (138.76.29.7)
  - zistiť existujúce mapovania portov
  - pridávať/odoberať mapovania portov (na daný čas)



NAT traversal problem

- riešenie 3: relaying/preposielanie (napr. v Skype)
  - príjemca hovoru vytvorí spojenie na preposielateľa
  - externý klient sa tiež napojí na preposielateľa
  - preposielateľ preposiela datagramy komunikácie



ICMP: Internet Control Message Protocol

- používaný stanicami a routrami na výmenu sieťových informácií
  - hlásenie chýb: nedostupná stanica, sieť, port, protokol
  - echo request/reply (používané programom ping)
- sieťová vrstva "nad" IP:
  - ICMP správy vo vnútri IPv4 datagramov
- ICMP správa: typ, kód, plus prvých 8 bajtov IP datagramu, ktorý spôsobil vytvorenie správy

Typ	Kód	popis
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest. host unreachable
3	2	dest. protocol unreachable
3	3	dest. port unreachable
3	6	dest. network unknown
3	7	dest. host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Skúste si ping

- ping -t 3 adresa (unix), -i (windows)
  - Time To Live = 3
- ping -b broadcastováAdresa (unix)
  - pingujeme všetkých v našej sieti
- ping -s veľkosť adresa, -l (windows)

nastavíme veľkosť ICMP paketu (bez hlavičky) (keď je výsledný paket väčší ako MTU, nastáva IP fragmentácia)

Traceroute a ICMP

- Pošleme sériu UDP segmentov k cieľu (alebo TCP či ICMP)
  - prvé 3 majú TTL =1
  - druhé 3 majú TTL=2, atď.
  - nepravdepodobné číslo portu
- Keď datagram vyslaný s TTL=n dôjde k n-tému routru:
  - router zahodí datagram
  - pošle na zdrojovú adresu ICMP správu "TTL expired" (type 11, code 0)
  - IP datagram s ICMP správou má zdrojovú adresu routra, ktorý ju posielal
- Keď príde ICMP správa, vypočítame RTT

Ukončenie

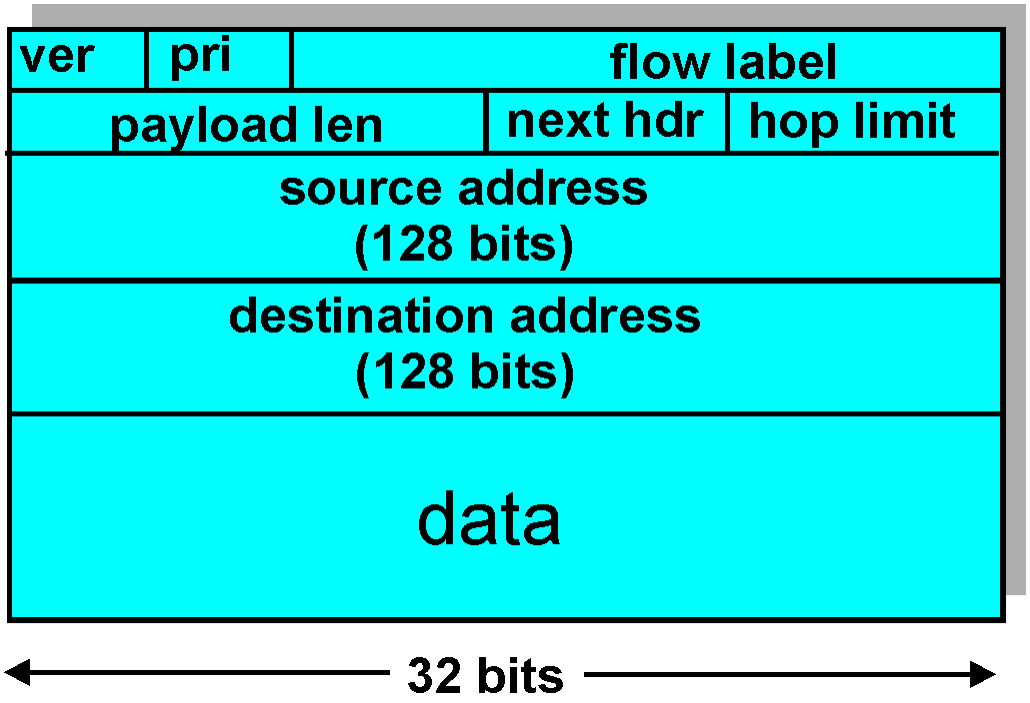
- UDP segment môže dôjsť až k cieľu
- Cieľová stanica vráti ICMP správu "port unreachable" (type 3, code 3)
- Keď odosielateľ dostane túto správu, končí

