

# Rodina protokolů TCP/IP verze 3

## Téma 8: Protokol IPv6

Jiří Peterka

- připomenutí:
  - blokům, které IPv6 přenáší, se častěji říká **pakety** (než datagramy, jako u IPv4)
    - ale nepanuje v tom konsensus, i některá novější RFC hovoří o IPv6 datagramech
      - protipříkladem je možnost použití velkých paketů, tzv. jumbogramů
    - fakticky v tom není rozdíl, protože i IPv6 přenáší své pakety nespojovaným způsobem
      - takže je na místě i jejich označení jako datagramy
- skutečné rozdíly oproti IPv4:
  - jednodušší formát paketu
    - jiný význam položek hlavičky
  - rozšiřující hlavičky
    - jsou úspornější a efektivnější
      - než volitelné položky
  - povinná podpora multicastu
    - u IPv4 je dobrovolná
  - jiný přístup k fragmentaci
    - fragmentovat může jen odesílající uzel

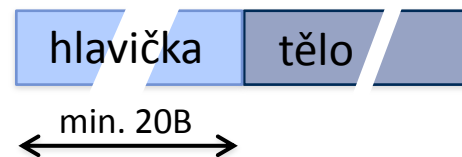
- „modernější“ směrování
  - lepší podpora hierarchického směrování
- zabudovaná podpora bezpečnosti
  - povinný IPSEC
- podpora pro alokaci zdrojů a QoS
- podpora mobility
- možnost velkých paketů
  - tzv. jumbogramů (nevyužíváno)
- .....

a samozřejmě se používají větší (128-bitové) IPv6 adresy

- více hlaviček místo jedné

- IPv4 datagram má jen jednu hlavičku proměnné velikosti

- minimálně (a obvykle) 20 bytů, včetně 2 IPv4 adres



- IPv6 paket má více hlaviček

- (povinnou) **základní hlavičku** (main header): vždy 40 bytů, včetně 2 IPv6 adres
- další **rozšiřující hlavičky** (extension headers): proměnné velikosti
  - připojují se pouze v případě, že jsou skutečně zapotřebí !!!

- dále:

- méně položek v základní hlavičce

- „méně potřebné“ položky byly přesunuty do rozšiřujících hlaviček
  - např. položky sloužící potřebám fragmentace mají vlastní rozšiřující hlavičku

- přejmenování některých položek

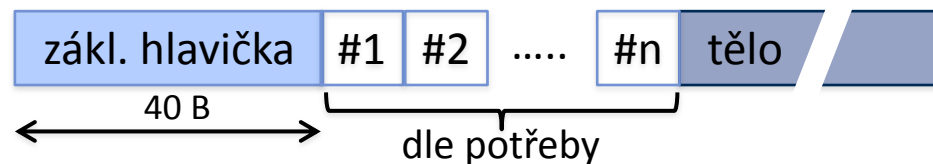
- aby lépe odpovídalo jejich skutečnému významu

- odstranění položky pro délku hlavičky (není potřeba) a kontrolního součtu

- který se musel přepočítávat při každém přeskočení (u IPv6 není co přepočítávat)

- lepší podpora QoS

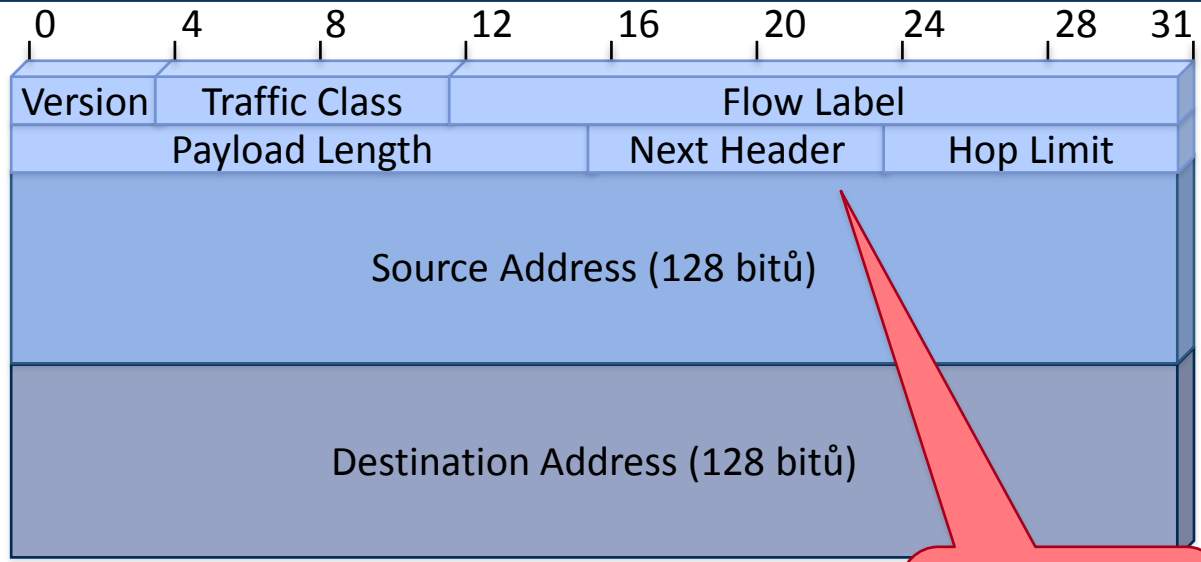
- skrze položky Traffic Class u Flow Label v základní hlavičce



# základní hlavička IPv6 paketu

- význam položek:

- **Version: 6**
- **Traffic Class**
  - nahrazuje byte ToS v IPv4 datagramu
    - pro Differentiated Services
- **Flow Label**
  - další podpora QoS a přenosům v reálném čase
    - datagram se přihlašuje k určitému proudu (flow) pro potřeby QoS
- **Payload Length**
  - velikost nákladu (16 bitů, maximální velikost paketu je tedy  $2^{16} = 64$  kB)
    - nezahrnuje základní hlavičku (ale zahrnuje rozšiřující hlavičky, pokud jsou použity)
- **Next Header:**
  - nejsou-li přítomny rozšiřující hlavičky: udává typ nákladu (jako Protocol u IPv4)
  - jsou-li přítomny rozšiřující hlavičky, udává typ první z nich
- **Hop Limit**
  - jako TTL u IPv4, jen lépe vystihuje skutečný význam

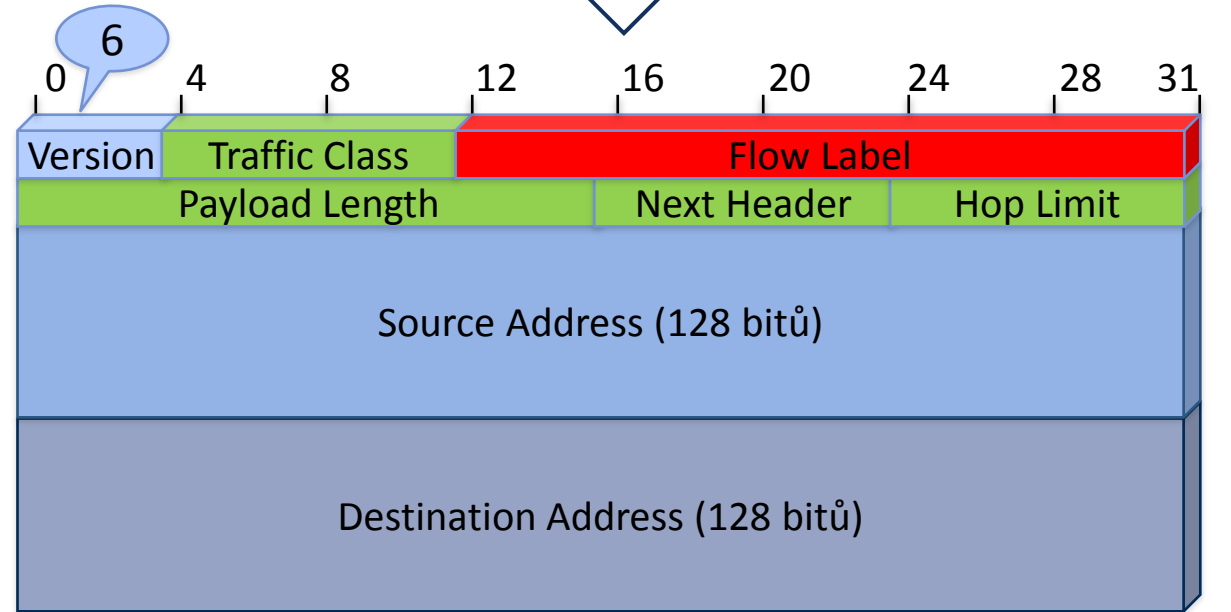
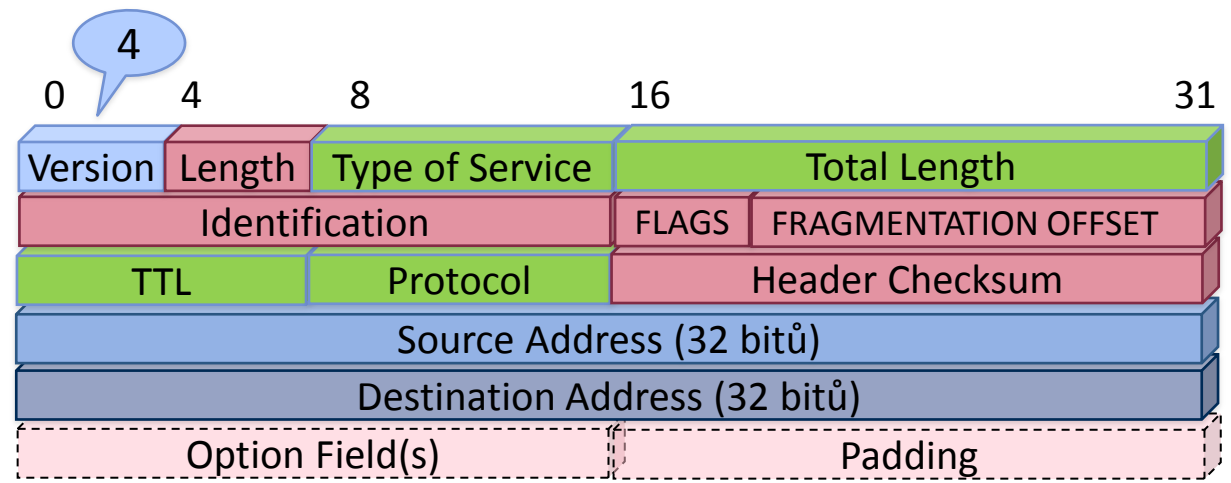


jen 8 bytů navíc ke 2 IPv6 adresám (u IPv4 to je 12)

# srovnání hlaviček IPv4 a IPv6

- beze změny:
  - Version
- zvětšeno (32 na 128 bitů):
  - Source Address
  - Destination Address
- přejmenováno:
  - ToS na Traffic Class
  - Total Length na Payload Length
  - TTL na Hop Limit
  - Protocol na Next Header
- přidáno:
  - Flow Label
- odstraněno:
  - Header Length
  - Header Checksum
  - Identification, Flags, Fragmentation Offset
  - Option Field(s), Padding

nemusí se přepočítávat



u IPv6 jde pouze o základní hlavičku !!!

