



Katedra softwarového inženýrství,
Matematicko-fyzikální fakulta,
Univerzita Karlova, Praha



Rodina protokolů TCP/IP, verze 2.6

Část 5: Protokol IP et al.

Jiří Peterka, 2010

charakteristika protokolu IP

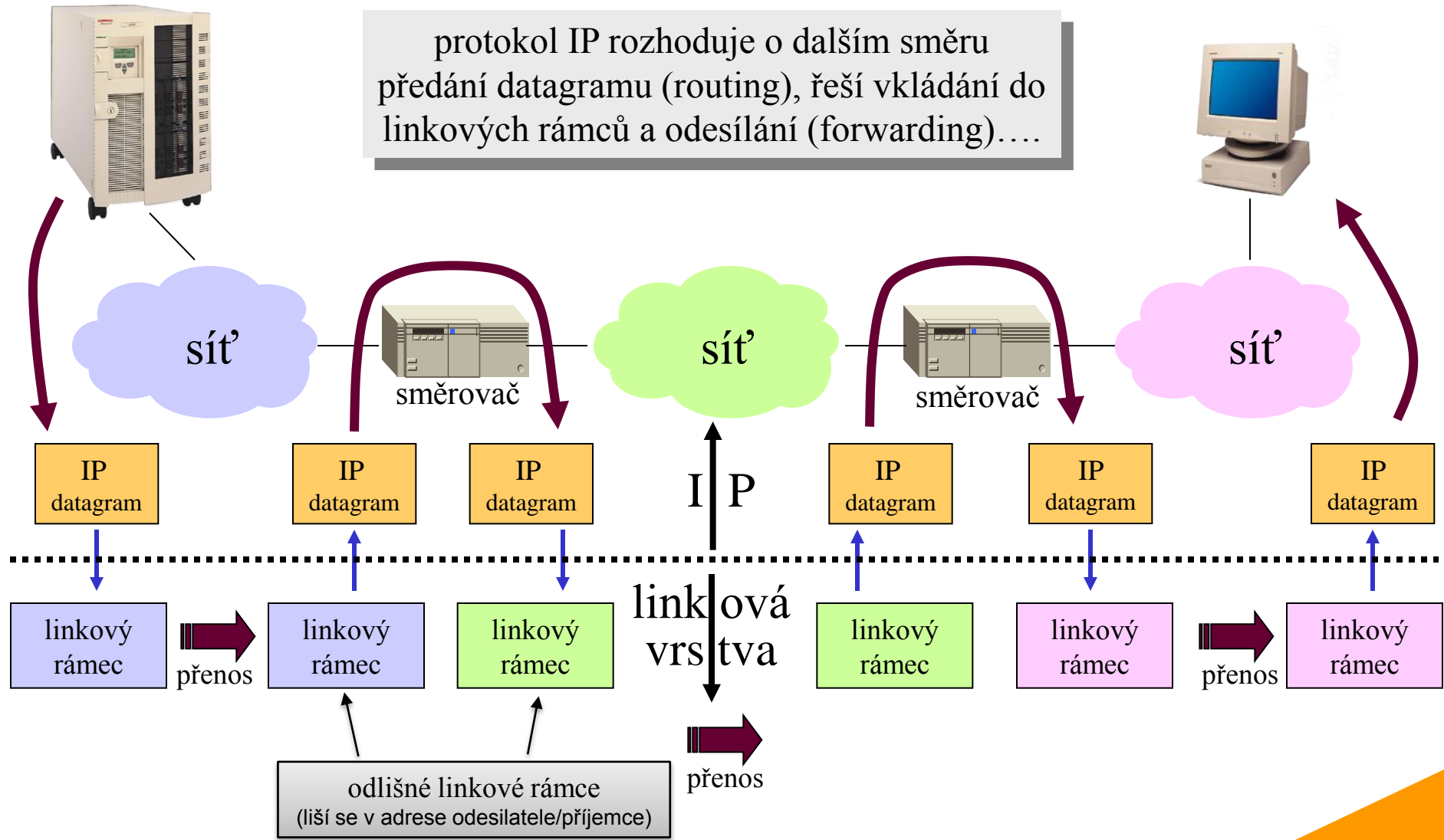
- jde o přenosový protokol síťové vrstvy
 - je to univerzální přenosový protokol
 - snaží se fungovat "nad vším", nad libovolnou přenosovou technologií
 - je jediným přenosovým protokolem TCP/IP
 - na síťové vrstvě
 - používá virtuální pakety
 - nemají ekvivalent v HW, musí se zpracovávat v SW
 - říká se jim **IP datagramy**
- zajišťuje:
 - směrování (volbu směru přenosu)
 - přenos (IP) datagramů
- je implementován
 - v hostitelských počítačích
 - ve směrovačích
- je (funguje jako):
 - nespolehlivý
 - nespojovaný
- dnes používaná verze má číslo 4
 - a v rámci ní 32-bitové IP adresy
 - existuje i verze 6
 - se 128-bitovými adresami a vylepšenými vlastnostmi

vlastnosti protokolu IP

- je univerzální, nabízí jednotné přenosové služby
 - nevyužívá specifika fyzických přenosových technologií
 - chce jen "společné minimum"
 - snaží se zakrýt odlišnosti
 - vytváří jednotné prostředí pro všechny aplikace
- pracuje s proměnnou velikostí paketu (datagramu)
 - velikost určuje odesílatel (aplikace)
 - problém: může docházet ke fragmentaci
- je zaměřen na jednoduchost, efektivnost a rychlost
- je nespojovaný
 - nečísluje přenášené pakety
 - negarantuje pořadí doručení
 - negarantuje dobu doručení
 -
- funguje jako nespolehlivý/"best-effort"
 - negarantuje doručení
 - negarantuje nepoškozenost dat
 - nepoužívá potvrzení
 - nepodporuje řízení toku
 - smí zahodit datagram když:
 - má chybný kontrolní součet
 - překročil svou životnost
 - hrozí zahlcení sítě
 -

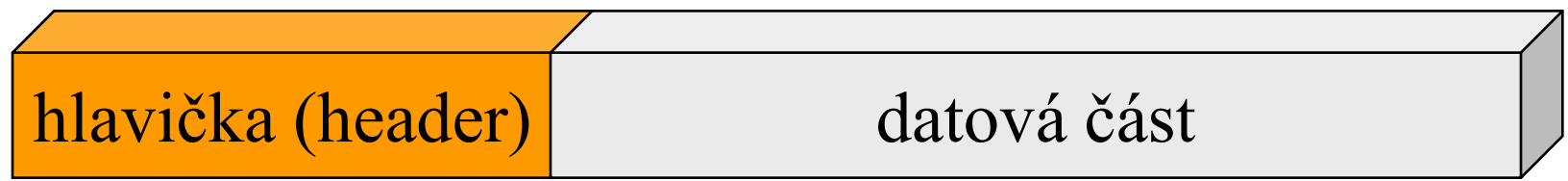
představa fungování protokolu IP

protokol IP rozhoduje o dalším směru předání datagramu (routing), řeší vkládání do linkových rámců a odesílání (forwarding)....



formát IP datagramu

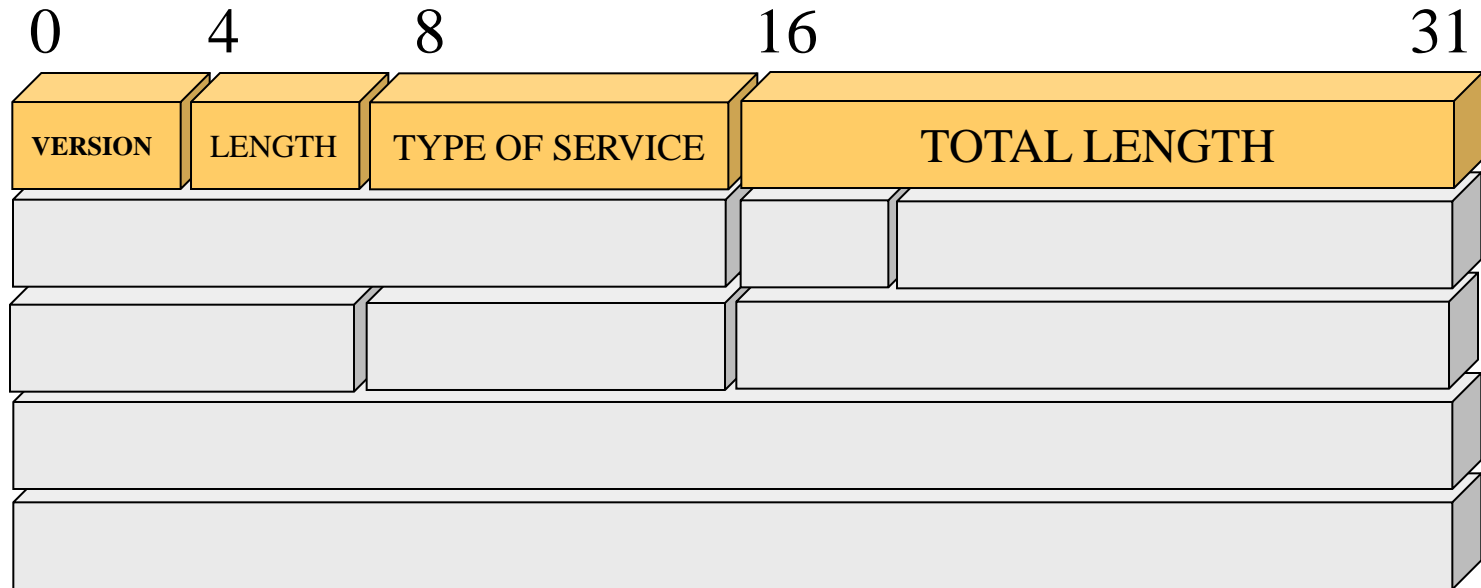
- velikost je proměnná
 - max. 64 K (65535 bytů)
 - minimální podporovaná velikost: 576 bytů
 - bez toho, aby docházelo k fragmentaci
 - odpovídá to 512 bytům užitečného "nákladu", ostatní je reže
 - včetně reže vyšších vrstev



HLEN (Header LENgth, 4 bity) velikost hlavičky je také proměnná (ale: typická velikost je 20 B)

