

Kapitola 7

Směrování

Směrování IP datagramů (*IP routing*) a předávání IP datagramů (*IP forwarding*) jsou dva procesy, na kterých Internet stojí.

Základní schéma směrování je zobrazeno na obr. 7.1. Představme si, že jsme právě na obr. 7.1 přijali linkový rámec nesoucí IP datagram pomocí prvního síťového rozhraní. Jeho zpracování pak prochází následujícími kroky:

1. IP datagram je vybalen z linkového rámce a předán do vstupní fronty ke zpracování. Linková adresa + IP adresa odesílatele mohou být uloženy do paměti ARP pro další využití.
2. IP datagram je vybrán ze vstupní fronty ke zpracování.
3. Zpracování začíná vyhodnocením volitelných položek záhlaví IP datagramu. Začíná se zpracováním volitelných položek Explicitní směrování. V případě, že IP datagram byl pouze explicitně odkloněn přes tento systém, je přímo předán subsystému směrování (7) k odeslání dále.
4. Nyní se zjišťuje, je-li IP datagram adresován této stanici. V případě, že nikoliv, pak je rovněž předán subsystému směrování (7) k odeslání dále. (Klasickým případem, kdy IP datagram není adresován této stanici, je, když tato stanice slouží jako směrovač.)
5. Nyní se zkoumá, nenese-li IP datagram zprávu ICMP. V takovém případě se IP datagram předá subsystému ICMP pracujícímu na IP vrstvě ke zpracování (5). Obdobně se postupuje i v případě IGMP-paketů či paketů směrovacích protokolů. To již však není na obr. 7.1 znázorněno.
6. Nyní jsme již vyčerpali možnosti zpracování IP datagramu na vrstvě IP, tj. IP datagram v sobě nese buď TCP segment, nebo UDP datagram. Ty jsou vybaleny z IP datagramu a předány vyšší vrstvě ke zpracování (6).
7. Všechny odchozí IP datagramy jsou nejprve předány subsystému Směrování (7). Cílem tohoto subsystému je nalézt síťové rozhraní, kterým bude IP datagram zabalený do linkového rámce odeslán. K tomu subsystému Směrování pomáhají Směrovací tabulky.
8. Nyní se zkoumá, nejedná-li se o oběžník, který je též adresován této stanici. Pokud ano, pak se kopie IP datagramu též předá do vstupní fronty (2).
9. Jedná-li se o oběžník, pak se přejde k bodu 11. V opačném případě se přejde k bodu 10.
10. Nyní se zkoumá, není-li cílová adresa adresou této stanice. Pokud ano, pak se IP datagramu též předá do vstupní fronty (2).
11. Jelikož se jedná o oběžník, proběhne jeho přímé mapování na linkovou adresu a odeslání do sítě.
12. Jelikož se jedná o jednoznačnou IP adresu příjemce (*unicast*), tak se pomocí ARP zjistí adresa linkového příjemce, IP datagram se zabalí do linkového rámce a odešle do sítě.