# toky (flow)

- tok = skupina paketů, které spolu „nějak souvisí“
  - mají stejného odesilatele i příjemce, a navíc i nějaký stejný význam/smysl
- v IPv4 by toku odpovídala posloupnost datagramů se stejnou pěticí:

- transportní protokol,
- zdrojová IP,
- zdrojový port,
- cílová IP,
- cílový port

jenže toto jsou údaje, dostupné až na transportní vrstvě

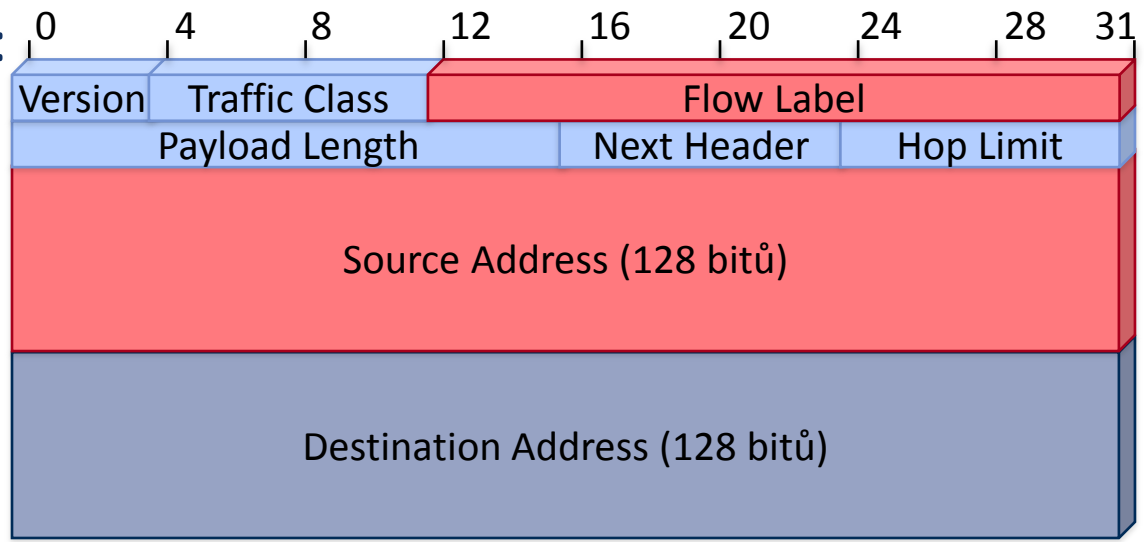
směrovač, který by je chtěl identifikovat (a zpracovávat) jako proud, by musel analyzovat nákladovou část datagramu

- v IPv6 je tok identifikován:

- zdrojovou IPv6 adresou
- identifikátorem toku
  - položkou Flow Label
    - již v základní hlavičce

- není nutné analyzovat data v těle paketu

- potřebné info je dostupné již na úrovni síťové vrstvy



- využití toků (flows)
  - nejde ani tak o zjednodušení směrování
    - resp. jeho nahrazení přepínáním (switchingem), jako např. u MPLS
  - předpokládá se spíše podpora QoS
    - různé nakládání s pakety, podle toho, do jakého patří toku
    - požadavky na QoS mohou být předkládány:
      - stavově: dopředu se sdělí všem směrovačům „po cestě“
        - prostřednictvím „rezervačního“ protokolu, např. RSVP
        - toto „sdělení“ ale musí být časově omezeno
      - bezstavově: každý paket si své požadavky nese v sobě, v rozšiřující hlavičce
        - hlavičce Hop-by-Hop

- problém:

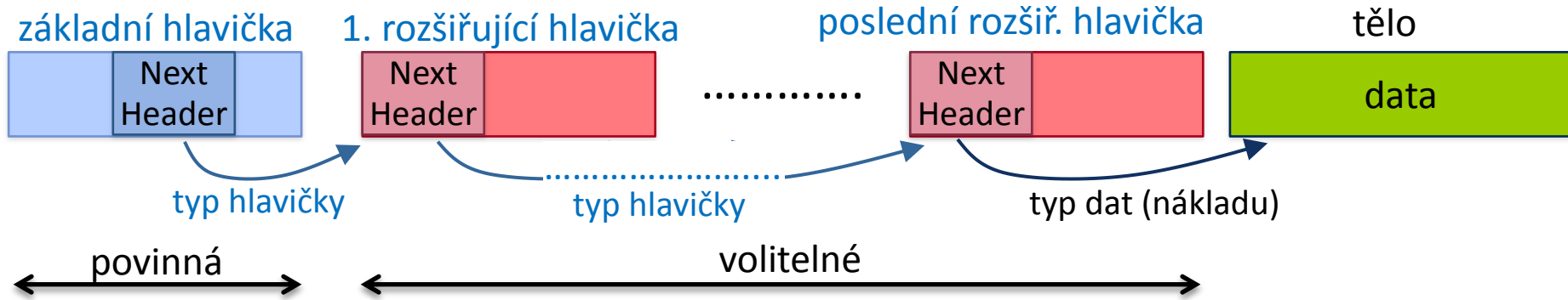
- konkrétní fungování dosud není dostatečně rozpracováno, v praxi se nepoužívá

- třída provozu (Traffic Class)
  - 1 byte v základní hlavičce
  - je určen pro vyjádření priority
    - třídy provozu ....
      - pro Differentiated Services



- DSCP: Differentiated Services Code Point
  - specifikuje prioritu
- ECN: Explicit Congestion Notification
  - určuje, zda směrovač má zahazovat pakety a/nebo posílat info o zahlcení

# rozšiřující hlavičky



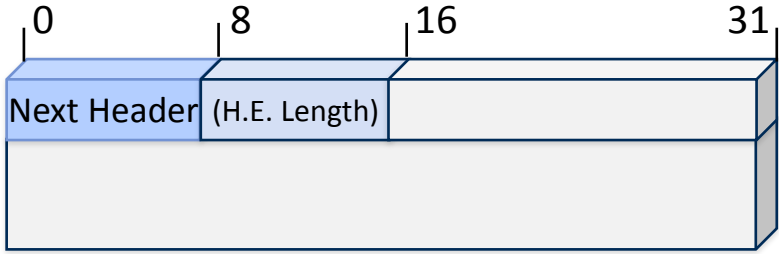
- rozšiřující hlavičky: jsou volitelné (nepovinné)
  - nahrazují doplňky (options) v IPv4, ale šikovněji
  - „odlehčují“ hlavičky IPv6 paketu
    - to, co je v hlavičce IPv4 datagramu využíváno jen občas, se v IPv6 přesouvá do rozšiřující hlavičky
  - nabízí více možností, než má IPv4
- jsou řazeny v sérii za sebou
  - položka Next Header udává **typ další rozšiřující hlavičky**
    - nebo **typ nákladu** v těle paketu (u poslední hlavičky)
    - speciální typ 59: už nenásleduje nic, ani tělo paketu
      - pokud by následovalo, musí být ignorováno
- identifikátory typů v položce Next Header
  - stejně jako u IPv4
    - spravuje je IANA
      - Protocol Numbers
  - například:
    - 0 = Hop-by-Hop options (typ hlavičky)
    - 6 = TCP, 17 = UDP (typ nákladu)
    - 44 = fragmentace
    - 59 = nic nenásleduje



# rozšiřující hlavičky

- některé rozšiřující hlavičky mají proměnnou velikost, ostatní pevnou

- 1. položkou je vždy Next Header (1 byte)
  - zbytek se liší
    - podle toho, o jakou rozšiřující hlavičku jde
- u položek s proměnnou velikostí:
  - 2. položkou je údaj o velikosti (Header Extension Length)