IPv6

- úvodná motivácia (1992-1995): 32-bitové adresy budú čoskoro všetky vyčerpané
  - prišiel však CIDR a presadil sa NAT
- ďalšia motivácia:
  - formát hlavičky umožňuje rýchlejšie spracovanie a smerovanie
  - hierarchické prideľovanie adries
  - podpora anycastu a lepšia podpora multicastu
  - autokonfigurácia koncových zariadení
  - lepšie riešenie mobility
- formát hlavičky datagramu IPv6:
  - presne 40 bajtová hlavička (žiadne voliteľné časti hlavičky), ale možnosť ďalších doplňujúcich hlavičiek
  - fragmentácia už len vo voliteľnej hlavičke vytvorenej odosielateľom (nie routrami)

IPv6 hlavička

Priorita (traffic class): identifikuje prioritu datagramu
Flow label: identifikácia toku dát
Next header: identifikácia rozširujúcej hlavičky alebo protokolu "vyššej vrstvy" v dátach

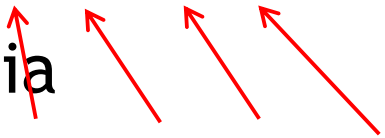


Niektoré zmeny oproti IPv4

- *Kontrolný súčet* : zrušený ako redundantný
  - kontrolné súčty sa robia v transportnej aj spojovej vrstve
  - netreba kontrolovať na routroch (šetríme čas)
- *Options:* nahradené ďalšími hlavičkami.
  - V časti "Next Header" sa môže namiesto identifikácie transportného protokolu dať identifikácia "ďalšej špeciálnej IP hlavičky"
- *ICMPv6:* nová verzia ICMP
  - nové typy správ, napr. "Packet Too Big"
  - funkcie na riadenie multicastu (namiesto IGMP)
  - riadenie autokonfigurácie

IPv6 adresy

- úplný výpis všetkých bitov (hexadecimálne po dvoch bajtoch v slove)
  - fe80:0000:0000:0000:0221:5cff:fe64:d39a
- s odstránenými úvodnými nulami slov
  - fe80:0:0:0:221:5cff:fe64:d39a
- vynechané nulové sekvencie
  - fe80::221:5cff:fe64:d39a
- mixovaná notácia
  - fe80::221:5cff:254.100.211.154