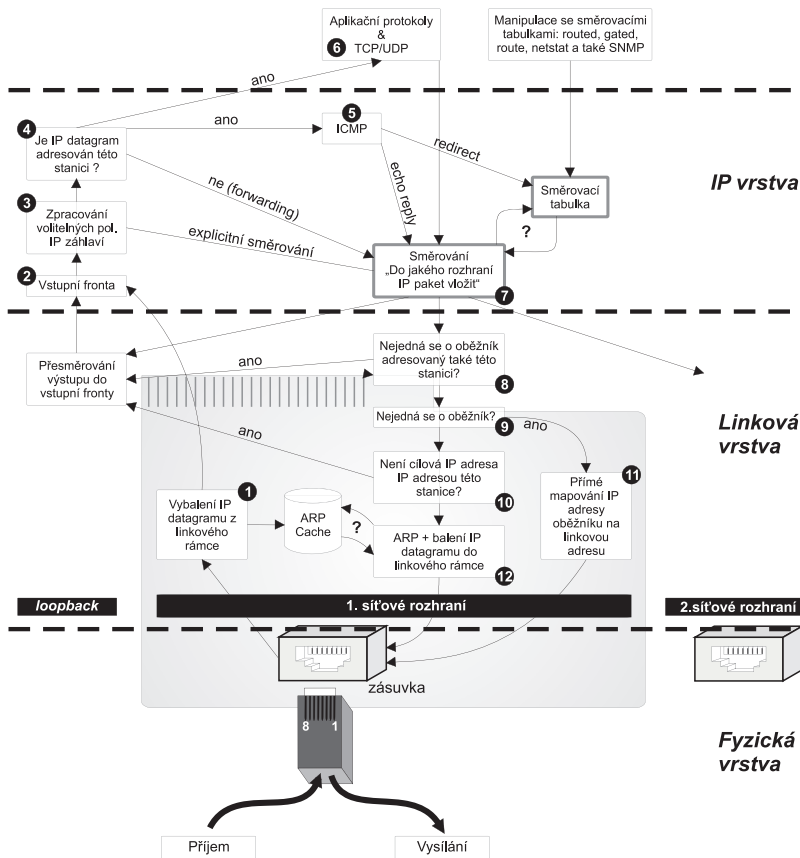


Poznámka: Prohlédnete-li si obrázek 7.1 podrobně, pak naleznete i odpověď na otázku: „A proč musím konfigurovat programovou smyčku (loopback)?“ Odpověď je jasná: „Vždy když nějaké rozhraní zjistí, že by se něco mělo předat zpět na vstup, tak se to předá na programovou smyčku, která to zařídí.“

Z obrázku 7.1 je také patrné, že při zpracování vstupů v některých případech operační systém informace automaticky předává na výstup (subsystému Směrování), tj. aplikační programy do tohoto předávání nezasahují. Jedná se zejména o:

- ◆ explicitní směrování (*source routing*),
- ◆ předávání (*forwarding*),
- ◆ požadavek o ICMP echo (*echo request*),
- ◆ přesměrování (*redirect*).

Operační systémy mají v jádře vždy nějaké parametry, kterými lze takováto automatická zpracování IP datagramů zakázat. Velice častý je např. zákaz explicitního směrování, naopak zpracování požadavku o echo se zakazuje méně často.



Obrázek 7.1: Směrování

Předávání (*forwarding*) a filtrace (*filtering*)

Předávání umožňuje stanici pracovat jako směrovač. Pokud stanice zjistí, že IP datagram není adresován pro ni, pak se jej pokouší předat dále, tj. odeslat jako odesílat své IP datagramy.

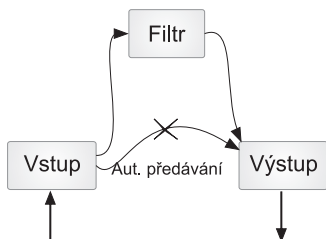
Předávání lze i zakázat – to bývá volbou jádra operačního systému. U starších systémů bylo nutné pro takový zákaz znovu sestavit jádro operačního systému. U dnešních systémů je to možné provádět dynamicky. Někdy je však nutné systém po takové změně restartovat.

Zajímavá je situace od Windows 2000 výše (tentokrát nemíníme Windows servery). Ve Windows totiž nastavujeme předávání vložení hodnoty 1 do klíče `IpEnableRouter`, který je uložen v registru `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`.

Zajímavou vlastností mnohých operačních systémů je, že IP datagramy nepředávají mechanicky, ale provádějí filtraci (*filtering* nebo též *screening*), tj. nepředávají všechny pakety, ale jen některé – pro-lustrované. Většinou filtrace pracuje tak, že před tím, než je IP datagram předán, tak se celý proces předávání pozastaví a rozhodnutí, zdali IP datagram předat, se ponechá na procesu (službě) běžícím na pozadí.

Předávaný IP datagram se předá filtračnímu procesu (obr. 7.2), který buď předání schválí, nebo zamítne. Filtrační proces se rozhoduje na základě informací v:

1. IP záhlaví, např. není-li adresát nebo příjemce na černé listině,
2. TCP záhlaví, např. podle čísel portu a nastavených příznaků ACK či SYN,
3. aplikačním protokolu, což používají firewally atp.