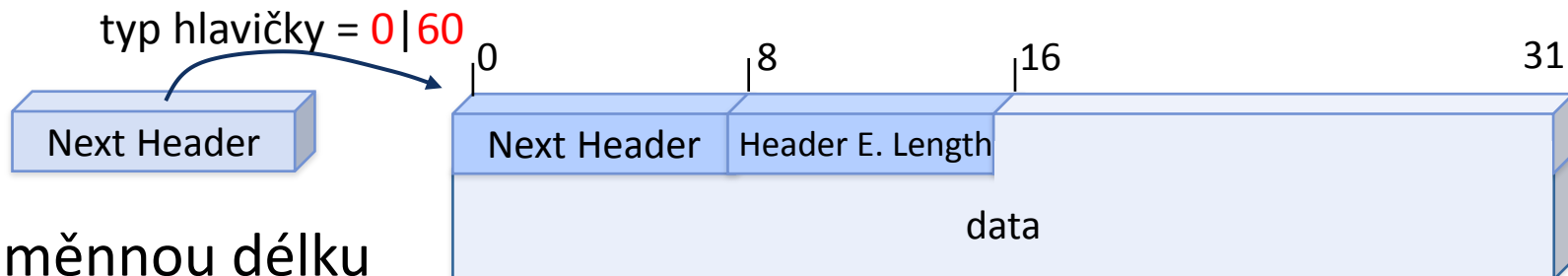


výjimka: Destination Options může 2x

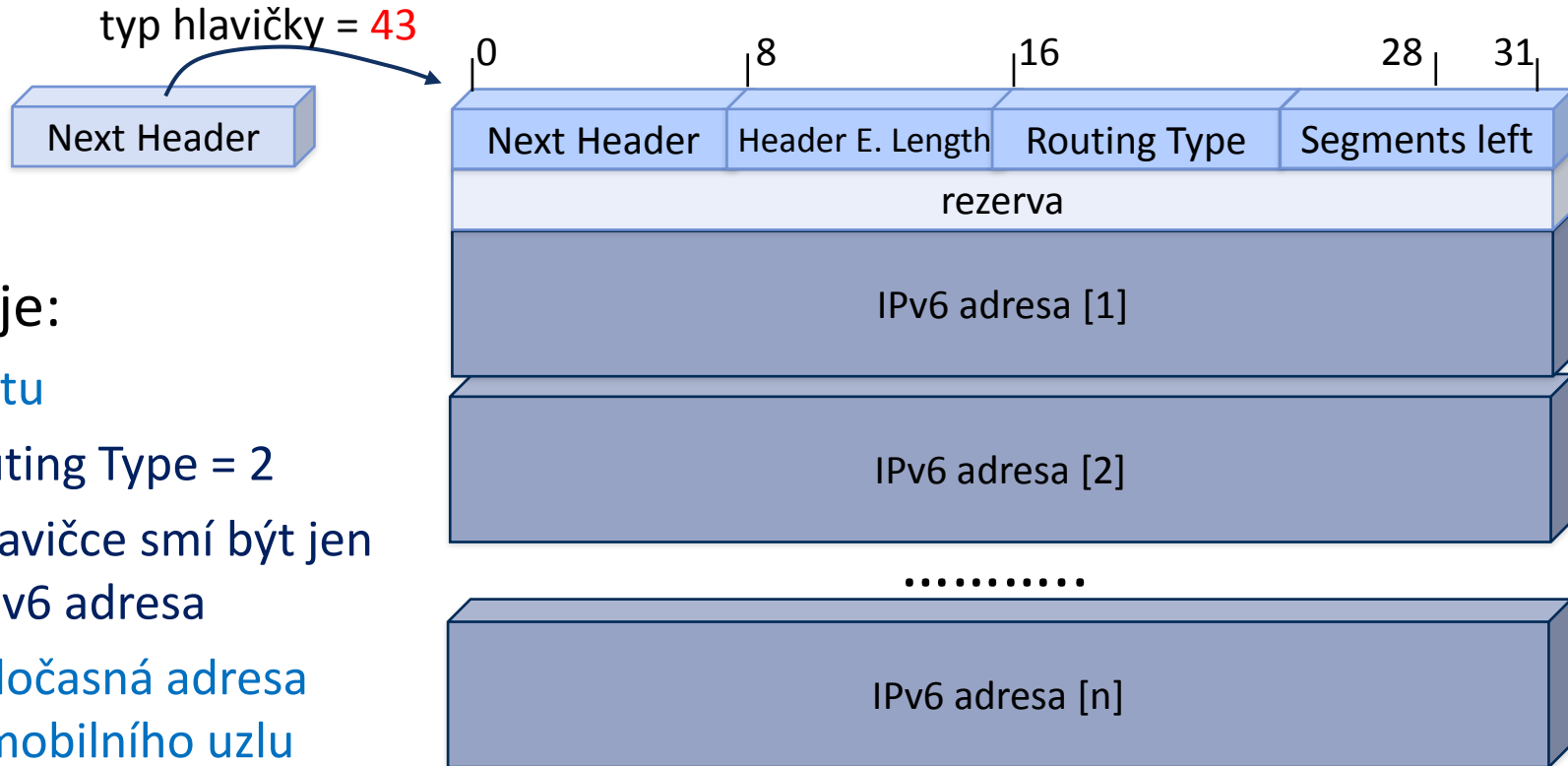
- řazení rozšiřujících hlaviček:

- každý typ rozšiřující hlavičky by se (v každém paketu) měl vyskytnout nejvýše 1x
- pořadí není striktně předepsáno, pouze velmi doporučeno
  - cílem je dát na začátek řetězce ty hlavičky, které je třeba zpracovat nejdříve
    - hlavně usnadnit práci směrovačům, aby nemusely analyzovat všechny hlavičky

1. základní hlavička IPv6
2. Hop-by-Hop Options
  - týká se „všech uzlů po cestě“
3. Destination Options
  - pro uzly v rámci source routing
4. Routing (směrování)
5. Fragment (fragmentace)
6. Authentication, AH (autentizace)
  - pro potřeby IPSEC
7. Encapsulating Security Payload, ESP
  - šifrování obsahu (pro IPSEC)
8. Destination Options
  - pro konečného příjemce
9. Mobility



- mají proměnnou délku
- jsou zamýšleny jako „zadní vrátka“
  - pro nejrůznější budoucí rozšíření (na která nebylo pamatováno již při návrhu IPv6)
- **Hop-by-Hop Options**
  - hlavička je na začátku řetězce rozšiřujících hlaviček (předchozí Next Header = 0)
    - je určena pro „každý uzel na cestě“ – hlavně pro všechny směrovače
- **Destination Options**
  - týká se koncového příjemce (předchozí Next Header = 60)
    - a také směrovačů („průchozích uzlů“), předepsaných v rámci source routingu
- zatím je definováno jen málo možností, například:
  - upozornění směrovači
    - pokyn směrovači, aby detailněji zkoumal obsah přenášených dat
  - jumbo obsah (pro pakety, větší než 64 KB)



- umožňuje:

- mobilitu

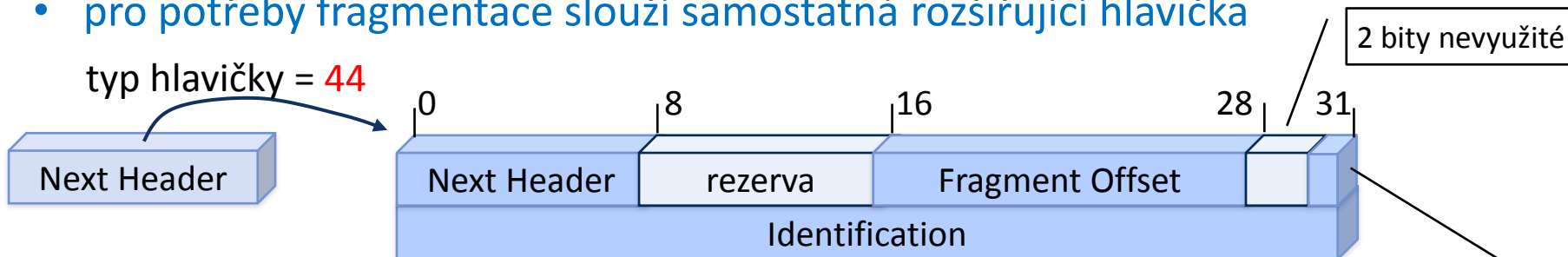
- Routing Type = 2
    - v hlavičce smí být jen 1 IPv6 adresa
    - dočasná adresa mobilního uzlu

- paket je pak fakticky doručen nejprve na dočasnou adresu mobilního uzlu

- source routing

- Routing Type = 0
    - v hlavičce je posloupnosti IPv6 adres, přes které má paket projít
      - podle RFC 5095 (2007) se již nemá používat, směrovače mají zahazovat
        - důvodem je možnost zneužití – snadné generování velkých datových toků, které může využít útočník

- rozdíl oproti IPv4:
  - v IPv6 může fragmentovat **pouze odesílající uzel !!**
    - pokud se chce „vyhnout problémům“, může:
      1. posílat IPv6 paket do max. velikosti **1280 bytů**
        - v IPv6 je garantováno, že takto velké pakety projdou bez nutnosti fragmentace
      2. zjistit si nejmenší MTU po cestě
        - skrze techniku MTU Path Discovery
  - směrovače v IPv6 už nefragmentují
    - aby nebyly zatěžovány dalšími úkoly
      - pokud směrovač narazí na paket, který by měl být fragmentován, musí ho zahodit
  - pro potřeby fragmentace slouží samostatná rozšiřující hlavička

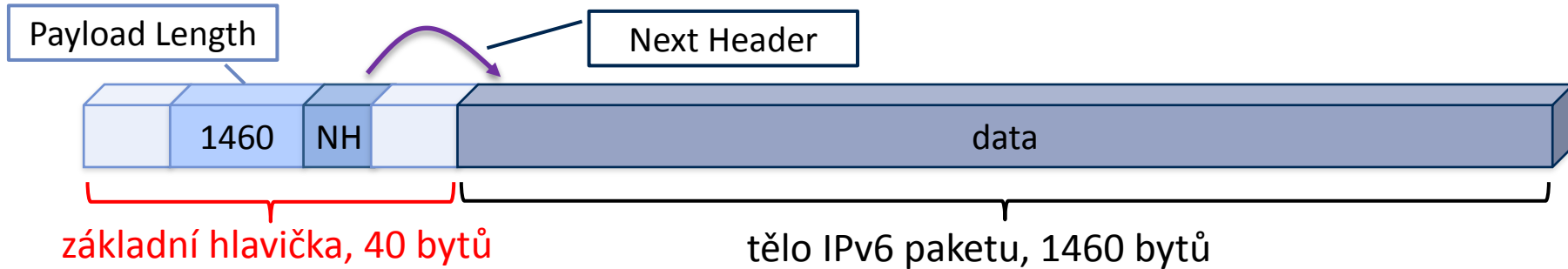


- **Fragment Offset**: stejně jako Fragmentation Offset u IPv4
  - posunutí vůči začátku datové části původního paketu
- **Identification**: stejné jako u IPv4 (ale v rozsahu 32 bitů)