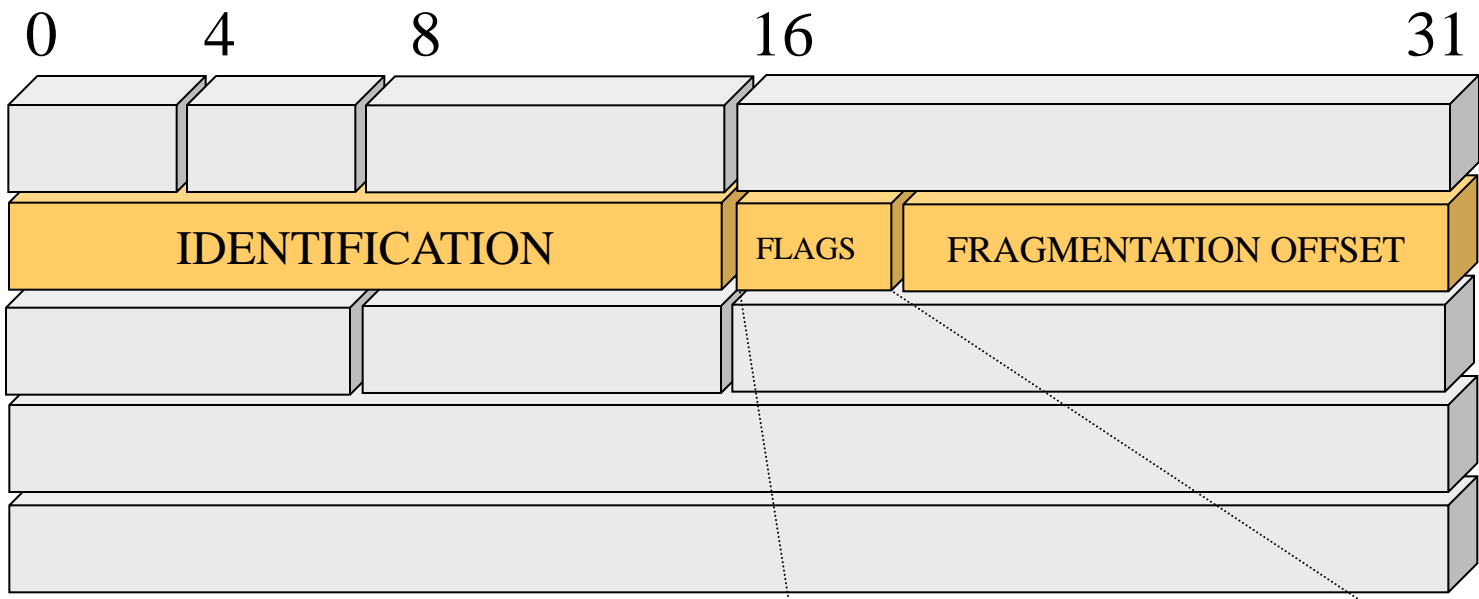
TOTAL LENGTH (16 bitů), max. 65535 bytů

formát hlavičky IP datagramu



- **VERSION:**
 - dnes = 4 (IPv4)
- **LENGTH:**
 - velikost hlavičky v jednotkách 32-bitů
 - při typické velikosti 20 bytů je LENGTH = 5
- **TYPE OF SERVICE**
 - dnes ignorováno
 - původně: mělo vyjadřovat požadavky na QoS
- **TOTAL LENGTH**
 - celková délka datagramu v bytech, včetně hlavičky!

formát hlavičky IP datagramu



- IDENTIFICATION

- slouží potřebám fragmentace
 - všechny fragmenty ze stejného celku mají v této položce stejnou hodnotu
 - podle toho se pozná že patří k sobě

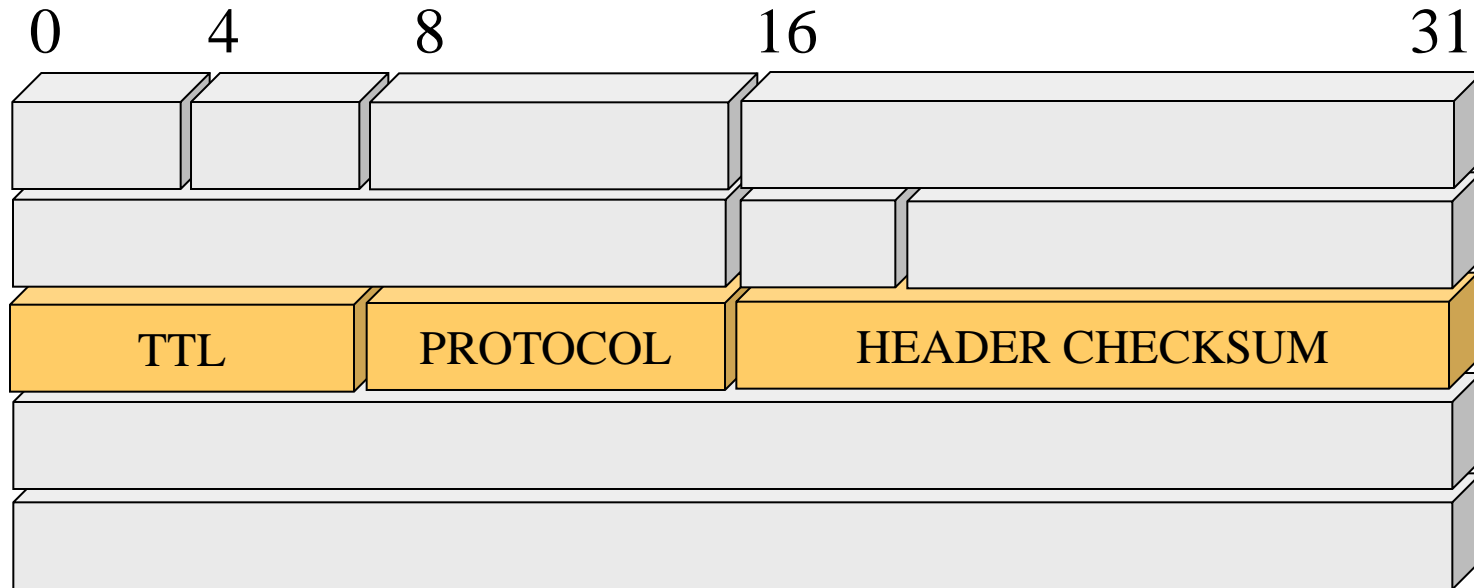


1=nefragmentuj 1 = jsou ještě další fragmenty
0 = poslední fragment

- FRAGMENTATION OFFSET

- offset fragmentu dat od začátku původního celku

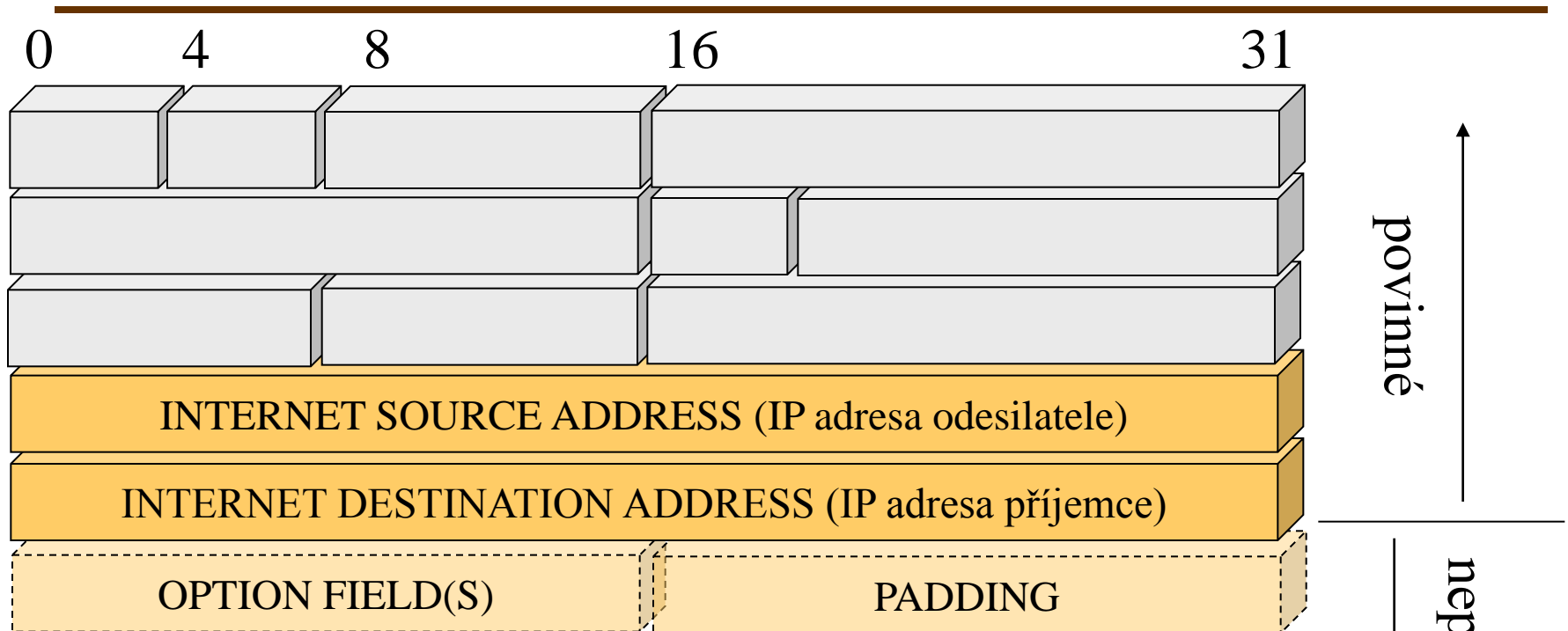
formát hlavičky IP datagramu



- TTL (Time To Live)
 - fakticky čítač průchodů přes směrovače
 - slouží k detekci zacyklení
- HEADER CHECKSUM
 - kontrolní součet hlavičky (!!)
 - počítaný jako 1-vý doplněk
 - při každém průchodu směrovačem se přepočítává
 - kvůli TTL !!!!
- PROTOCOL
 - udává typ užitečného nákladu
 - kterému protokolu patří:
 - 1 = ICMP
 - 4 = IP over IP (tunelování)
 - 6 = TCP
 - 17₁₀ = UDP
 -

představuje to velkou zátěž!!!