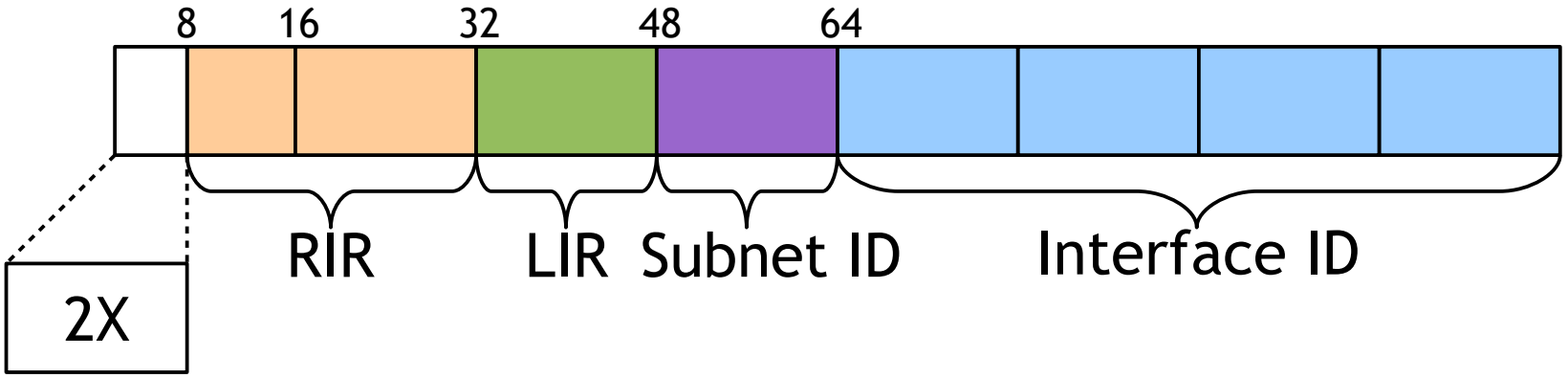
Niektoré špeciálne IPv6 adresy

- nešpecifikovaná lokálna IP adresa
  - ::/128 resp. 0:0:0:0:0:0:0:0/128 (analógia 0.0.0.0 z IPv4)
- loopback, localhost
  - ::1/128 resp. 0:0:0:0:0:0:0:1/128 (analógia 127.0.0.1 z IPv4)
- reprezentácia IPv4 adries v IPv6 notácii (IPv4-mapované adresy v SIIT [RFC 2765]) – v praxi sa nepoužíva
  - napr. ::ffff:158.197.31.4/128
- lokálne adresy fe80::/10 – novinka!
  - fe8X, fe9X,feaX,febX – IPv6 adresy v lokálnej sieti
  - IPv4 analógia nie je
- site-local (lokálna sieť sídla/organizácie) fec0::/10 – zrušené!

Niektoré špeciálne IPv6 adresy

- unique-local (unikátne privátne adresy) fc00::/7 (fcXX, fdXX)
  - takmer analógia sietí 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
  - sieťový prefix tvorený z MAC adresy a dátumu – vysoká pravdepodobnosť unikátnosti! (RFC 4193)
- broadcast – zrušený!
  - analógia 255.255.255.255/32 a broadcastových adries sietí napr. 158.197.35.255/24
- multicastové adresy ff00::/8 (ffXX:hocičo)
  - analógia triedy D: 224.0.0.0/4
- globálne (celosvetové adresy) 2000::/3
  - 2XXX:hocičo a 3XXX:hocičo
  - najčastejšie 2001:hocičo
  - UPJŠ IPv4: 158.197.0.0/16
  - UPJŠ IPv6: 2001:4118:400/48

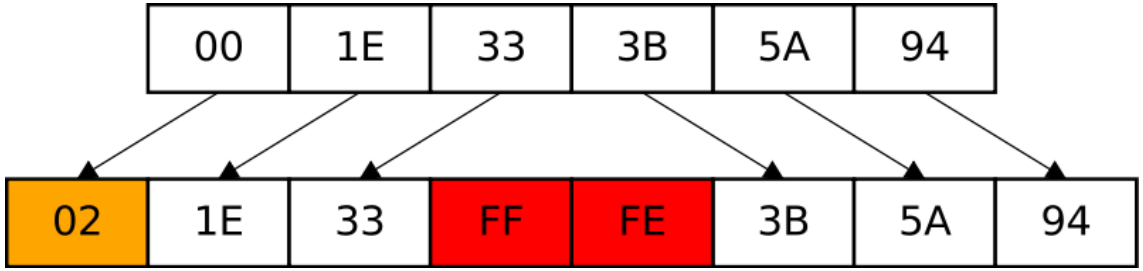
Štruktúra globálnych unicastových IPv6 adries