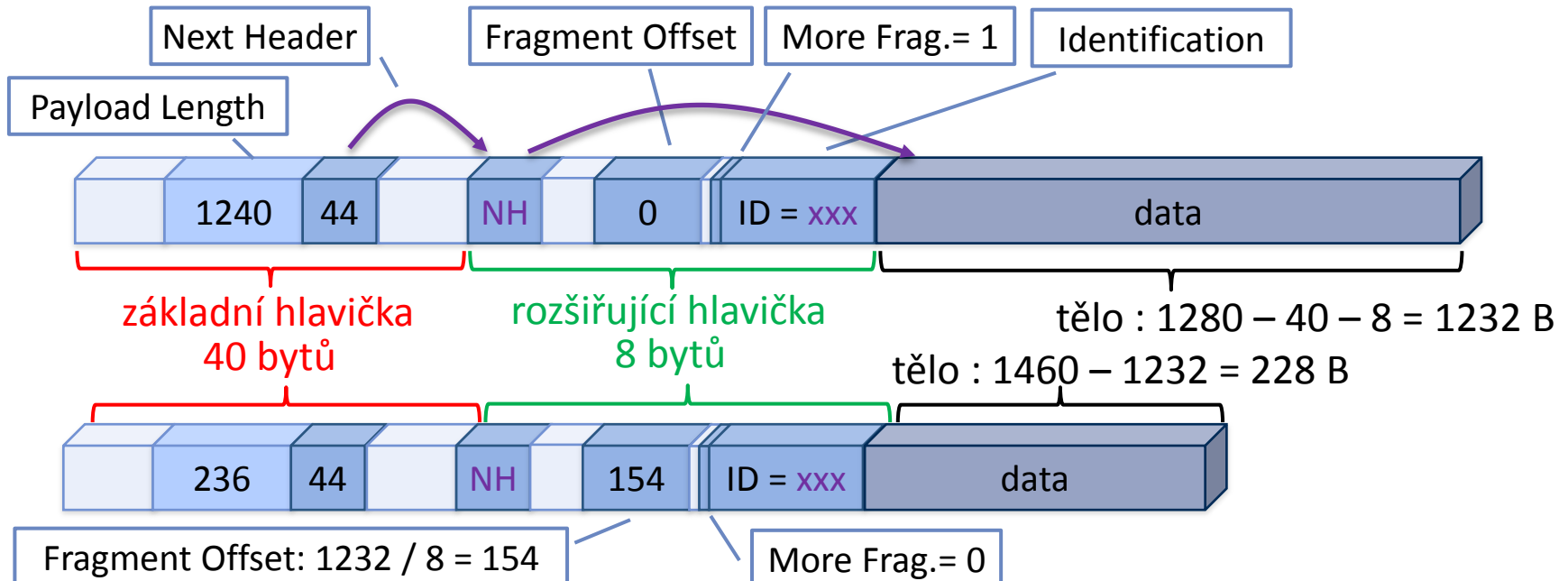
**More Fragments Flag:**  
1: jsou další fragmenty  
0: nejsou

# příklad fragmentace

- původní IPv6 paket má 1500 bytů
  - byl sestaven uzlem, který je zapojen (např.) do sítě Ethernet, s MTU=1500 bytů

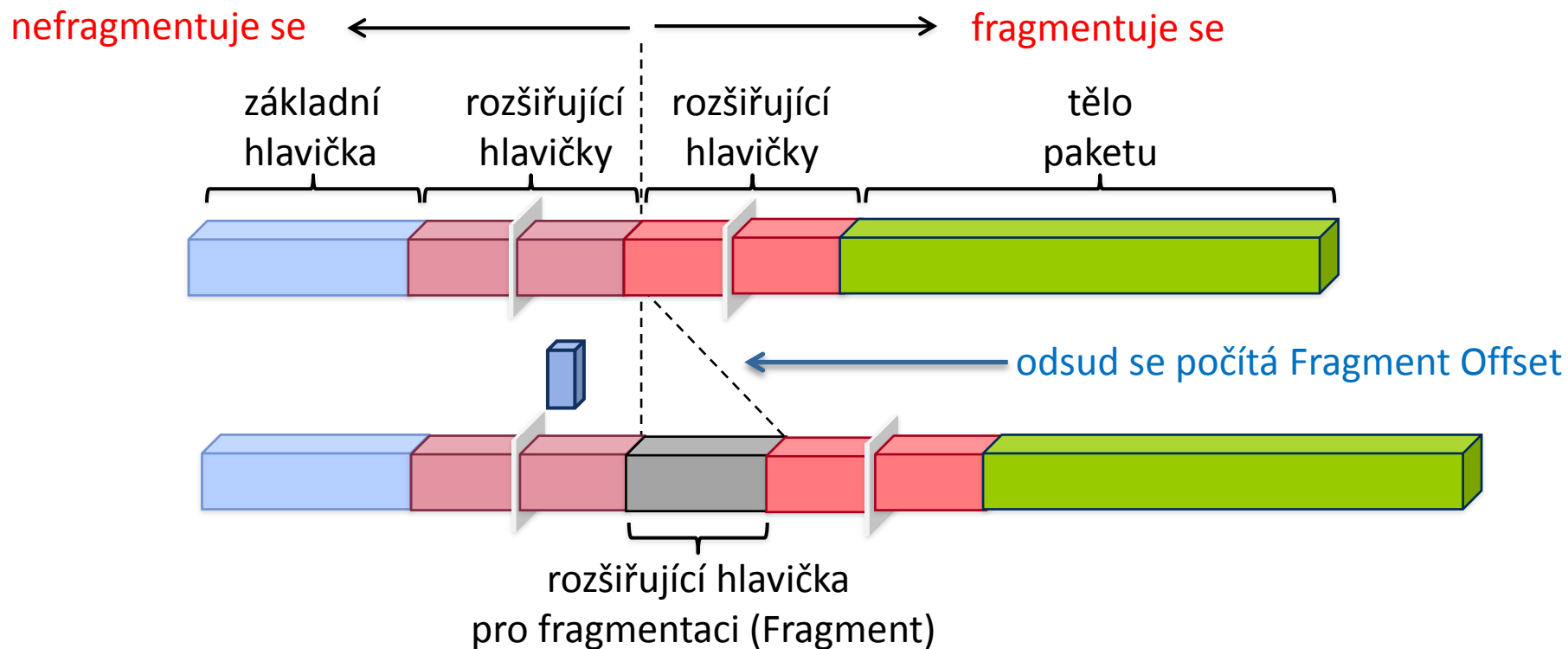


- ale „po cestě“ se zjistilo, že neprojde a je třeba jej zmenšit na 1280 B
  - celkem tedy na 2 fragmenty, o celkové velikosti paketů 1280 bytů a 276 bytů



# fragmentace v IPv6

- co s rozšiřujícími hlavičkami v případě fragmentace?
  - „počáteční“ rozšiřující hlavičky se nefragmentují
    - základní hlavička, Hop-by-Hop Options, Destination Options, Routing
  - „koncové“ rozšiřující hlavičky se naopak fragmentují
    - Authentication, Encapsulating Security Payload, Destination Options, Mobility
- rozšiřující hlavička Fragment se při fragmentaci vsouvá „mezi ně“

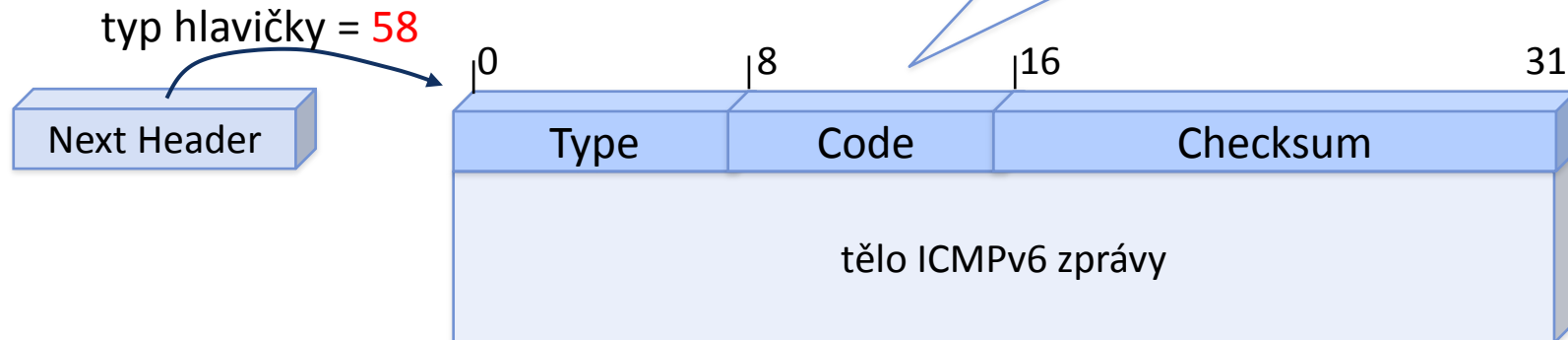


# protokol ICMPv6

- slouží
  - ke stejným účelům, jako ICMP (Internet Control Message Protocol) u IPv4
    - pro signalizaci „nestandardních situací“ v důsledku fungování protokolu IP
- zprávy ICMPv6 se přenáší
  - uvnitř IPv6 paketů (Next Header = 58)
  - stejně jako u IPv4, stejné porušení principů vrstevnatého modelu

- ICMPv6 zprávy mohou být:

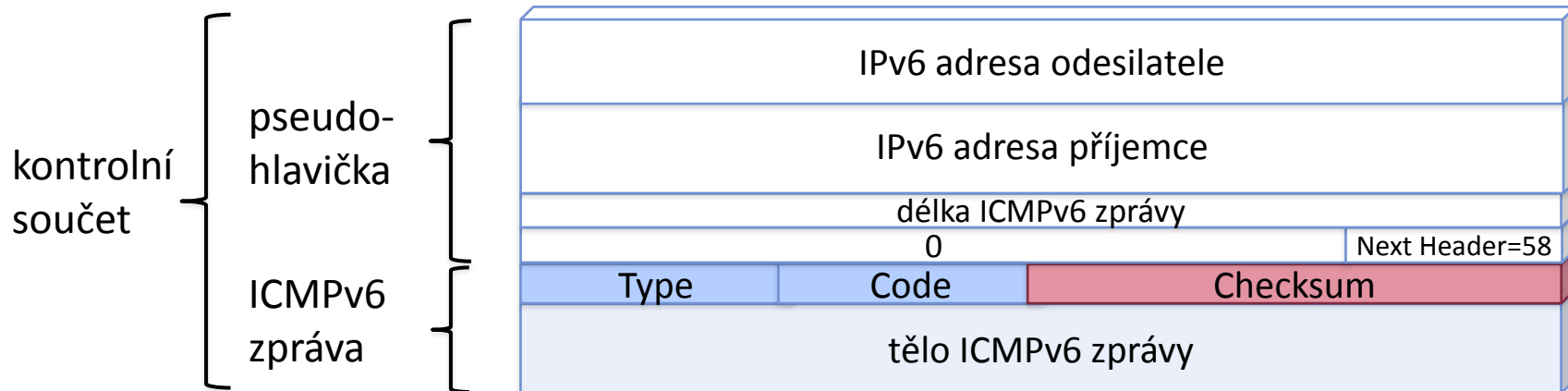
- **chybové** (Type = 0 až 127)
- **informační** (Type = 128 až 255)



- obsahuje také:

- kontrolní součet (Checksum), 16 bitů, počítaný včetně tzv. pseudohlavičky

- velikost chybových ICMP zpráv:
  - v těle ICMPv4 zprávy je hlavička IPv4 datagramu + prvních 8 bytů jeho těla
  - tj. jde o pevnou velikost
  - v těle ICMPv6 zprávy je co největší část IPv6 paketu, který způsobil chybu
  - „co největší“ = taková, aby velikost IPv6 paketu s ICMPv6 zprávou nepřesáhla 1280 bytů
    - a nehrozila fragmentace
- výpočet kontrolního součtu ICMP zprávy
  - u ICMPv4 se počítá (pouze) ze samotné zprávy
  - u ICMPv6 se počítá navíc také z tzv. **pseudohlavičky**
    - důvod: ochrana zprávy před chybným doručením
    - stejně je tomu u UDP datagramů i TCP segmentů
      - jak u IPv4, tak u IPv6



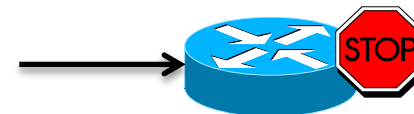


# chybové zprávy ICMPv6

- jsou celkem 4:

- **Type=1: Destination Unreachable**

- nelze pokračovat v přenosu, cíl je nedosažitelný



- možné důvody:

- **Code=0: No route to destination** – nelze najít cestu k cíli

- **Code=1: Administratively Prohibited** – přenos je zakázán správcem, např. na firewallu

- důvody lze podrobněji rozlišit pomocí jiných hodnot Code

- Code=5 | 6: nepřípustná (zakázaná, na black listu) je zdrojová | cílová adresa

- **Code=2: Beyond scope of source address** – nešlo by odpovědět zpět

- **Code=3: Address unreachable** – cílový uzel je nedosažitelný (neodpovídá)

- **Code=4: Port unreachable** – cílový port je nedosažitelný

- **Type=2: Packet Too Big**

- reakce směrovače na příliš velký paket, který by měl být fragmentován

- v těle ICMP zprávy je kromě zahozeného paketu i „nová“ hodnota MTU



- **Type=3: Time Exceeded**

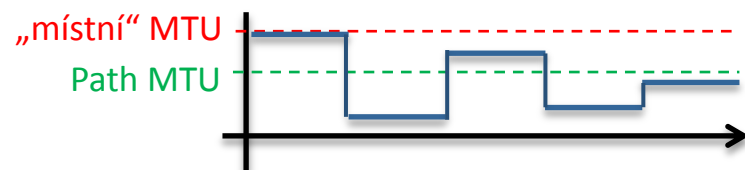
- reakce na vynulování položky Hop Count či na to, že chybí některý z fragmentů

- **Type=4: Parameter Problem**

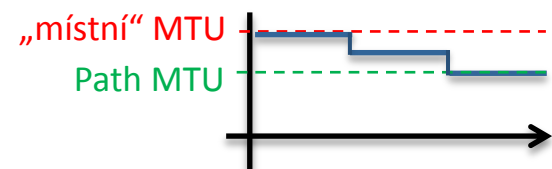
- reakce na jiné chyby

# Path MTU Discovery

- postup zjišťování (nejmenšího) MTU po cestě od daného uzlu ke zvolenému cíli
- Path MTU Discovery v IPv4:
  - zdrojový uzel odesílá datagramy určité velikosti (začne s velikostí místního MTU)
    - se zakázanou fragmentací (nastaveným bitem Don't Fragment)
  - pokud datagram kvůli velikosti neprojde, dostane zdroj zpět ICMPv4 zprávu **Destination Unreachable**, s **Code=4** (Fragmentation Needed nad DF Set)
    - z ní se ale **nedozví**, jaké je MTU v dalším úseku, který vyvolal potřebu fragmentace
  - proto cílový uzel odhadne novou (nižší) hodnotu MTU a iteruje
    - pokud znovu neprojde, MTU ještě sníží
    - pokud již projde, MTU zase o něco zvýší



- Path MTU Discovery v IPv6:
  - zdrojový uzel odešle paket o velikosti místního MTU
  - pokud paket kvůli velikosti neprojde, dostane zdroj zpět ICMPv6 zprávu **Packet Too Big**
    - ze které se **dozví**, jaké je MTU v úseku, který vyvolal potřebu fragmentace
  - použije novou (nižší) hodnotu MTU (... a celý postup opakuje)



- je jich větší počet, například:
    - ICMPv6 Echo Request a Echo Reply
      - fungují obdobně jako u ICMPv4
    - ICMPv6 Router Advertisement a Router Solicitation
      - jako u ICMPv4: „inzerát“ o existenci směrovače a výzva „je zde směrovač?“
    - ICMPv6 Neighbor Advertisement a Neighbor Solicitation
      - obdobně (jako pro směrovače), ale pro koncové uzly (hosts)
    - ICMPv6 Redirect
      - jako u ICMPv4: reakce na nesprávné směrování
        - rozdíl: v ICMPv4 je to chybové hlášení (v reakci na chybu), zde informační zpráva
    - ICMPv6 Router Renumbering
      - specialita v6: možnost změny prefixu (síťové části IPv6 adresy)
- využívá je utilita Ping6
- využívají se zejména pro protokol IPv6 Neighbor Discovery
- a ještě mohou být kombinovány s volitelnými doplňky (options)
    - které obsahují např. linkovou adresu, síťový prefix, hodnotu MTU apod.

# IPv6 Neighbor Discovery

RFC 4861

- IPv4: pro převod IPv4 adres na HW adresy používá protokol ARP
  - nebo převod tabulkou či přímým výpočtem
- IPv6 má pro tyto účely „šikovnější“ protokol **Neighbor Discovery**

- který slouží více účelům/potřebám současně:

- **Address Resolution**



+



- převod IPv6 adres na HW adresy (jako náhrada protokolu ARP u IPv6)
- včetně rychlé aktualizace položek a zjišťování změn v HW adresách

- **Router Discovery**



+



- hledání směrovačů, dostupných v dané síti
- směrovače mohou inzerovat svou dostupnost (včetně své HW adresy, MTU atd.)

- **Prefix Discovery**



- zjišťování síťového prefixů a dalších parametrů sítě

- **Parameter Discovery**



- zjištění „místního“ MTU a počáteční hodnoty, na kterou by měl nastavit Hop Count

- **Address Auto-Configuration**



- možnost, aby si uzel sám zvolil (nastavil) svou IPv6 adresu

- **Neighbor Unreachability Detection**



- zjišťování (ne)dostupnosti sousedních uzlů

- **Duplicate Address Detection**



- zjišťování duplicitních adres

- **Redirect**



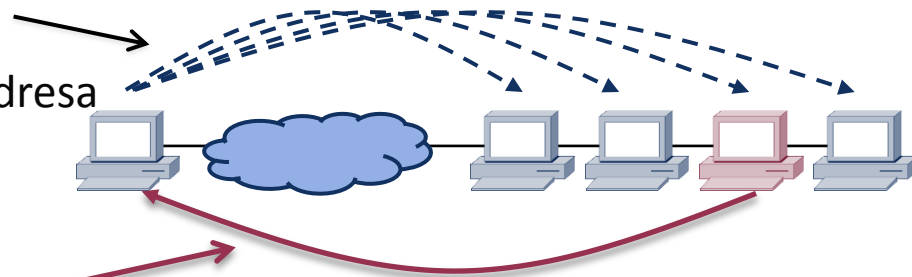
- informování hostitelského počítače o existenci lepší cesty

- ke svému fungování využívá informační ICMPv6 zprávy:
  - Router Solicitation, Router Advertisement
  - Neighbor Solicitation, Neighbor Advertisement
  - Redirect