formát hlavičky IP datagramu



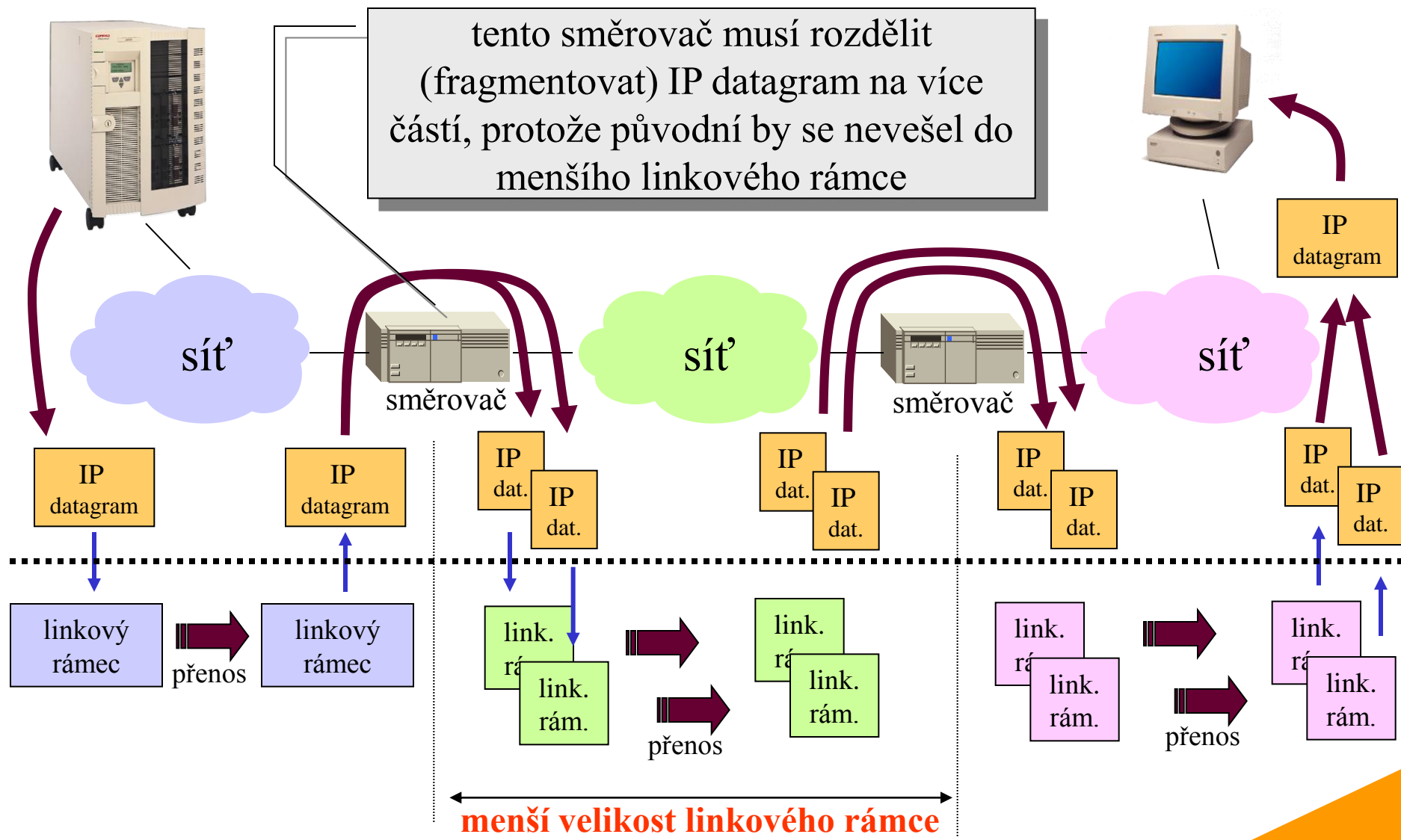
- **OPTION FIELD(S)**

- umožňuje rozšířit hlavičku o další funkce
 - např. source routing, záznam cesty, záznam času
- dnes se nepoužívá, směrovače obvykle zahazují datagramy s tímto rozšířením

- **PADDING**

- "vycpávka"
 - aby hlavička měla velikost která je násobkem 4 bytů

problém fragmentace

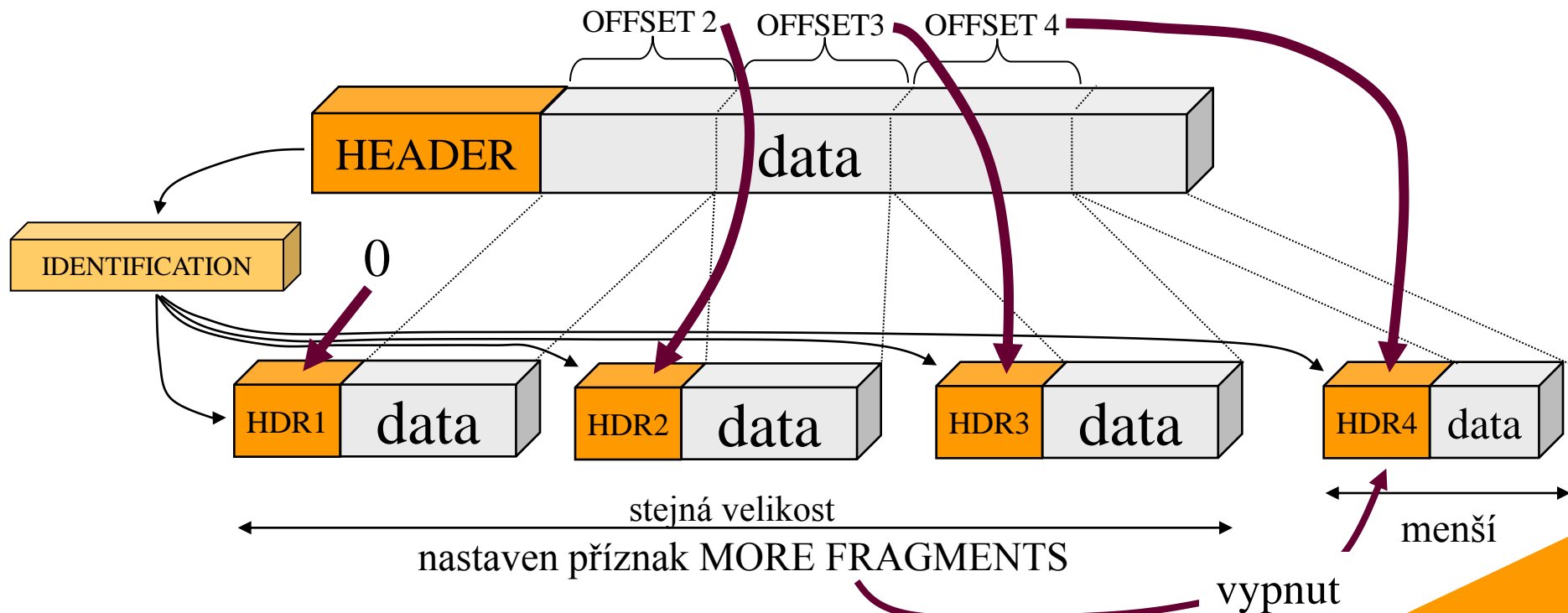


problém fragmentace

- příčina problému:
 - různé přenosové technologie pracují s různými velikostmi linkových rámců
 - velikost udává parametr MTU (Maximum Transfer Unit)
 - např.
 - 576: X.25
 - 1492: IEEE 802.3
 - 1500: Ethernet II
 - 1500, 2048, 4096: Token Ring
 - 4325, 2048: FDDI
 - 65515: HPPI (High Performance Parallel Interface)
- ten, kdo určuje velikost datového paketu, se může přizpůsobit známé velikosti MTU
- problém:
 - znalost MTU se týká jen místní sítě (segmentu), netýká se celé cesty !!!
 - díky nespojovanému charakteru IP protokolu (nenavazuje spojení) nelze fragmentaci vyloučit
 - i když bude respektováno "místní" MTU !!

řešení fragmentace

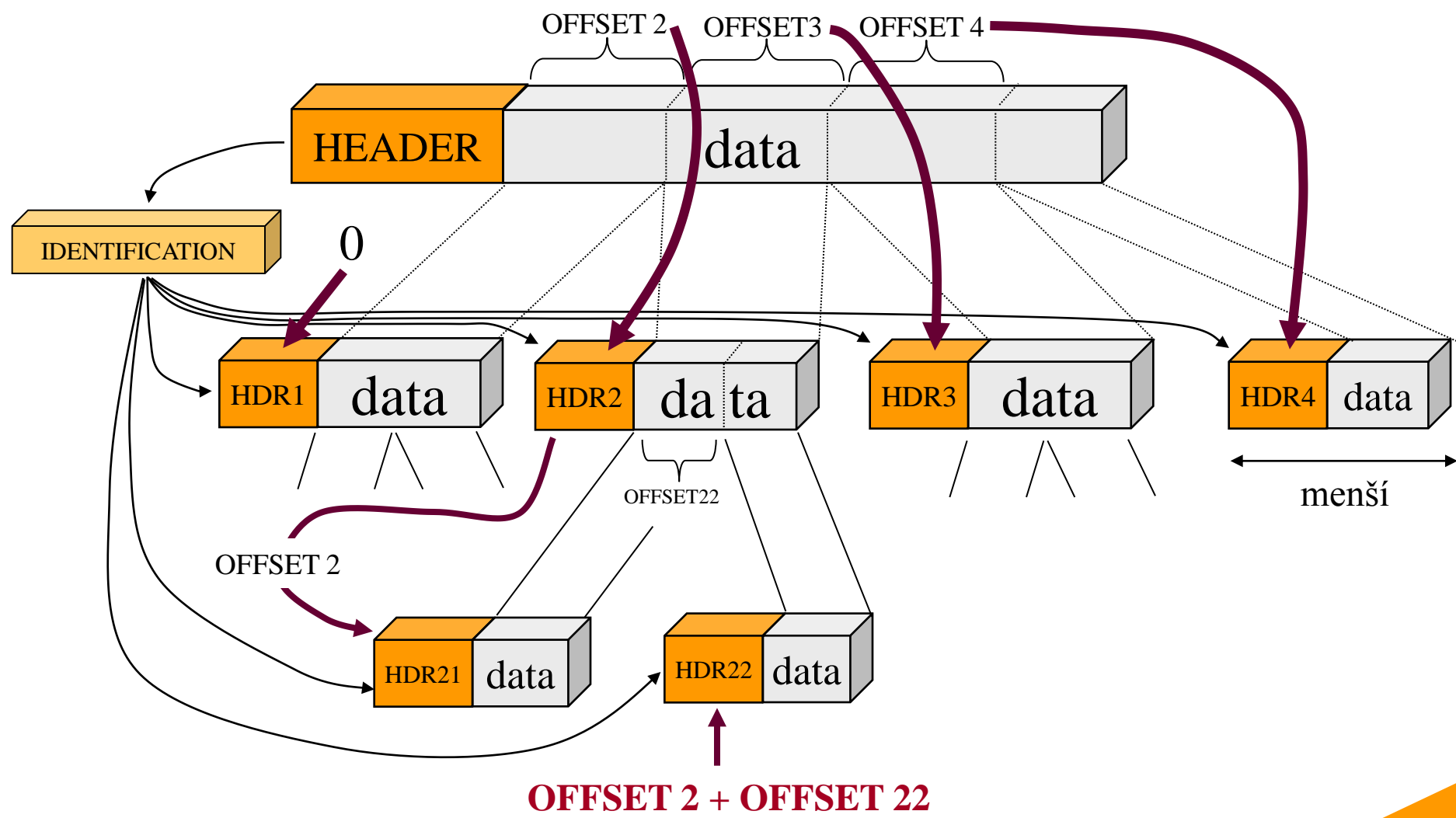
- směrovač, který zjistí že "následující síť" má příliš malé MTU, rozdělí IP datagram na několik menších IP datagramů (fragmentů)
 - každému dá stejnou hodnotu IDENTIFICATION jako má mateřský datagram
 - odesílatel generuje systematicky, inkrementací
 - každému fragmentu nastaví příznak MORE FRAGMENTS (kromě posledního)
 - každému segmentu nastaví položku OFFSET podle skutečnosti