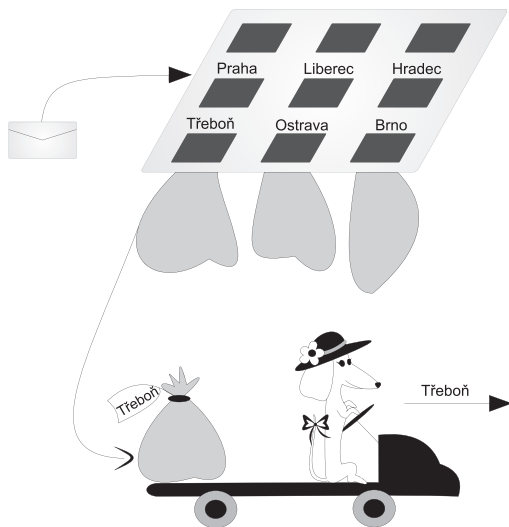


Obrázek 7.2: Filtrace

Směrování (*routing*)

Směrování IP datagramů je velice podobné třídění dopisů na poště. Na poště mají třídící stůl s vyřezanými otvory. Pod každým otvorem je přivázán poštovní pytel. Nad otvorem jsou napsány názvy měst, kam je z místní pošty přímé poštovní spojení.

Třídění probíhá tak, že poštovní úředník bere dopis za dopisem. Na každém dopisu si prohlédne adresu. Je-li adresát z Brna, pak dopis vhodí do otvoru Brno. Je-li adresát z Roztok u Prahy, pak dopis vhodí do otvoru Praha (protože do Roztok není přímé poštovní spojení, to je nejbližší Roztokům do Prahy). Až poštovní úředník vytrídí všechny dopisy, pak pytel po pytli odváže z třídícího stolu. Každý pytel zaváže a přiváže k němu visačku, na kterou napíše název města, kam se má pytel odeslat. Poté se pytel naloží...

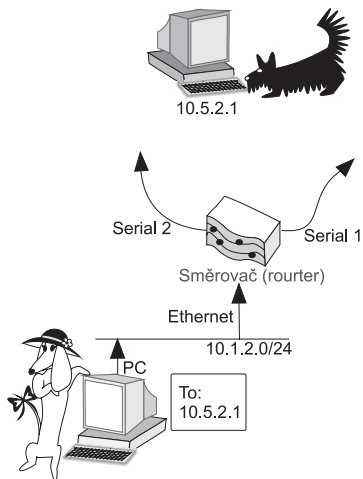


Obrázek 7.3: Třídění na poště

Směrovač netřídí dopisy, ale IP datagramy. Tento proces se nazývá směrováním. Směrovač obdrží IP datagram a musí rozhodnout, do kterého svého rozhraní jej má vhodit, kterému svému sousedovi (*next hop*) jej má poslat.

Zjednodušeně řečeno: směrovač je zařízení, které předává IP datagramy z jednoho svého rozhraní do jiného rozhraní. Směrovač umí předat IP datagram i do téhož rozhraní, ze kterého IP datagram přišel. Považuje to však ze výstřednost, takže o tom odesílatele IP datagramu upozorní ICMP pakem „*redirect*“.

Na obrázku 7.4 směrovač obdržel IP datagram adresovaný stanici 10.5.2.1 a musí rozhodnout, zdali jej vložit do rozhraní Serial1, Serial2 nebo snad zpět do rozhraní Ethernet.



Obrázek 7.4: Dilema směrovače: „Do kterého rozhraní IP datagram vložit?“

Směrovači k rozhodování slouží směrovací tabulka (obdoba třídícího stolu na poště). Náš směrovač z obr. 7.4 má následující obsah směrovací tabulky:

Sít	Maska	Next Hop	Síťové rozhraní	Metrika
192.168.1.0	255.255.255.0	192.168.254.5	Seriál 1	4
10.1.2.0	255.255.255.0	Lokální rozhraní	Ethernet	0
10.5.1.0	255.255.255.0	10.10.10.2	Seriál 2	3
10.5.0.0	255.255.0.0	10.5.5.5	Seriál 1	2
...				
0.0.0.0	0.0.0.0	10.10.10.2	Seriál 2	1

Směrovací tabulka má v prvním sloupci IP adresu cílové sítě. Představme si pro jednoduchost, že směrovací tabulka je podle prvního sloupce tříděna sestupně. To nám umožní snadno aplikovat základní pravidlo směrování: **Více specifická adresa cílové sítě má přednost před méně specifickou.** Více specifickou adresou sítě se rozumí adresa, která má v síťové masce více jedniček. V případě, že by se ve směrovací tabulce našly dvě či více cest k cíli, pak se zvolí více specifická cesta. V případě, že se najdou dvě stejně specifické cesty, pak se zvolí cesta s nejnižší metrikou (cenou).