} považují se za součást (zprávy) protokolu Neighbor Discovery
- předpokládá, že každý uzel (směrovač i host) má vlastní:
  - **Neighbor Cache:** obdoba ARP cache v IPv4, týká se uzlů ve stejné síti
    - ve které si uchovává údaje o svých susedech (se kterými komunikoval)
      - unicast adresu suseda, jeho HW adresu, zda jde o host/směrovač, aktuální dostupnost, doba dalšího testování dostupnosti, zda jsou pro něj připravena (v datové cache) data k odeslání
  - **Destination Cache:** týká se všech uzlů, se kterými daný uzel komunikoval
    - „blízkých uzlů“ (ve stejné síti) i „vzdálených uzlů“ (v jiných sítích)
    - jde o nadmnožinu k Neighbor Cache (zahrnuje ji)
      - pro „vzdálené uzly“ je v cache HW adresa 1. hop-u (odchozího směrovače)
  - **Prefix List:** seznam prefixů, které používají přímo dosažitelné (neighbor) uzly
  - **Default Router List:** seznam směrovačů v dané síti
    - všech, které mohou být implicitním (default) směrovačem

# Address Resolution

- princip je stejný jako u IPv4 (s využitím protokolu ARP):
  - IPv4: „do pléna“ (pomocí broadcastu) se rozešle dotaz (ARP zpráva) s hledanou IP adresou, její držitel odpoví ARP zprávou již cíleně (unicast-em) přímo tazateli
- v IPv6:
  - neexistuje broadcast, **dotaz se pošle na multicastovou adresu**
    - nikoli na `ff02::01` (tj. „všechny uzly“ s dosahem „link-local“)
    - ale na **adresu pro vyzývaný uzel** (solicited node address) `ff02:0:0:0:0:1:ffxx:xxxx`
      - kde `xx.xxxx` je posledních 24 bitů hledané IPv6 adresy
      - fakticky jde většinou o unicast (členem skupiny v dané síti je pouze jeden uzel)
        - uzlů „se stejným koncem“ IPv6 adresy ve stejné síti je obvykle jen minimum
    - dotaz nemá podobu ARP zprávy, ale ICMPv6 zprávy **Neighbor Solicitation**
      - ve které se vyplní (celá) hledaná IPv6 adresa
    - odpověď má podobu ICMPv6 zprávy **Neighbor Advertisement**
      - s vyplněnou HW adresou, posílá se zpět



již cíleně, na unicast adresu tazatele

# Neighbor Solicitation a Advertisement

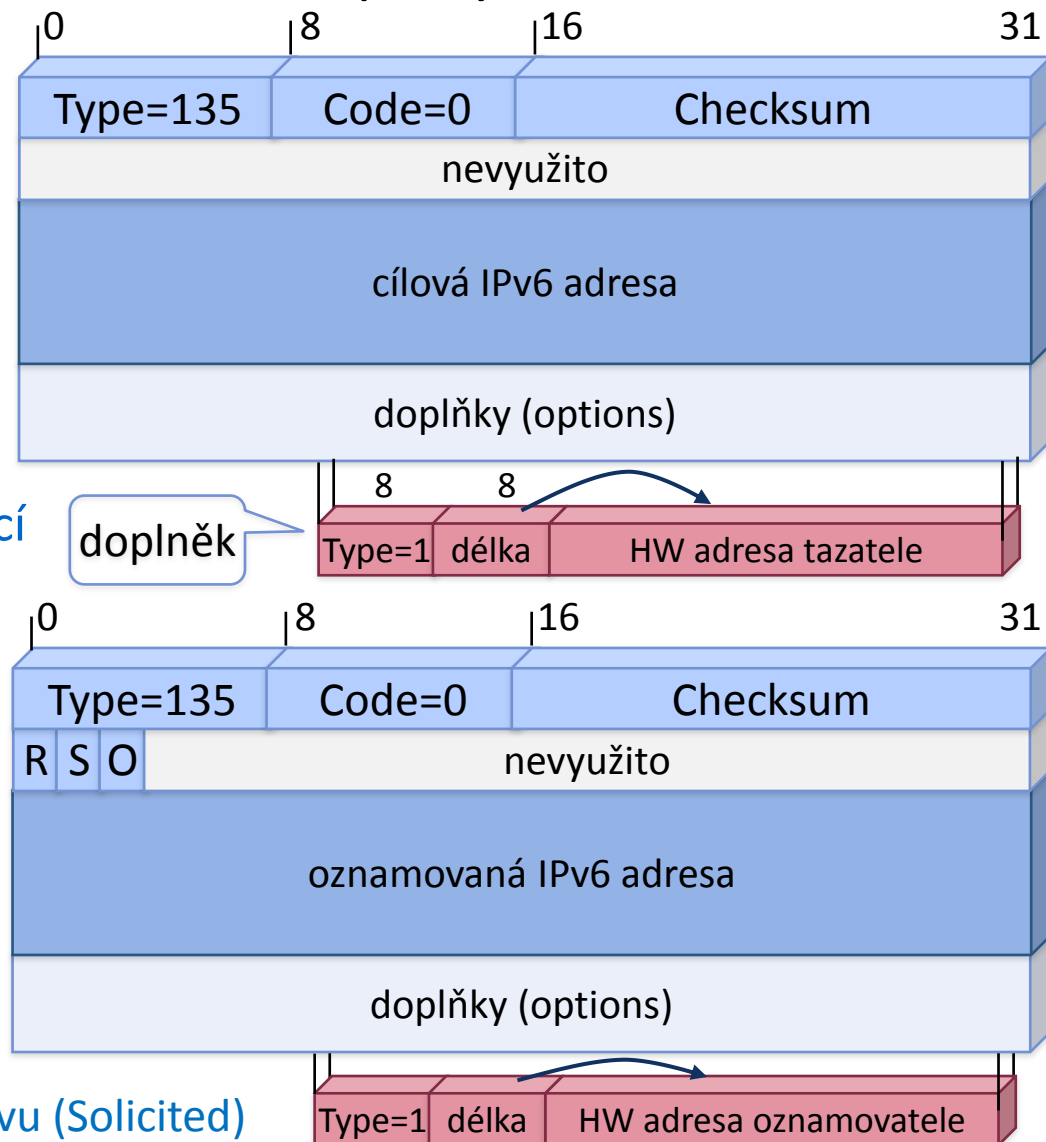
- jde o vzájemně „komplementární“ ICMPv6 zprávy

- **Neighbor Solicitation** je výzvou

- obsahuje cílovou IPv6 adresu
  - které se týká dotaz
- doplněk může obsahovat HW adresu tazatele
  - kterou si příjemce uloží do své neighbor cache

- **Neighbor Advertisement** je reakcí

- obsahuje „oznamovanou“ IPv6 adresu
  - původně (v dotazu): cílovou
- doplněk musí obsahovat HW adresu oznamujícího uzlu
- příznaky:
  - R: odesílatel je/není směrovač
  - S: jde/nejde o odpověď na výzvu (Solicited)
  - O: nová informace má přepsat dosavadní informace o dané adrese (Override)



- pokud uzel zjistí změnu své HW adresy, může ji sám aktivně ohlásit
  - rozešle ICMPv6 zprávu Neighbor Advertisement na multicastovou adresu ff02::01
    - tj. všem uzlům v dané síti („všechny uzly“ s dosahem „link-local“)
    - s vyplněnou vlastní IPv6 adresou a vlastní (novou) HW adresou
      - v doplňku ICMPv6 zprávy
- ostatní uzly si osvěží svou Neighbor cache
  - pokud v ní již mají IPv6 adresu oznamujícího uzlu, aktualizují si jeho HW adresu
    - jinak nedělají nic, aby zbytečně nezaplňovali svou neighbor cache
- každý uzel průběžně sleduje dostupnost svých sousedů
  - a informace o ní si vede ve své Neighbor Cache
    - tj. informace o dostupnosti zde pravidelně zastarává a je nutné ji obnovovat
- možnosti obnovení informace o dostupnosti sousedního uzlu
  - informace od vyšších vrstev
    - ty informují, zda komunikace se sousedním uzlem stále probíhá
  - vlastní test
    - pokud se údaj v Neighbor cache blíží své expiraci, daný uzel pošle svému sousedovi zprávu Neighbor Solicitation
      - a dle výsledku obnoví informaci o dostupnosti ve své Neighbor cache

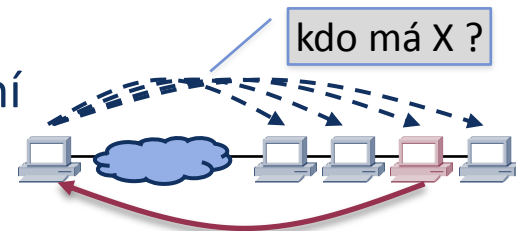


# Duplicate Address Detection

- je řešením (např.) pro situaci, kdy:
  - uzel A si sám zvolí svou IPv6 adresu (pomocí autokonfigurace): **adresu X**
  - ale potřebuje se ještě ujistit, že je unikátní
    - že ji nepoužívá současně i někdo jiný – zda není duplicitní
- postup
  - uzel A se snaží provést Address Resolution na svou novou IPv6 **adresu X**
    - postupuje stejně jako při skutečném Address Resolution
      - tj. do zprávy Neighbor Solicitation vloží **adresu X** jako cílovou IPv6 adresu
    - ale: **adresu X** ještě nesmí používat jako svou adresu (jako adresu odesílatele)
    - proto: 

od ::	kdo má X ?	adresa vyzývaného uzlu (X)
-------	------------	----------------------------
    - v hlavičce odesílaného IPv6 paketu s ICMPv6 zprávou uzel A místo své nové IPv6 **adresy X** použije **nespecifikovanou IPv6 adresu (samé 0, resp. ::)**
  - pokud jiný uzel B (ve stejné síti) již používá stejnou IPv6 **adresu X**, odpoví
    - ale: nemůže odpovědět přímo uzlu A, protože nezná jeho IPv6 adresu
    - proto: 

všem uzlům v dané síti	od B	já mám X !
------------------------	------	------------
    - svou odpověď rozešle na multicastovou adresu ff02::01
      - tj. na všechny uzly v dané síti – mezi nimi je i uzel A, který odpověď přijme