řešení fragmentace

- na fragmenty rozděljuje ten, kdo zjistí že je to zapotřebí ...
- **.. zpětné sestavení zajišťuje až koncový příjemce !!!!**
 - nikoli první směrovač, který by tak mohl učinit
- zpětné sestavování se musí vyrovnat s možností ztráty některého fragmentu
 - není garantováno že všechny fragmenty dorazí k cíli
 - původní datagram je sestaven pouze pokud dorazí všechny fragmenty
 - v opačném případě je vše zahozeno
 - problém i s časem – příjemce sleduje timeout
- nelze zabránit opakovaným fragmentacím
 - řeší se další fragmentací jednotlivých fragmentů
 - IP protokol to dovoluje, nerozlišuje mezi "úrovněmi" fragmentů
 - "fragment fragmentu" je stále fragmentem
- je garantována minimální velikost IP datagramu (576 B) která by neměla být nikdy fragmentována
 - odpovídá to "užitečnému nákladu" nejméně 512 B
- pomocí příznaku DON'T FRAGMENT lze požadovat aby datagram nebyl fragmentován
 - pak musí být zahozen

řešení fragmentace²

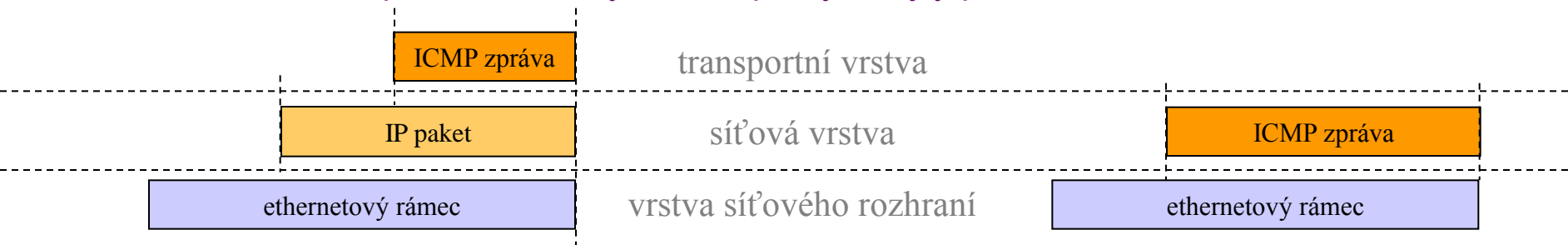


ICMP (Internet Control Message Protocol)

- protokol IP není "bezcitný"
 - nezahazuje datagramy bezdůvodně
 - má právo zahodit datagram jen při nestandardních situacích
 - zacyklení, chybný kontrolní součet hlavičky, přetížení, když nelze fragmentovat, ...
 - když něco zahodí, nemusí se starat o nápravu
 - snaží se ale informovat o tom, že se něco stalo
 - koho? jak?
- pro potřeby informování o nestandardních situacích byl vyvinut protokol ICMP
 - kromě chybného kontrolního součtu hlavičky, pak nelze důvěřovat údajům o odesilateli a dalším ...
- protokol ICMP je integrální součástí protokolu IP (?)
 - musí být povinně implementován spolu s IP
 - je vzájemně provázán s IP
 - příjemcem ICMP zpráv je IP protokol odesilatele
 - ICMP pakety cestují sítí vložené do IP datagramů
 - ztráty datagramů obsahujících ICMP pakety nejsou oznamovány (hrozilo by zacyklení)
- přehled situací/informací, které ICMP hlásí:
 - Source Quench (varování před hrozícím zahlcením)
 - Time exceeded
 - Destination unreachable
 - Redirect
 - Parametr problem
 - Echo request/reply
 - Address mask request/reply (uzel si řekne o síťovou masku)
 - Router solicitation/advertisement

dilema: jak přenášet ICMP zprávy?

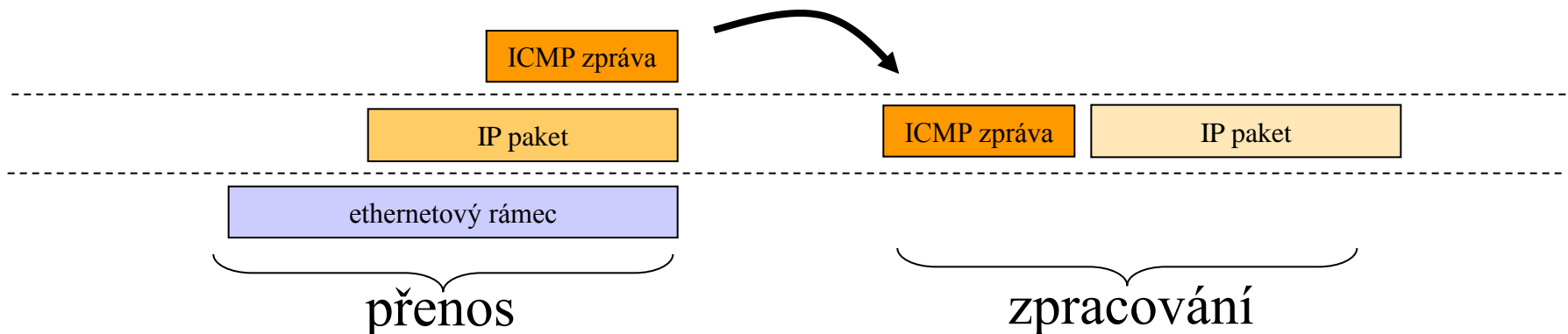
- zprávy protokolu ICMP slouží potřebám síťové vrstvy
 - generují je nejčastěji směrovače
 - a směrovače mohou být i jejich příjemci
 - ale směrovače fungují (nejvýše na) síťové vrstvě
 - na transportní vrstvě by ICMP zprávy nebyly pro směrovače viditelné



- varianta 1:
 - vkládat je do IP paketů
 - pak snadno "projdou" všude tam, kde IP pakety
 - ale pak ICMP zprávy patří na transportní vrstvu !!!
 - podle pravidel vrstevných modelů
 - a tudíž nejsou pro směrovače viditelné
- varianta 2:
 - vkládat je přímo do linkových rámců
 - pak "patří" na síťovou vrstvu a pro směrovače jsou viditelné
 - ale pak nemusí "projít" všude tam, kde "prochází" protokol IP
 - všechny směrovače by nutně musely být multiprotokolové a podporovat jak směrování IP, tak i směrování ICMP

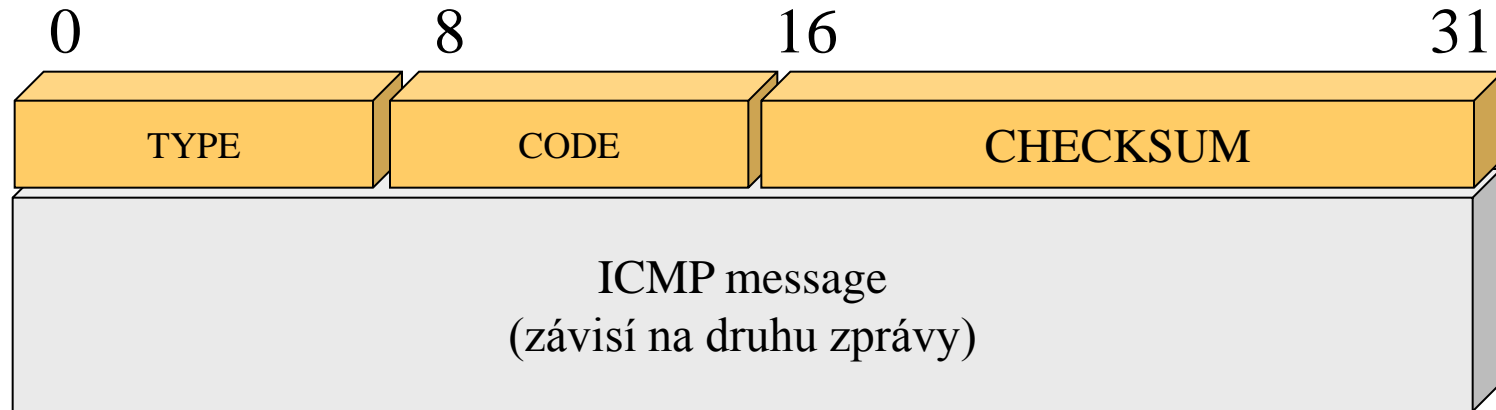
ICMP dilema

- ani jedna z variant 1 a 2 není správná
- praktický kompromis:
 - ICMP zprávy se vkládají do IP paketů
 - aby "prošly všude tam, kde IP" a nevyžadovaly multiprotokolové směrovače
 - a současně se považují za součást síťové vrstvy
 - aby ICMP bylo dostupné i pro směrovače, fungující na síťové vrstvě