Zpracování

V případě, že jsou řádky směrovací tabulky tříděny sestupně, pak stačí směrovací tabulku procházet od shora dolů. Na každém řádku se vezme síťová maska, kterou se bit po bitu vynásobí IP adresa příjemce IP datagramu. Výsledek se porovná s prvním sloupcem. Pokud se výsledek nerovná IP adrese sítě v prvním sloupci, pak se přejde na zpracování následujícího řádku. Pokud se výsledek shoduje s IP adresou v prvním sloupci, pak se ještě otestuje následující řádek, zdali ve směrovací tabulce neexistuje k cíli ještě jiná cesta, (pak by vstoupila do hry metrika).

Vraťme se k příkladu z obr. 7.4. Směrovač je postaven před rozhodnutí, kterým svým síťovým rozhraním IP datagram o adrese 10.5.2.1 odeslat. Postupně shora dolů prochází směrovací tabulku:

1. řádek:

192.168.1.0	255.255.255.0	192.168.254.5	Seriál 1	4
-------------	---------------	---------------	----------	---

Vynásobíme-li bit po bitu cílovou adresu 10.5.2.1 s maskou 255.255.255.0, obdržíme 10.5.2.0, což se nerovná IP adrese sítě v prvním sloupci (ta je 192.168.1.0). Přecházíme na vyhodnocení následujícího řádku.

2. řádek:

10.1.2.0	255.255.255.0	Lokální rozhraní	Ethernet	0
----------	---------------	------------------	----------	---

Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.255.0 obdržíme 10.5.2.0, což se nerovná IP adrese sítě v prvním sloupci (ta je 10.1.2.0). Přecházíme na vyhodnocení následujícího řádku.

3. řádek:

10.5.1.0	255.255.255.0	10.10.10.2	Seriál 2	3
----------	---------------	------------	----------	---

Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.255.0 obdržíme 10.5.2.0, což se nerovná IP adrese sítě v prvním sloupci (ta je 10.5.1.0). Přecházíme na vyhodnocení následujícího řádku.

4. řádek:

10.5.0.0	255.255.0.0	10.5.5.5	Seriál 1	2
----------	-------------	----------	----------	---

Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.0.0 obdržíme 10.5.0.0, což **se rovná** IP adrese sítě v prvním sloupci (ta je 10.5.0.0). Budeme proto náš IP datagram vkládat do rozhraní Serial 1 a předávat jej dalšímu směrovači o IP adrese 10.5.5.5. Pokud by se nejednalo o sériovou linku, ale např. o Ethernet, pak by bylo třeba zjistit linkovou adresu směrovače o IP adrese 10.5.5.5 protokolem ARP.

Poslední řádek obsahující v prvním sloupci 0.0.0.0 s maskou 0.0.0.0 se nazývá **default**. Tímto implicitním směrem jsou pak odesílány všechny IP datagramy, pro které nevyhovoval žádný řádek směrovací tabulky (všimněte si, že vyhovuje každé IP adrese: nula krát nula je nula). Implicitní směr ve směrovací tabulce může a nemusí být – závisí to na správci, jak tabulku naplnil. Implicitní směr používají např. firmy pro cestu do Internetu.

S implicitním směrem se setkáváme i na silnici. Když jedu z Budějovic do Prahy, tak implicitní směr je do Prahy, ale na mnohých křižovatkách je značeno jen odbočení. Je tam šipka do Třeboně či do Bechyně, ale mnohdy chybí přímý směr do Prahy. Implicitně každý ví, že tahle silnice vede do Prahy (tj. *default* je do Prahy), tak to přece není třeba stále na každé mezi opakovat.

Manipulace se směrovacími tabulkami

Směrovací tabulku je třeba jednotlivými položkami naplnit. Položky jsou pak v tabulce trvale, dokud je někdo nezruší nebo nevypne systém. Pokud je plní směrovací aplikační protokoly, pak je sledována doba jejich života, po které jsou z tabulky vypuštěny.

V příkazech se anglicky často nepoužívá slovo *router*, ale *gateway*. Setkáváme se s tím zejména ve starší literatuře. Ve směrovací tabulce se tím rozumí následující směrovač (*next hop*).