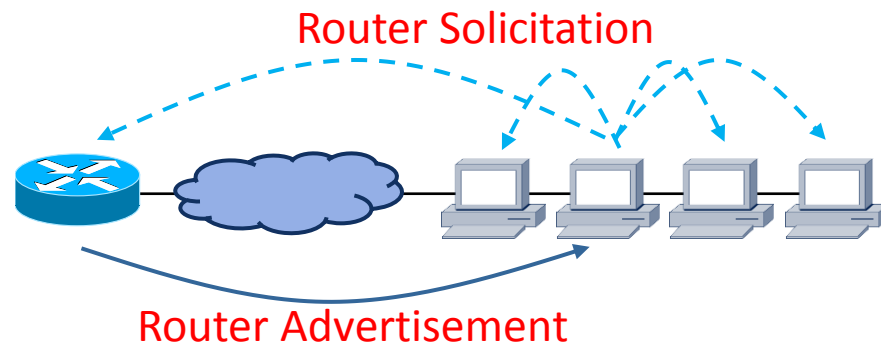
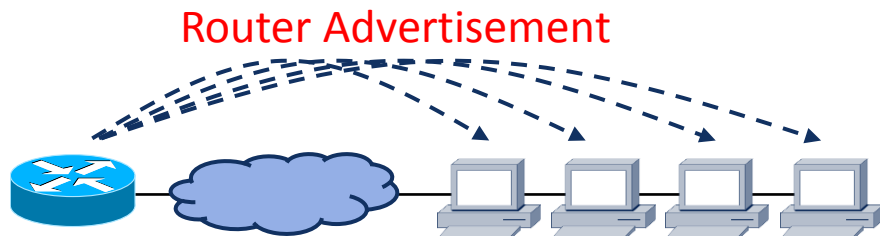


- zjednodušeně:

- směrovače průběžně inzerují svou existenci
- opakovaným rozesíláním zpráv **Router Advertisement** na adresu ff02::01
  - tj. všem uzlům v dané síti
- včetně informací o své HW a IPv6 adrese
  - i jak dlouho bude směrovač dostupný, jak často mají uzly testovat jeho dostupnost
  - s jakým síťovým prefixem pracuje, s jakým MTU pracuje, .....

- hostitelské počítače se mohou chovat:

1. pasivně: počkat, až směrovač rozešle další svou zprávu **Router Advertisement**
2. aktivně: rozeslat zprávu **Router Solicitation** na adresu ff02::02
  - tj. všem směrovačům v dané síti
  - a tím je „vyprovokovat k reakci“ – k odpovědi formou zprávy **Router Advertisement**
    - odpověď je již cíleně adresována tazateli (pomocí jeho unicast adresy)



# zpráva Router Advertisement

- zprávy Router Advertisement obsahují také další informace, určené pro konfiguraci koncových uzlů (host-ů)

- položky:

- **Current Hop Limit**

- doporučená hodnota Hop Limit pro odesílané pakety

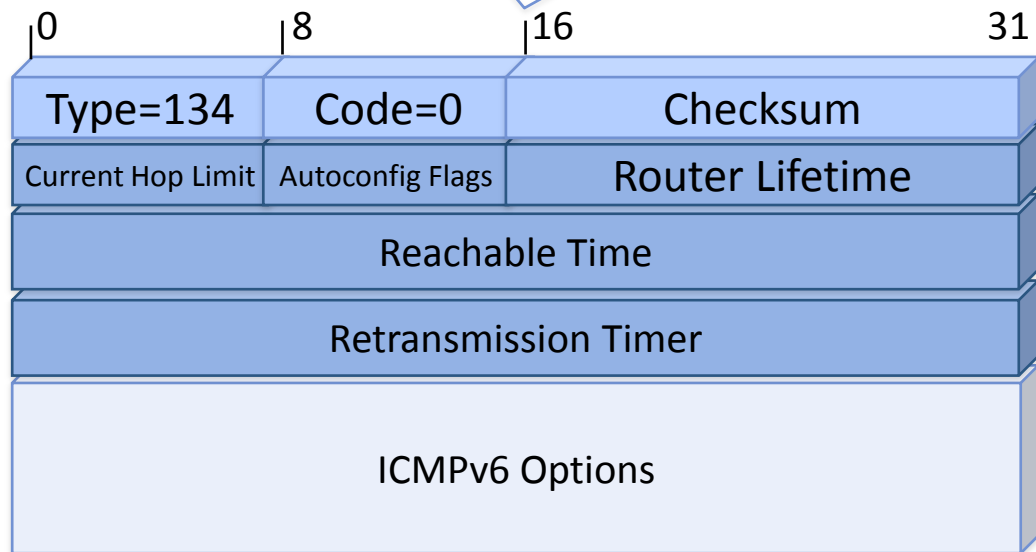
- **Autoconfig Flags**

- příznak M (Managed Address)
  - příjemce (koncový uzel) by měl využít stavovou konfiguraci (DHCP) pro přidělení své adresy
- příznak O (Other ..)

- **Router Lifetime**

- čas v sekundách, po který by měl být (odesílající) směrovač považován příjemce za implicitní směrovač
  - 0 = neměl by být považován za implicitní

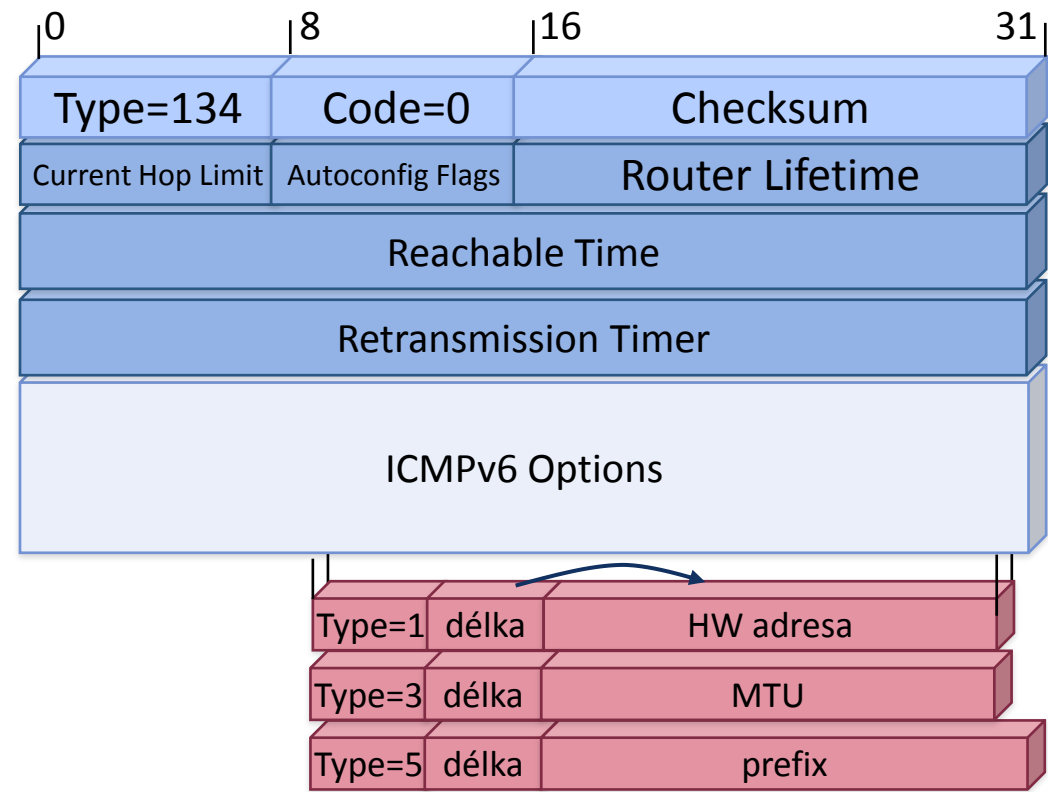
o který směrovač jde, vyplývá z IPv6 adresy odesílatele