- je to výjimka z pravidla o fungování vrstevnatých modelů
 - porušení těchto pravidel, motivované praktickými důvody

formát ICMP zprávy/paketu

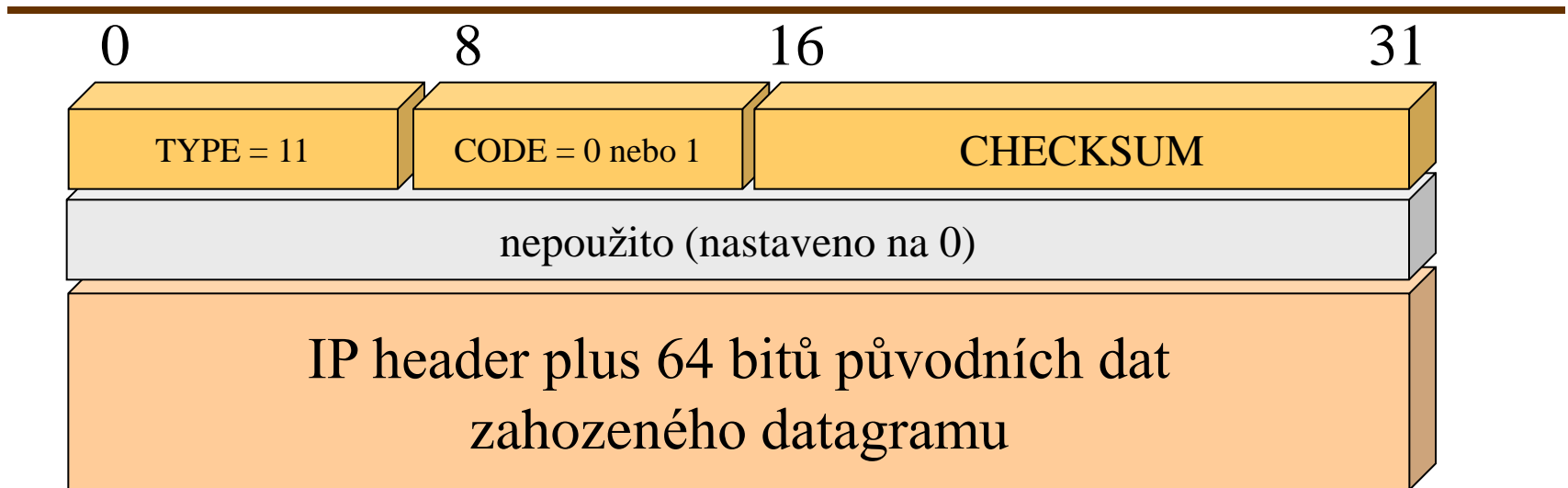


- **TYPE:** určuje druh ICMP zprávy/paketu
 - 0 Echo reply (odpověď na požadavek, při PINGu)
 - 3 Destination unreachable (dále se větví dle pole CODE)
 - 4 Source Quench
 - 5 Redirect
 - 6 Alternate host address
 - 8 Echo Request
 - 9 Router Advertisement
 - 10 Router Solicitation
 - 11 Time Exceeded (TTL kleslo k 0)
 - 12 Parameter Problem
 - 13 Timestamp Request
 - 14 Timestamp Reply
 - 15 Information Request
 - 16 Information Reply
 - 17 Address Mask Request
 - 18 Address Mask Reply
 -
 - 30 Traceroute
 -
- **CODE:** upřesňuje význam položky TYPE

ICMP Time Exceeded

- při směrování IP datagramu se může stát, že dojde k zacyklení
- rozpoznává se pomocí položky TTL (Time To Live) v hlavičce datagramu
 - původní význam položky: bude zde čas v sekundách
 - ukázalo se jako nereálné
 - dnešní význam: počet "přeskoků" (hop count)
 - při každém průchodu směrovačem je hodnota položky TTL dekrementována
 - kvůli tomu musí být přepočítáván kontrolní součet !!!
- když IP protokol zjistí, že položka TTL dekrementuje k nule, má právo datagram zahodit
- ale má povinnost o tom informovat odesilatele zahozeného datagramu
 - pomocí ICMP zprávy typu "TIME EXCEEDED"
 - s nastaveném CODE=0
- jiná situace:
 - při zpětném sestavování fragmentů vyprší timeout (vynuluje se nastavený timer)
 - IP reaguje zasláním ICMP zprávy TIME EXCEEDED s nastavením CODE=1

formát "ICMP TIME EXCEEDED"

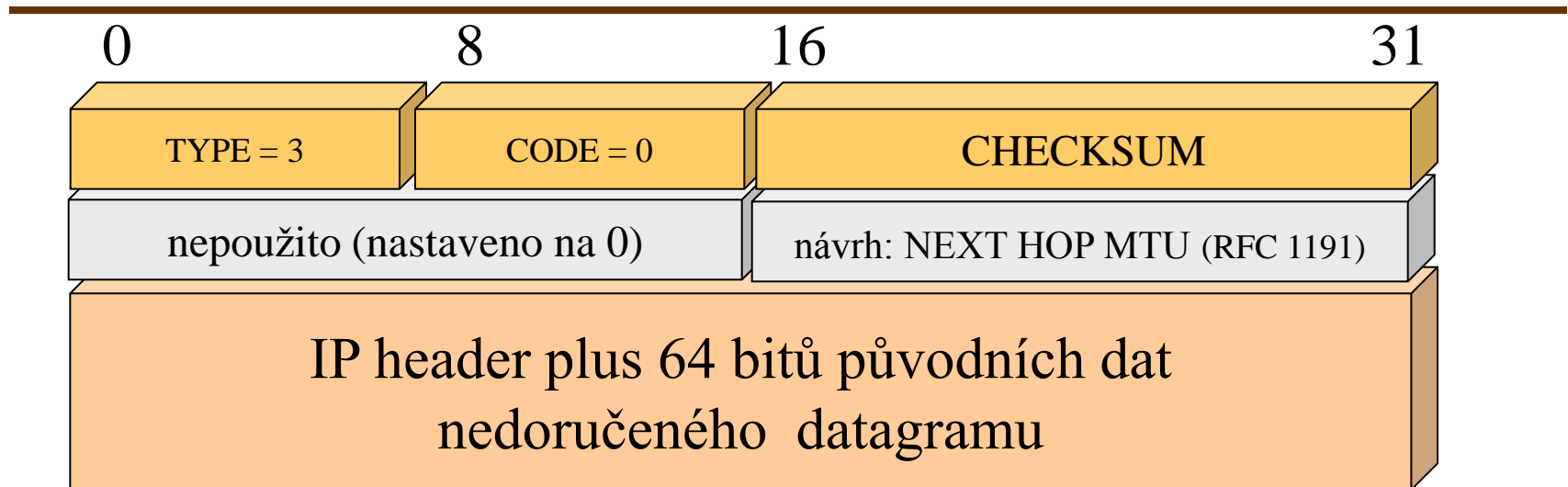


- do ICMP TIME EXCEEDED je vložen začátek zahozeného paketu, aby se příjemce mohl dozvědět o co se jedná (co je příčinou chyby)
- situace, na kterou IP protokol reaguje zprávou ICMP TIME EXCEEDED, je někdy generována záměrně
 - např. pro potřeby utility traceroute

utilita Traceroute

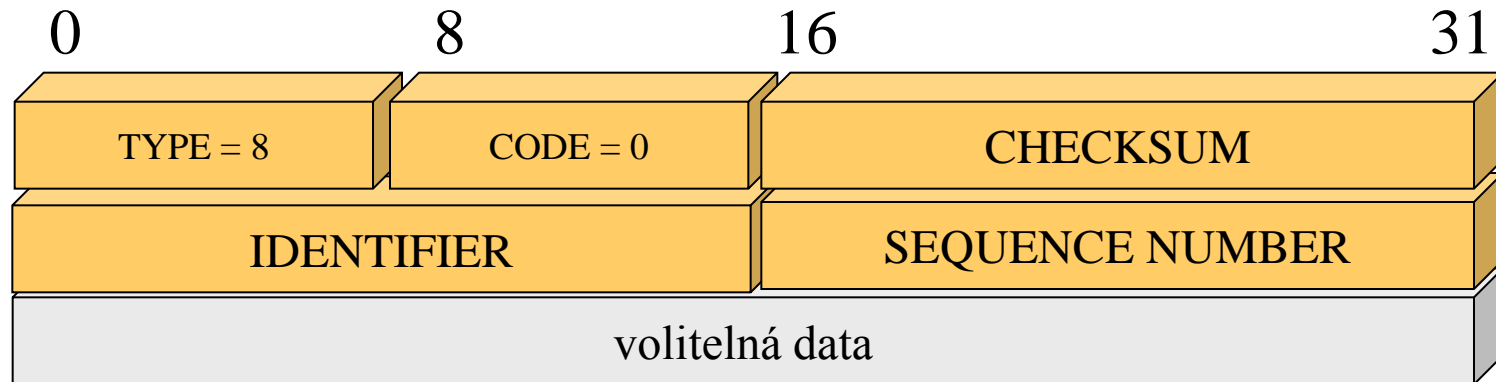
- #1 gw.peterka.cz (.....): TTL Exceeded, ttl=128, 2 ms
 - #2 Praha14.core.Czech.Net (194.213.225.6): TTL Exceeded, ttl=254, 153 ms
 - #3 Praha9.core.Czech.Net (194.213.225.9): TTL Exceeded, ttl=253, 144 ms
 - #4 Praha0.core.Czech.Net (194.213.225.1): TTL Exceeded, ttl=253, 144 ms
 - #5 nix.ten.cz (194.50.100.191): TTL Exceeded, ttl=252, 329 ms
 - #6 porta-ten.pasnet.cz (195.113.69.105): TTL Exceeded, ttl=251, 149 ms
 - #7 atmms-1.pasnet.cz (195.113.68.132): TTL Exceeded, ttl=250, 1506 ms
 - #8 ksi.ms.mff.cuni.cz (195.113.19.213): Echo Reply, ttl=122, 157 ms
-
- testující uzel pošle datagram cílovému stroji, nastaví TTL=1
 - fakticky posílá ICMP Traceroute (CODE=30) nebo UDP datagram na neexistující port
 - první příjemce po cestě dekrementuje TTL na 0 a vygeneruje ICMP TIME EXCEEDED (z toho se pozná, kdo je 1. uzel po cestě)
 - testující opakuje s inkrementovaným TTL
 - poslední krok se dělá pomocí UDP datagramu na neexistující port

ICMP Destination Unreachable




- obecnější hlášení než TIME EXCEEDED
 - pokrývá další situace, kdy byl IP datagram zahozen, resp. nemohl být doručen, mj.:
 - nedostupná síť (CODE=0), nedostupný uzel (CODE=1),
 - neexistující adresy (CODE=2), porty (CODE=3),
 - překročení MTU při nastaveném příznaku že se nemá fragmentovat (CODE=4)
 - chyby routerů, chyby SW
 - nesprávné cesty (cykly, neexistující cesty apod.)
 - výpadky uzlů
 - zakázaný přístup pro určitý druh paketů / služeb