- RIR – IANA + regionálni registrátori (RIPE NCC, ARIN, APNIC, AFRINIC, LACNIC)
- LIR – lokálni registrátori prideľujú siete s prefixom dlhým 48-56 bitov
- Subnet ID – podsieť organizácie (65 535 alebo 256 možných podsietí)
- Interface ID – identifikátor rozhrania (sieťovej karty)

Interface ID

- máme k dispozícii $2^{64} \approx 10^{18}$ adries zariadení v jednej podsieti
- IPv6 EUI-64
  - odvodenie z MAC adresy (48 bitov na 64 bitov)



- Privacy extensions for stateless address autoconfiguration in IPv6
  - náhodné koncovky pravidelne menené

Nastavovanie IPv6 adresy

- ručne (hlavne servery s DNS AAAA záznamami)
- automatizovane
  - prideľovanie default routra
    - SLAAC (stateless address autoconfiguration) (RFC 2462)
  - prideľovanie sieťovej časti
    - SLAAC
    - stavový DHCPv6 – vrátane nejakého identifikátora rozhrania (nebráni to stanici dodať si ďalšie identifikátory podľa ľubovôle)
  - prideľovanie rekurzívnych lokálnych DNS serverov
    - DHCPv6 alebo DHCP(v4) ak máme dual-stack
    - od novembra 2010 aj cez SLAAC (RFC 6106)
  - prideľovanie identifikátora rozhrania
    - stanica si ho prideľuje sama (EUI 64 alebo privacy extensions)
    - a/alebo ho dostane od stavového DHCPv6

SLAAC + DHCPv6

- stanica si nastaví lokálnu adresu fe80::niečo
- stanica si overí unikátnosť cez detekciu duplicitných adries (typ paketu ICMPv6 na fe02::1:ffxx:xxxx) a prípadne nastaví inú
- stanica vyšle “router solicitation” (typ paketu ICMPv6)
- router mu zašle “router advertisment“ (typ paketu ICMPv6) s adresou brány a prípadnými ďalšími parametrami
  - ak príznak M=1 a O=0, má sa ešte použiť stavové DHCPv6
  - ak príznak M=0 a O=1, má sa ešte použiť bezstavové DHCPv6
  - ak príznak M=0 a O=0, v sieti sa nenachádza DHCPv6
- stavové DHCPv6 : stanica požiada o celú IPv6 adresu a ostatné parametre
- bezstavové DHCPv6 : stanica má sieťový prefix z “router advertisment” paketu a ďalšie parametre, hlavne lokálne rekurzívne DNS servery má získať z DHCPv6 serveru

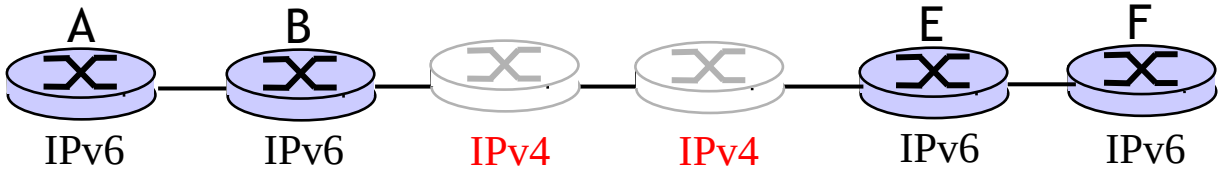
Prechod od IPv4 k IPv6

- Nemôžeme všetky zariadenia vymeniť naraz
  - žiaden “flag day” = “deň D”
  - Ako má sieť fungovať s pomiešanými IPv4 a IPv6 routrami?
- riešenia:
  - dual-stack: zariadenia zvládajú IPv4 aj IPv6
  - bezstavový preklad: SIIT, NAT64, TRT, BIH, SOCKS64
  - tunelovanie: IPv6 prenášaný ako telo IPv4 datagramu cez IPv4 routre: server/broker, 6to4, 6rd, 6over4, ISATAP, Teredo

Tunelovanie

Logický pohľad:

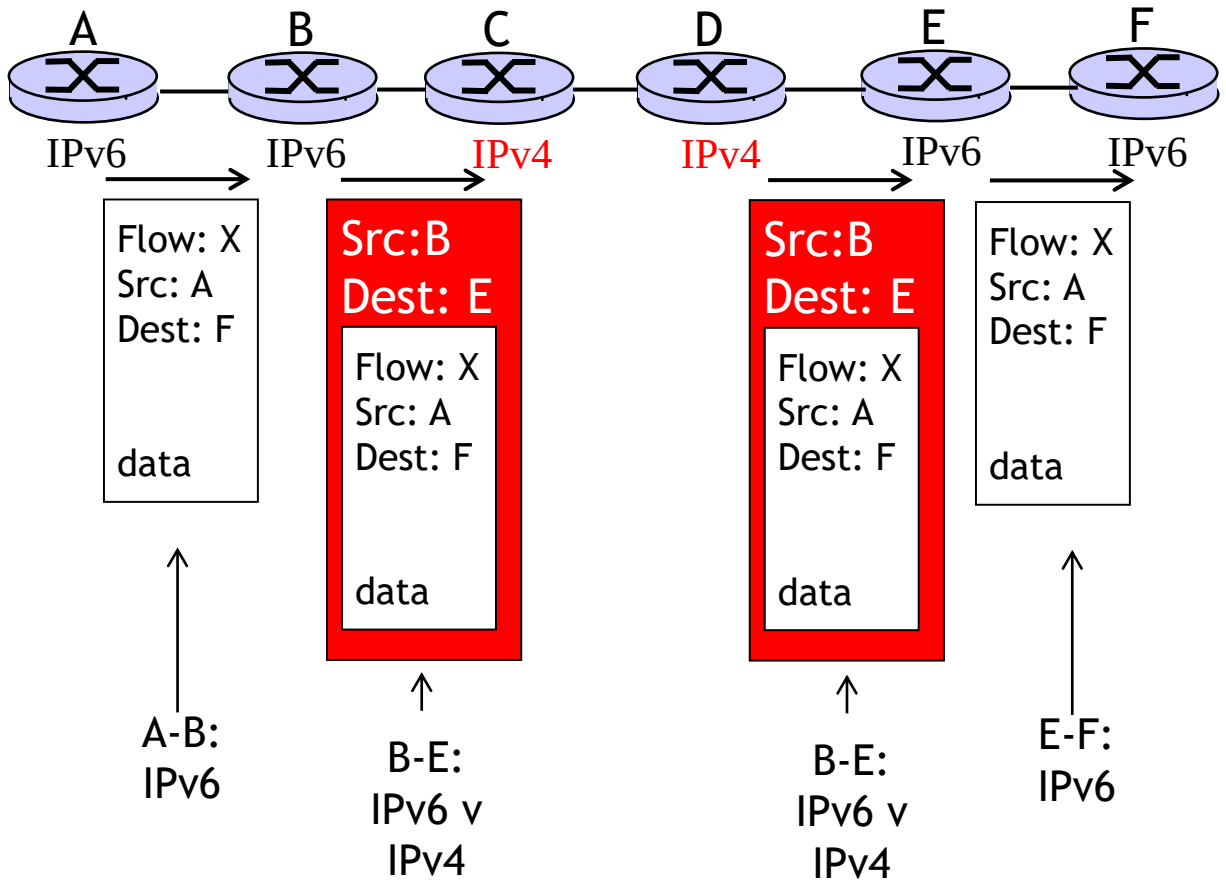
Fyzický pohľad:



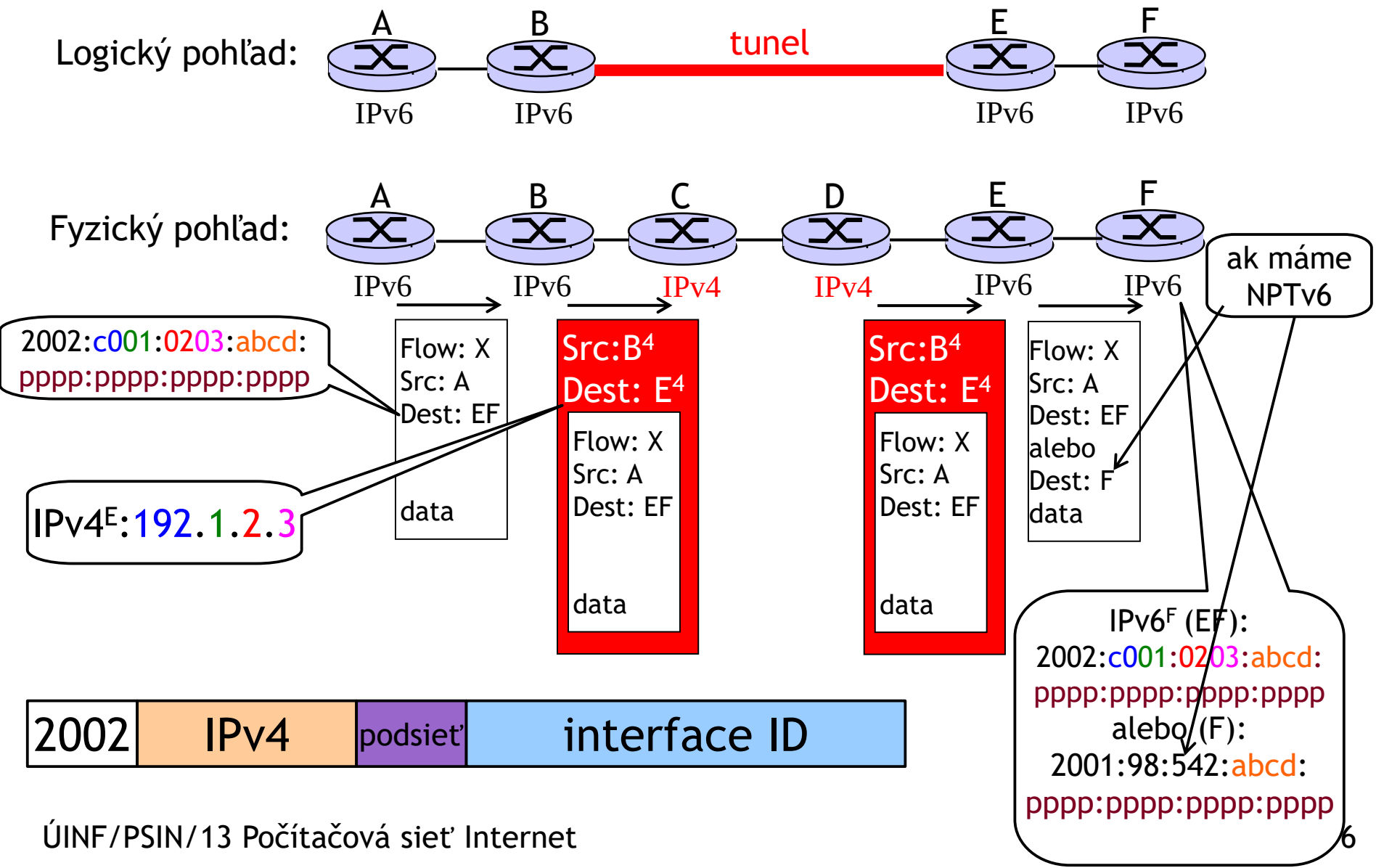
Tunelovanie (cez konfigurovaný tunel)

Logický pohľad:

Fyzický pohľad:



Tunelovanie [RFC 3056] (6to4)



Ďakujem za pozornosť

Modifikované slajdy z knihy:

Computer Networking: A Top Down Approach ,
4th edition.
Jim Kurose, Keith Ross
Addison-Wesley, July 2007.