# zpráva Router Advertisement

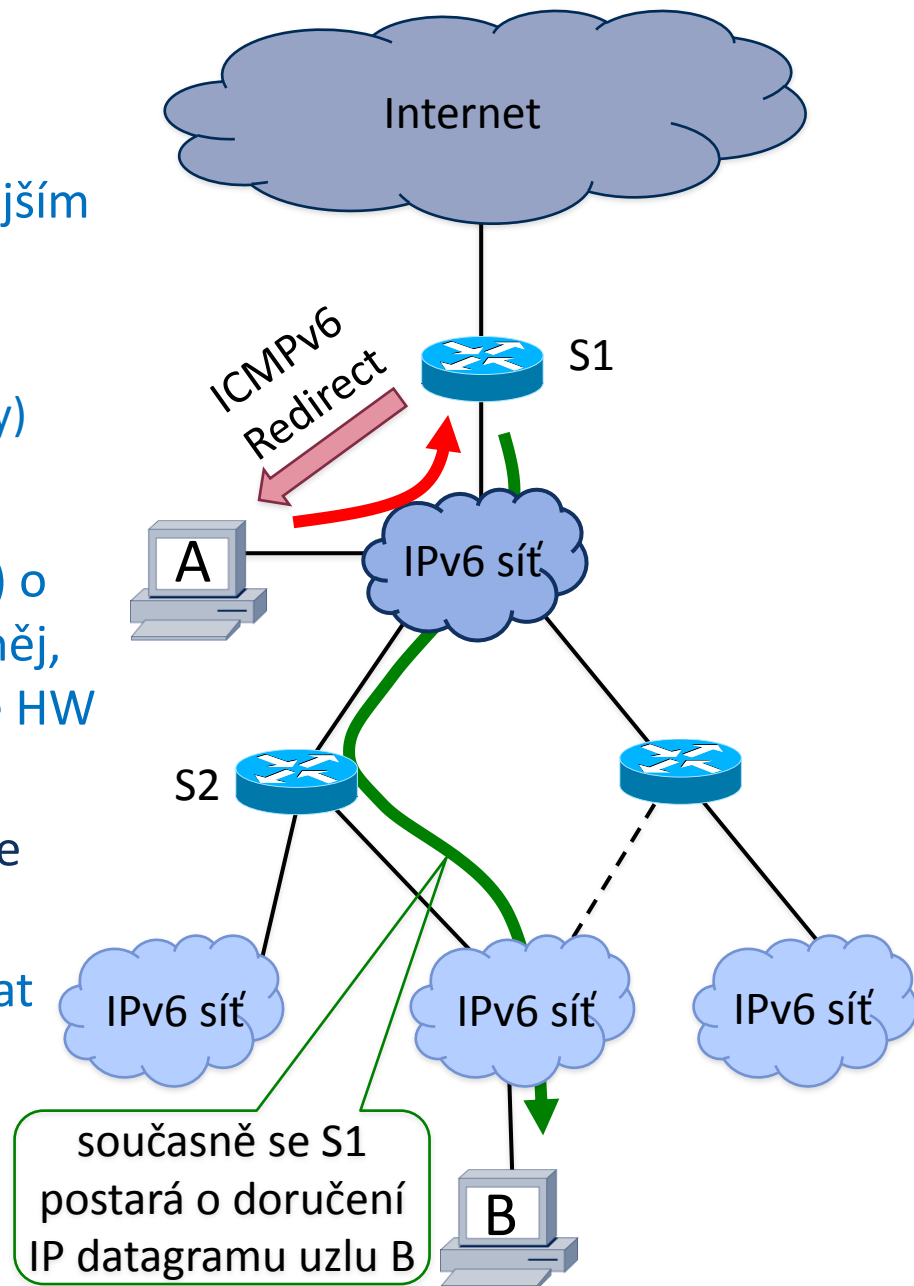
- položky **Reachable Time** a **Retransmission Timer**
  - týkají se toho, jak má příjemce zprávy (koncový uzel) testovat dostupnost svých sousedů
    - jak dlouho má být souseď považován za dostupného, od posledního kontaktu
      - dle **Reachable Time**
    - s jakými odstupy mají být zasílány zprávy Neighbor Solicitation (zjišťování dostupnosti)
      - dle **Retransmission Timer**

- položky **Options**
  - souvisí s konfigurací koncových uzlů
  - mohou být až 3 různé hodnoty
  - **HW adresa (Type=1)**
    - toho směrovače, který posílá zprávu Advertisement
  - **MTU (Type=3)**
    - hodnota MTU v místní síti
  - **prefix (Type=5)**
    - počet bitů a hodnota prefixu/ů



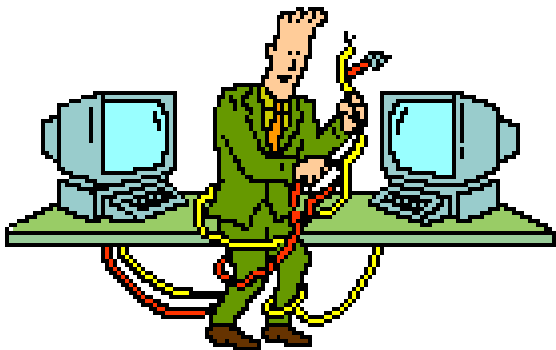
# Redirect

- funguje obdobně jako u verze 4
  - směrovač zjistí neoptimální chování koncového uzlu a „poučí ho“ o vhodnějším směru
    - skrze ICMPv6 zprávu Redirect
      - jde o součást ND (Neighbor Discovery)
- rozdíly oproti verzi 4
  - směrovač (S1), který informuje uzel (A) o existenci směrovače (S2) a cestě přes něj, vkládá do ICMPv6 zprávy Redirect také HW adresu (rozhraní) směrovače (S2)
    - aby koncovému uzlu (A) ušetřil práce s převodem
  - ICMPv6 zprávy Redirect mohou využívat autentizaci
    - aby se zabránilo zneužití/útokům

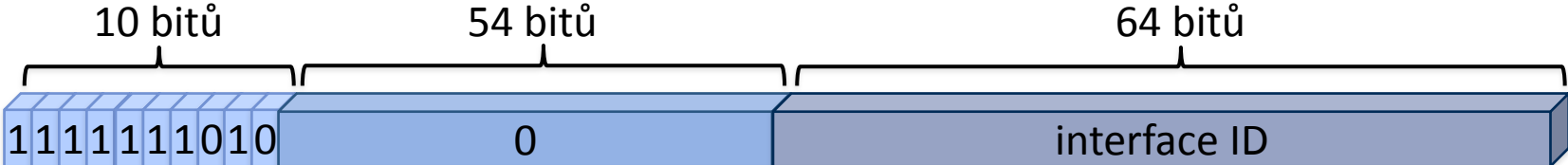


# (auto)konfigurace v IPv6

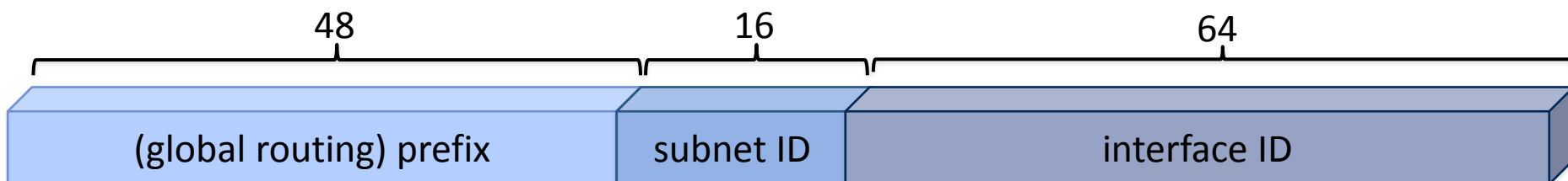
- konfigurace = získání údajů, nutných pro řádné vlastní fungování
  - IP adresa, maska/prefix sítě, adresa (prvního, implicitního) směrovače, DNS, .....
- možnosti konfigurace v IPv6:
  - tzv. **stavová konfigurace**
    - potřebné údaje jsou přiděleny vhodným serverem
      - typicky: DHCPv6 serverem
    - v zásadě stejné jako u IPv4
  - samozřejmě také manuální konfigurace
    - „ruční“ nastavení
  - tzv. **bezstavová konfigurace**
    - též: **autokonfigurace**
      - bezstavová: uzel začíná s nulovou výchozí informací
      - „auto“: uzel řeší sám ....
    - specialita IPv6, koncový uzel si dokáže sám:
      - přidělit svou lokální linkovou IPv6 adresu
        - a ověřit její unikátnost
      - zjistit přítomnost směrovačů
        - včetně síťového prefixu, .....
      - přidělit svou globální unikátní IPv6 adresu
        - na základě získaného síťového prefixu
      - nově i: zjistit si adresy DNS serverů
        - od směrovačů



# určení vlastní IPv6 adresy

- koncový uzel si vygeneruje vlastní **lokální linkovou** (link local) adresu
    - začínají prefixem **fe80:: /10**
- 
- The diagram illustrates the structure of a link-local IPv6 address. It is a 128-bit address divided into three segments:
- 10 bitů:** The prefix '1111111010'.
  - 54 bitů:** A field of zeros.
  - 64 bitů:** The 'interface ID'.
- pro adresu rozhraní (Interface ID) použije svou HW adresu
    - případně nějaký jiný postup (použije nějaký „token“)
  - uzel si otestuje jednoznačnost vygenerované lokální linkové adresy
    - použije k tomu postup **Duplicate Address Detection**, který je součástí mechanismů ND (Neighbor Discovery)
      - v zásadě: snaží se najít uzel, který by takovouto adresu již měl
  - pokud je vygenerovaná IPv6 adresa unikátní, uzel ji začne používat
    - jen v dané síti, protože je „lokální“ a platná jen pro danou síť
      - může jí využít pro oslovení směrovačů
        - pasivní: čeká na jejich ICMPv6 zprávy Router Advertisement
        - aktivní: vygeneruje ICMPv6 zprávy Router Solicitation

- z informací/odpovědí směrovačů se koncový uzel dozví:
  - zda je k dispozici stavová konfigurace
    - zda je v síti k dispozici DHCPv6 server, který by mu přidělil konfigurační údaje
      - pak využije tuto možnost
    - nebo zda má pokračovat v bezstavové (auto)konfiguraci
  - jak má směřovat
    - uzel se dozví o existenci směrovačů a o cestách, které přes ně vedou
    - naplní si své vyrovnávací paměti
      - Destination Cache, Prefix Cache, Default Router Cache
    - doplní si svou směrovací tabulku
      - i skrze ICMPv6 zprávy Redirect
  - v jaké síti se nachází
    - dozví se (globální směrovací) síťový prefix i identifikátor podsítě (subnet ID)
      - a na základě toho si může sám přidělit svou globální unikátní IPv6 adresu





- součástí konfigurace musí být i nastavení IP adres DNS serverů
  - původně: nebylo to součástí autokonfigurace
    - muselo se řešit stavovou konfigurací (přidělením od DHCP serveru) – nelogické
  - nově (RFC 6106): lze řešit i v rámci (bezstavové) autokonfigurace

- řešení:

- konfigurační informace o DNS poskytují směrovače
  - v rámci ICMPv6 zpráv **Router Advertisement**
    - samy nebo v reakci na zprávu Router Solicitation
- skrze 2 nové doplňky (options), které se vkládají do těchto ICMPv6 zpráv
  - **RDNSS (Recursive DNS Server)**
    - doplněk obsahující seznam IPv6 adres DNS serverů (1 nebo více), včetně jejich životnosti
  - **DNSL (DNS Search List)**
    - doplněk obsahující seznam DNS přípon (suffix-ů), které lze přidávat za symbolická doménová jména při kladení dotazů