ICMP Echo Request



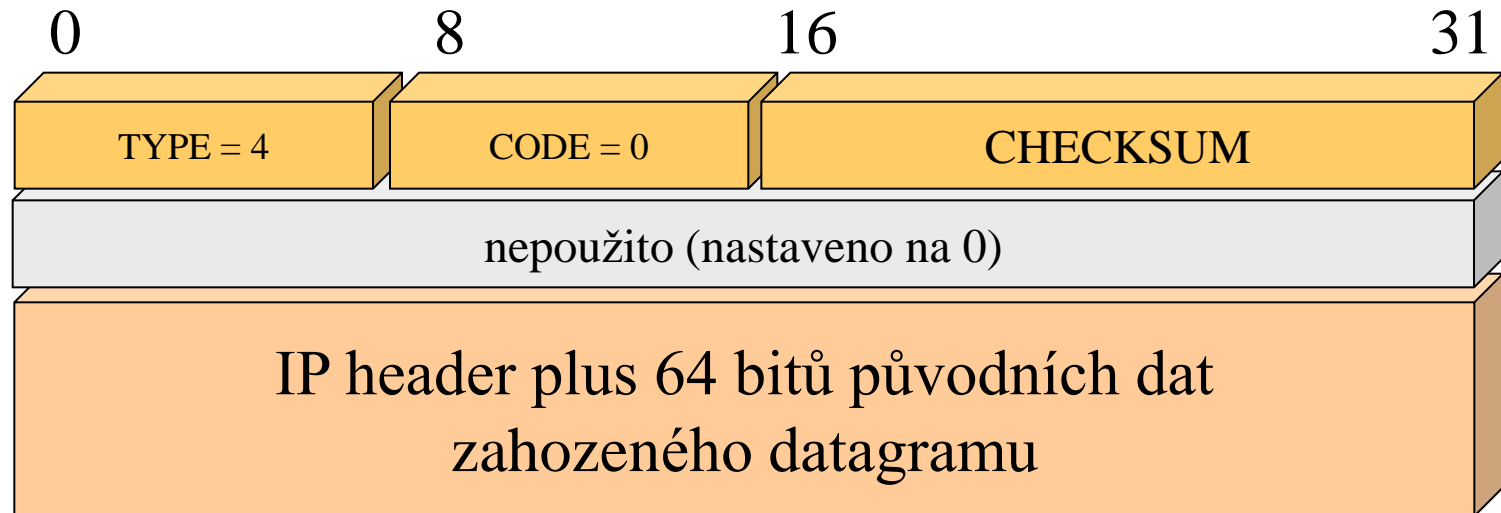
- slouží k:
 - testu dostupnosti (konektivity)
 - změření doby přenosu (RTT, Round Time Trip)
 - zjištění zda dochází k fragmentaci
 - určení počtu "hopů" (přeskoků) k cíli (pozná se podle TTL ve výpisu odpovědi)
- zprávu ICMP Echo Request (TYPE = 8) může poslat kterýkoli host nebo router.
- dotázaný pošle ICMP Echo Reply (TYPE=0)
 - Odpověď generuje přímo ICMP software, jinak ale nemění datový obsah žádosti (volitelná data)
- volbou velikosti paketu (volitelných dat) lze testovat fragmentaci

PING (Packet InterNet Groper)

- Ping - Monday, April 03, 2000 10:44:51
- Address: ksi.ms.mff.cuni.cz, Number of Packets: 3, Packet size: 1400
- Don't Fragment: Yes  test MTU
- #1 ksi.ms.mff.cuni.cz (195.113.19.213): Echo Reply, ttl=122, 805 ms
- #2 ksi.ms.mff.cuni.cz (195.113.19.213): Echo Reply, ttl=122, 811 ms
- #3 ksi.ms.mff.cuni.cz (195.113.19.213): Echo Reply, ttl=122, 839 ms
- Statistics: Out 3, in 3, loss 0%, times (min/avg/max) 805/818/839 ms

"vzdálenost", měřená v počtu přeskoků odpovědi (ECHO REPLY)
jde o doplněk do výchozí hodnoty, kterou různé implementace nastavují různě!!
Zde bylo výchozí hodnotou 128, tj. vzdálenost je $6+2 = 8$!!

ICMP SOURCE QUENCH



- jde o hlášení od směrovače, že u něj dochází k přetížení (hrozícímu zahlcení)
- možné důvody:
 - jeden zdroj posílá data směrovači rychleji než je on dokáže zpracovat
 - tím že souběh požadavků přicházejících na směrovač překračuje jeho kapacitu
 -

ICMP SOURCE QUENCH

- jednoduchý směrovač:
 - posílá ICMP SOURCE QUENCH po každém paketu, který dostane když je přetížen
- dokonalejší směrovač:
 - monitoruje zdroje dat a jejich toky, a posílá ICMP SOURCE QUENCH jen těm, kteří mají podstatný vliv na jeho přetížení
- ještě dokonalejší směrovač:
 - může generovat ICMP SOURCE QUENCH ještě dříve, než dojde k přetížení, když se snaží o predikci a předcházení tomuto jevu
- zpráva SOURCE QUENCH je pouze informace o tom, že hrozí/dochází k přetížení směrovače.
 - nespecifikuje proto konkrétní požadovanou akci
 - ta je ponechána na vůli toho, kdo SOURCE QUENCH dostane
 - také tento uzel se může chovat různě inteligentně, a svou akci volit podle toho, že dostal prvních 64 bitů dat, která způsobila přetížení
 - neexistuje inverzní zpráva k SOURCE QUENCH
 - nelze oznámit konec přetížení, je nutné to dělat nějak postupně a testovat průchodnost, například postupně zvyšovat objem dat)

Address Resolution

symbolická jména

překlad: DNS

IP adresy

překlad:
ARP, RARP

linkové adresy
(HW adresy)

- připomenutí:
 - IP adresy jsou abstraktní (virtuální), nemají "fyzickou" analogii v linkových adresách
 - hardwarových adresách
- musí existovat převodní mechanismy
 - AR, Address Resolution
 - převod z IP na HW adresu
 - RAR, Reverse Address Resolution
 - opačně, z HW na IP adresu

možné přístupy k překladu adres

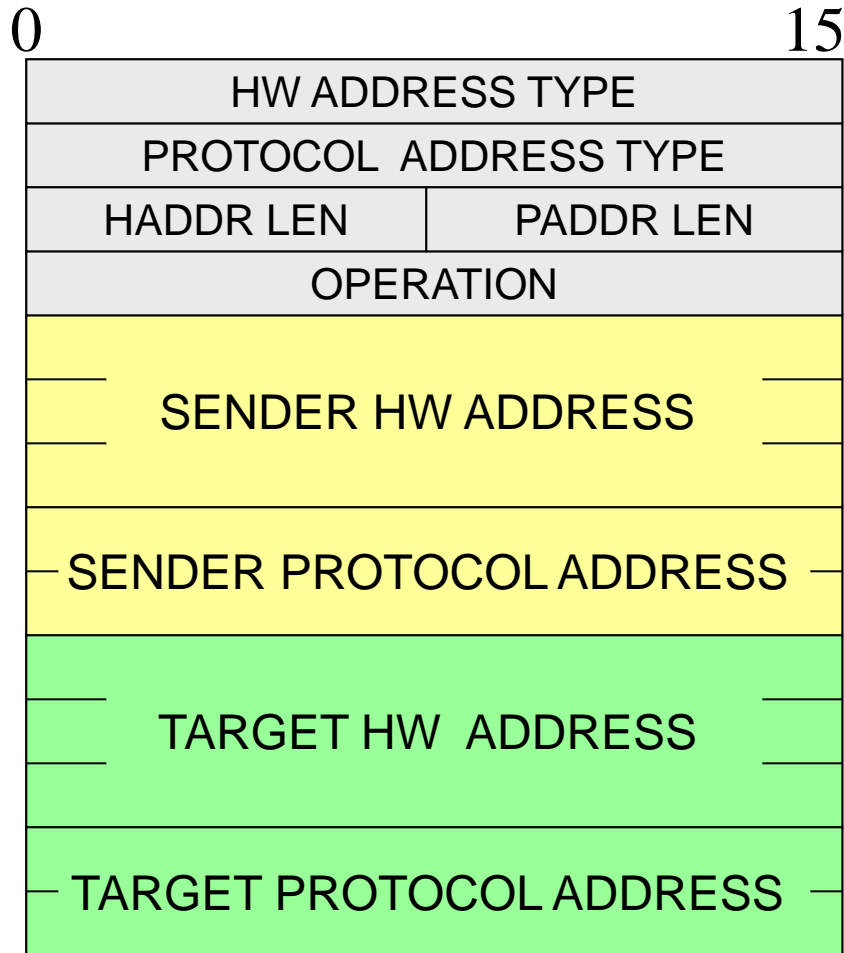
- tabulkový převod
 - k dispozici je tabulka s převodními informacemi
 - malou tabulku lze prohledávat systematicky, u větší je třeba použít vhodnou optimalizaci (hašování apod.)
 - výhoda: je to rychlé
 - nevýhoda: je to problematické tam, kde dochází ke změnám
- přímý výpočet
 - je k dispozici funkce, která z IP adresy vypočítá HW adresu
 - např. tak že vezme nejnižší byte IP adresy
 - lze použít jen tam, kde lze nastavit HW adresu
- pomocí dotazů&odpovědí
 - ten, kdo potřebuje udělat převod, pošle dotaz tomu kdo to dokáže
 - centralizované řešení:
 - existuje "převodní server", který zná všechny adresy a jejich vztahy (analogie DNS)
 - distribuované řešení
 - na převodní dotazy odpovídá ten, komu adresa patří
 - ten by měl znát svou HW adresu i IP adresu
 - lze použít i pro oba směry převodu

nejčastější řešení, vyžaduje ale broadcast

ARP, Address Resolution Protocol

- jde o protokol, určený pro "překlady adres" metodou dotaz&odpověď
 - v distribuované variantě
 - je použitelný pouze tam, kde je k dispozici všesměrové vysílání (broadcasting)
- princip fungování:
 - pošle dotaz všem v dané síti (pomocí broadcastu)
 - ten koho se dotaz týká odpoví
- definuje:
 - formát dotazu a odpovědi
 - způsob přenosu / šíření dotazů a odpovědí
- dotazy jsou vkládány přímo do linkových rámců
 - do Ethernetových rámců, v prostředí Ethernetu, EtherType=0x806
- rozesílány jsou všem uzlům
 - v Ethernetu pomocí broadcastu (v adrese jsou samé 1)
- dotaz je typu: **"kdo má takovouto IP adresu?"**
 - odpovídá pouze ten uzel, který rozpozná svou IP adresu,
 - do odpovědi vloží svou HW adresu
- odpověď je zasílána cíleně, nikoli jako broadcast

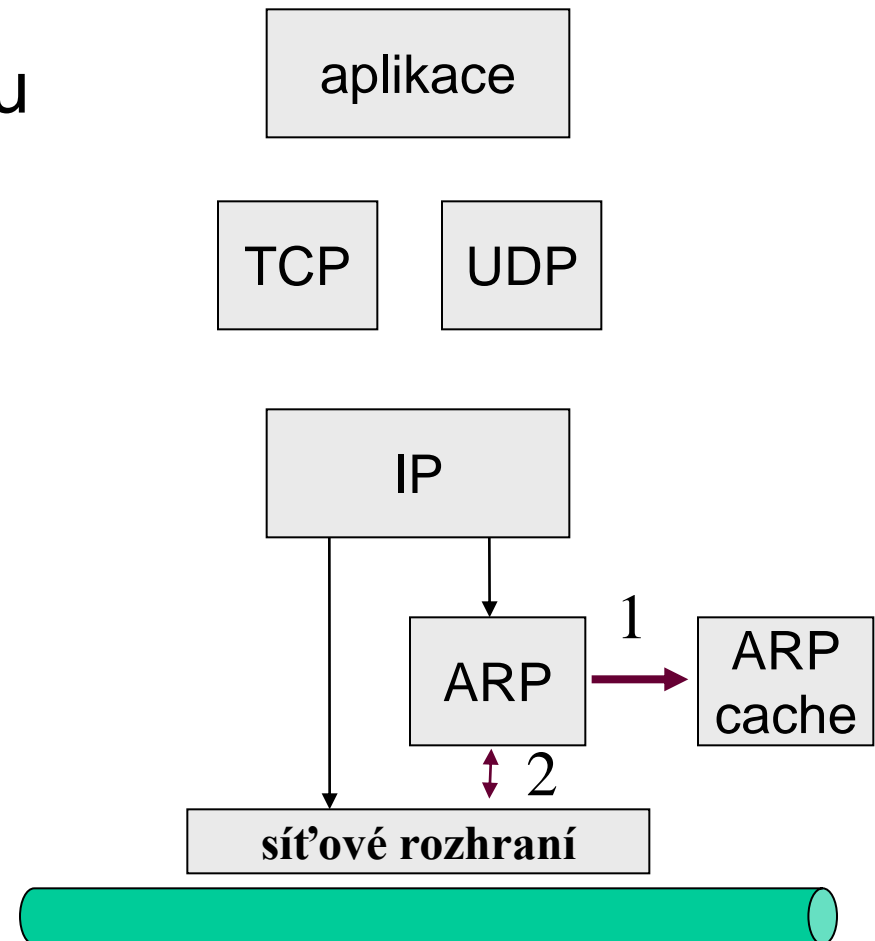
formát ARP zprávy



- HW ADDRESS TYPE:
 - 1 když jde o Ethernet
- PROTOCOL ADDRESS TYPE
 - 0x800 když jde o IP
- HADDR LEN (délka HW adresy v oktetech)
- PADDR LEN (délka protokolové (IP) adresy)
- OPERATION
 - 1 pro dotaz, 2 pro odpověď
- SENDER HW ADDRESS, SENDER PROTOCOL ADDRESS
 - adresy toho kdo je odesilatelem dotazu nebo odpovědi
- TARGET HW ADDRESS, TARGET PROTOCOL ADDRESS
 - když se posílá zpráva, je TARGET PROTOCOL ADDRESS vyplněna nulami
 - když se posílá odpověď, je podstatná informace v SENDER HW ADDRESS.

ARP caching

- kvůli celkové efektivnosti překladového mechanismu jsou výsledky ARP dotazů cacheovány (na cca 20 minut)
- obecný postup:
 - když je potřeba provést překlad, ARP se nejprve podívá do cache.
 - teprve když tam nenajde potřebnou vazbu (binding), pošle dotaz



zpracování ARP dotazu

- každý uzel který přijme broadcast:
 - extrahuje "binding" odesilatele dotazu (vazbu mezi jeho IP a HW adresou).
 - pokud jej už má ve své cache, osvěží ji (znova to vloží do své cache). Tím se mj. ošetří i změna adresy odesilatele
 - příjemce se podívá na pole OPERATION, zda jde o dotaz nebo odpověď.
 - pokud jde o dotaz, zjistí příjemce zda se ho týká
 - porovná pole TARGET PROTOCOL ADDRESS se svou adresou.
- pokud jde o odpověď, už nejde o broadcast
 - odpověď by měla být cílená, tj příjemce dříve vyslal dotaz a nyní čeká na odpověď
 - odpověď využije.
- pokud se dotaz týká daného příjemce, je povinen odpovědět
- sestaví ARP odpověď:
 - přehodí význam SENDER a TARGET, vyplní adresy druhé strany (byť je to zbytečné)
 - doplní svou HW adresu (do pole SENDER HW ADDRESS)
 - nastaví pole OPERATION na odpověď (2)
 - pošle cíleně tazateli
- poté, co poslal odpověď, dotázaný si sám zanesse binding (překládovou informaci o tazateli) do své cache paměti
 - je to optimalizace kvůli tomu, že za chvíli mu nejspíše bude něco posílat, a bude potřebovat jeho adresu

Proxy ARP

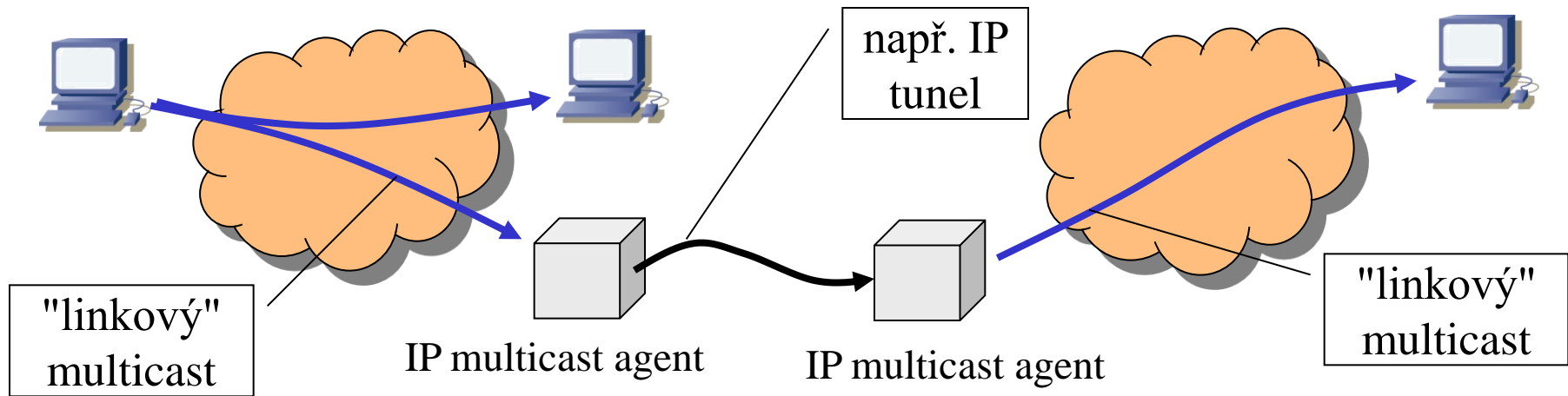
- zvláštní varianta ARP, kdy jeden uzel (směrovač) odpoví na ARP dotaz týkající se uzlu který je schován na něm
 - typicky to je uzel na pomalém jednobodovém spoji
 - vlastně tak směrovač skrývá sám sebe (resp. dává do jedné sítě i to co je za ním.).
- používá se to například tam, kde jeden uzel chce vystupovat jménem druhého
 - např. při mobilním IP bude plnit roli místního agenta

IP Multicast

- multicast:
 - možnost odeslat paket, který bude doručen více příjemcům (současně)
 - linkový multicast:
 - doručuje se uzlům v "přímém dosahu"
 - to je relativně snadné, s podporou v linkových technologiích
 - IP multicast
 - doručuje se uzlům, které mohou být "kdekoli" v dosahu IP protokolu
 - komplikované
- IP multicast adresy:
 - jsou "skupinové"
 - uzly se dynamicky přidávají/odstraňují ze skupiny
 - skupiny jsou:
 - permanentní: IP adresa je jim přidělena administrativně a trvale
 - a je "well known"
 - např. **224.0.0.2**: všechny směrovače v podsíti
 - "dočasné" (transient): vznikají a zanikají na základě potřeby
 - členství uzlů ve skupině je vždy dynamické

další dělení: veřejné vs. privátní

IP multicast – princip fungování



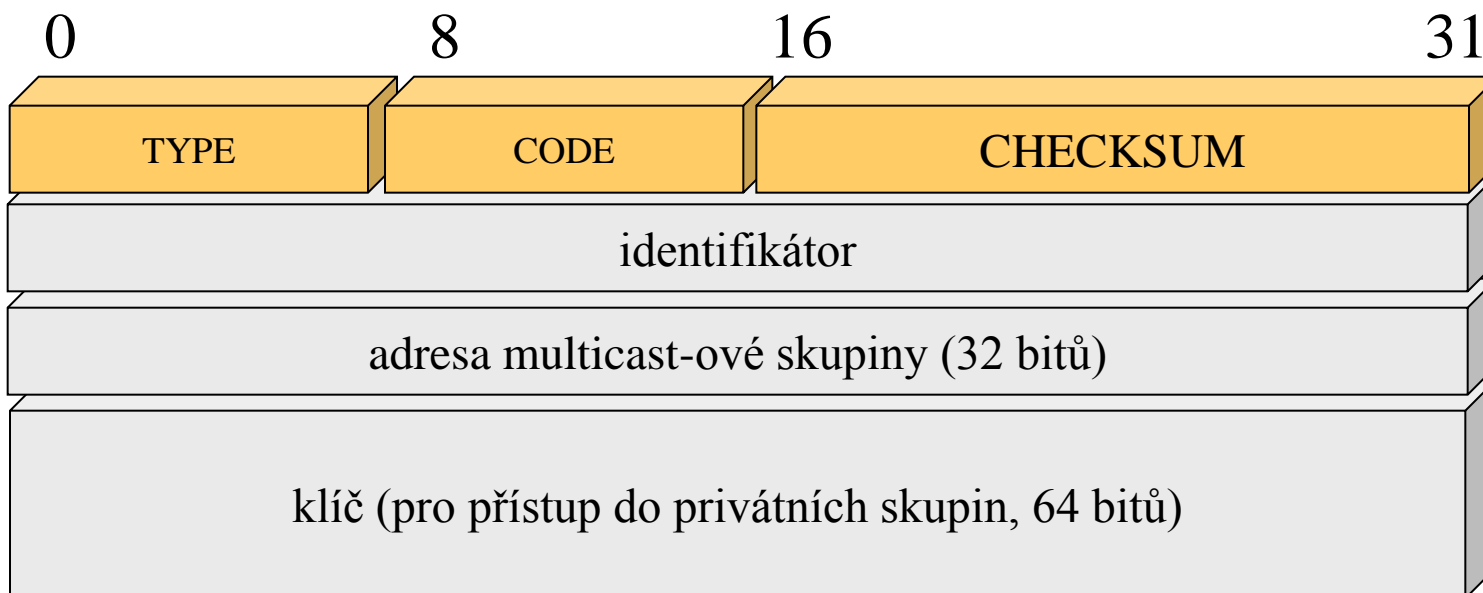
- princip řešení IP Multicast-u:
 - odesílatel "odesílá" paket na IP multicast adresu
 - ty uzly, které jsou "místní" (v dané síti), přijímají paket přímo
 - prostřednictvím multicastu na linkové vrstvě (linkového multicastu)
 - pokud je členem příslušné multicastové skupiny i nějaký uzel mimo danou síť, musí být v této síti "IP multicast agent"
 - ten přijme multicastový paket a zajistí jeho přenos do cílové sítě
 - v cílové síti je další agent, který zajistí místní rozeslání, prostřednictvím linkového multicastu

IGMP (Internet Group Management Protocol)

- slouží k řízení multicastových skupin
 - ke zřizování/rušení dočasných skupin
 - k přidávání/odebírání uzlů ze skupin
 -
- pakety IGMP se vkládají do IP paketů
 - obdobně jako u ICMP

- Type:
 - 1: žádost o vytvoření (dočasné) skupiny
 - 2: odpověď na žádost o vytvoření skupiny
 - 3: žádost o přidání uzlu do skupiny
 - 4: odpověď
 - 5: žádost o vyjmutí uzlu ze skupiny
 - 6: odpověď ...
 - 7: potvrzení vzniku (dočasné) skupiny
 - 8: odpověď ...

- Code:
 - 0: veřejná skupina
 - 1: privátní skupina



IP Multicast v praxi

- IP multicast je "definován" jako součást IPv4 již od začátku
 - pamatuje na něj i původní rozdělení 32-bitového prostoru IP adres
- praktická podpora je ale malá
 - důvody jsou různé:
 - malá poptávka po službách, které multicast vyžadují
 - chybějící podpora multicastu v koncových zařízeních
- teprve v poslední době se situace mění
 - poptávka roste
 - podpora v koncových zařízeních je lepší
 - v IPv6 je na multicast kladen velký důraz
- projekt MBONE (Multicast Backbone)
 - akademický projekt, cca od roku 1992
 - snaží se spojit "ostrůvky podporující IP multicast" přes veřejný Internet
 - pomocí "IP tunelu" skrze ty části Internetu, které nepodporují multicast
 - využívalo se pro videokonference
 - jednání IETF, konference NASA, vysílání internetových rádií, ...

IPv6

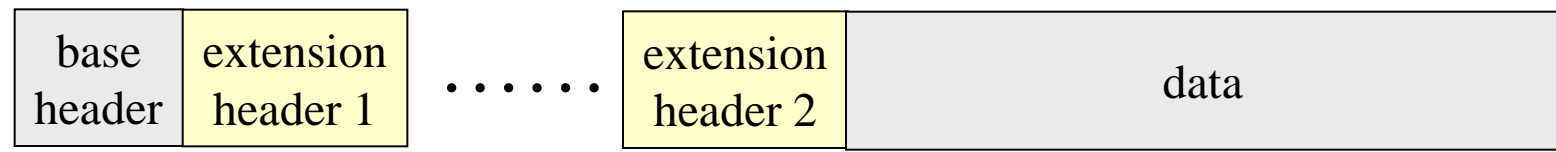
- proč?
 - jako zásadní řešení problému s nedostatkem 32-bitových IP adres
 - $2^{32} = 4\,294\,967\,296$
 - odhad reálně použitelných adres: jen cca 2 mld.
- důvody:
 - roste počet uzlů, připojených k Internetu
 - xDSL, kabel, satelit, Wi-Fi,
 - mobilní zařízení (GPRS telefony, PDA, UMTS/3G....)
 - inteligentní zařízení (ledničky, pračky, mikrovlny,
 - nevyrovnanost přidělů stávajících 32-bitových IP adres
 - Japonsko, Čína (Asie obecně) – dostali málo
 - dočasná řešení nevystačí navždy
 - privátní IP adresy, NAT, CIDR
- očekávání (IPv6 Forum):
 - již dnes obsahuje každá domácnost na 250 zařízení, která by mohla být někdy v budoucnu připojena k Internetu
 - v roce 2004 bude na světě 1,3 mld. mobilních telefonů, 750 mil. aut a desítky milionů dalších zařízení, která by mohla být připojena
- řešení:
 - stávajícímu protokolu IP (verze 4) nelze vnutit větší adresy
 - je (bylo) nutné vyvinout zcela nový protokol IP verze 6

hlavní změny v IPv6

- zvětšení IP adres
 - z 32 bitů na 128 bitů
 - viz přednáška o IP adresách
- jiná struktura IP paketu
 - IPv4:
 - jedna hlavička a možná rozšíření (OPTIONS)
 - IPv6:
 - více různých hlaviček, bez možnosti rozšiřování
 - jedna hlavička je "základní" (base header) a povinná
 - má 40 bytů
 - další hlavičky jsou "rozšiřující" (extension headers) a nepovinné
- struktura hlaviček je zjednodušená
 - málo používané položky jsou odstraněny nebo přesunuty do rozšiřujících hlaviček
 - je odstraněno přepočítávání kontrolního součtu v každém směrovači
 - ke kterému docházelo u IPv4
- je lépe řešena podpora QoS
- je řešeno zabezpečení
 - s využitím rozšiřujících hlaviček
- lepší podpora směrování
- fragmentace je řešena jinak
- další
 - místo broadcastu je multicast
 - je zaveden anycast
 -

formát IPv6 paketu

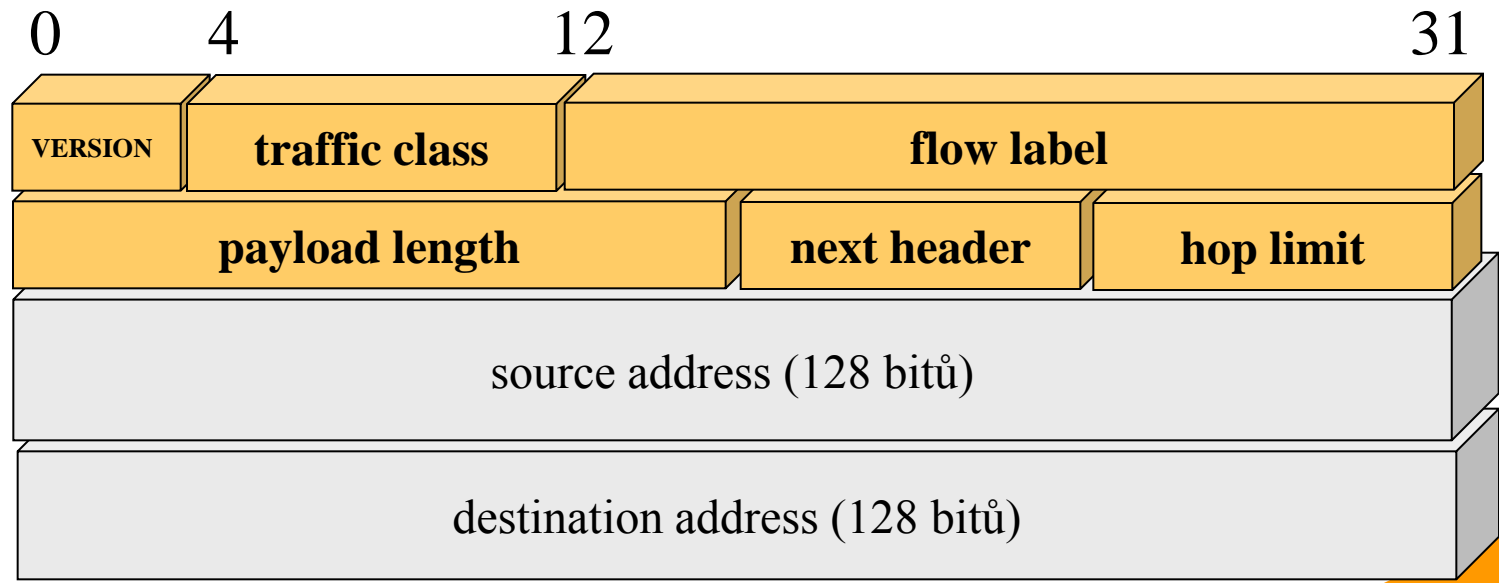
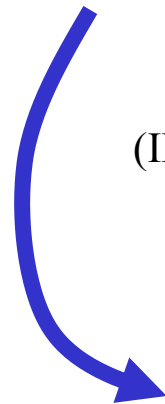
nepovinné hlavičky



(IPv4 má jen jednu hlavičku a volitelné OPTIONS)

každá hlavička vždy určuje typ následující části

v základní hlavičce IPv6 chybí kontrolní součet



IPv6 a fragmentace

- IPv6 požaduje MTU minimálně 1280 bytů
 - IPv4 požaduje nejméně 576 bytů
- už nedochází k fragmentaci "po cestě"
 - fragmentovat v IPv6 může jen odesílatel
 - pokud se někde (na trase) zjistí, že paket je větší než místní MTU, přenos skončí a odesílateli je odeslána zpráva ICMP Packet Too Big
 - jako u IPv4, pokud by byl nastaven bit DON'T FRAGMENT
 - v IPv4 mohl fragmentovat jak odesílatel, tak i kterýkoli směrovač "po cestě"
- záměr:
 - omezit režii, která vzniká při fragmentaci ve směrovačích
 - je výhodnější, když aplikace samy redukuje velikost svých paketů
- doporučení:
 - pro "plné" (chytré) implementace IPv6:
 - používat Path MTU Discovery
 - postup je obdobný jako u IPv4
 - pro "minimální" implementace:
 - generovat pakety velikosti do 1280 bytů
- rozšiřující hlavička pro fragmentaci
 - když odesílatel musí fragmentovat, přidá k povinné hlavičce rozšiřující hlavičku
 - "fragmentační" hlavičku