Výpis obsahu směrovací tabulky

Ve většině operačních systémů se obsah směrovací tabulky vypisuje příkazem:

```
netstat -r
```

Totéž lze zpravidla vypsat i příkazem:

```
route print
```

Výpis obsahu směrovací tabulky v UNIXu/Linuxu

Položky směrovací tabulky jsou zde příkazem „`netstat -r`“ vypisovány sestupně:

```
# netstat -rn
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Iface
192.168.1.0	192.168.254.5	255.255.255.0	UG	ttyS1
10.1.2.0	*	255.255.255.0	U	eth0
10.5.1.0	10.10.10.2	255.255.255.0	UG	ttyS2
10.5.0.0	10.5.5.5	255.255.0.0	UG	ttyS1
127.0.0.0	*	255.0.0.0	U	lo
0.0.0.0	10.10.10.2	0.0.0.0	UG	ttyS2

Sloupec *Iface* specifikuje síťové rozhraní, do kterého se budou IP datagramy předávat. Zajímavý je ale sloupec s příznaky (*Flags*), které mohou nabývat mj. následujících hodnot:

- ◆ U (*up*). Směr je dostupný.
- ◆ G (*gateway*). Příznak G určuje, že cesta k cílové síti vede přes směrovač, tj. next hop je směrovač. Linková vrstva bude hledat linkovou adresu uvedeného směrovače, nikoliv přímo adresáta (ten není přímo dostupný).
- ◆ H (*host*). Příznak H určuje, že se jedná o adresu počítače, nikoliv o adresu sítě, tj. maska je 255.255.255.255.
- ◆ D (*dynamically*). Položka byla vytvořena dynamicky směrovacím procesem (démonem) nebo na základě ICMP zprávy redirect.
- ◆ M (*modified*). Položka byla modifikována směrovacím procesem (démonem) nebo na základě ICMP zprávy redirect.
- ◆ ! (reject). Zakázaný směr.

Výpis v operačních systémech Microsoft

Příkaz „netstat -r“ zde vypisuje obsah směrovací tabulky setříděný vzestupně, takže pokud chcete vyhodnocovat tabulku, jak jsem popsali, pak ji musíte procházet zdola nahoru. Trochu nezvyklé je, že síťová rozhraní (*interface*) se jmenují svou IP adresou, ale po chvíli si na to zvyknete. Výpis směrovacích tabulek v MS Vista se skládá až ze tří částí:

1. Seznam síťových rozhraní. Každé rozhraní má své číslo, které je pak možné použít v některých příkazech operačního systému.
2. Směrovací tabulku pro IP protokol verze 4.
3. V případě instalovaného IPv6 vypíše i směrovací tabulku pro IP protokol verze 6.

```
C:\> route print
```

```
=====
Seznam rozhraní
11 ...00 19 d2 29 51 53 ..... Intel(R) PRO/Wireless 3945ABG Network Connection
 8 ...00 16 36 fe f3 b6 ..... Intel(R) PRO/1000 PM Network Connection
 1 ..... Software Loopback Interface 1
20 ...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter
12 ...00 00 00 00 00 00 00 e0 isatap.{A4A09537-5ED6-4A75-8CB7-74401D56DEA1}
10 ...02 00 54 55 4e 01 ..... Teredo Tunneling Pseudo-Interface
=====
```

IPv4 Směrovací tabulka

Aktivní směrování:

Cíl v síti	Síťová maska	Brána	Rozhraní	Metrika
0.0.0.0	0.0.0.0	10.0.0.138	10.0.0.2	25
10.0.0.0	255.255.255.0	Propojené	10.0.0.2	281
10.0.0.2	255.255.255.255	Propojené	10.0.0.2	281
10.0.0.255	255.255.255.255	Propojené	10.0.0.2	281
127.0.0.0	255.0.0.0	Propojené	127.0.0.1	306
127.0.0.1	255.255.255.255	Propojené	127.0.0.1	306
127.255.255.255	255.255.255.255	Propojené	127.0.0.1	306
224.0.0.0	240.0.0.0	Propojené	127.0.0.1	306
224.0.0.0	240.0.0.0	Propojené	10.0.0.2	281
255.255.255.255	255.255.255.255	Propojené	127.0.0.1	306
255.255.255.255	255.255.255.255	Propojené	10.0.0.2	281

Trvalé trasy:

Síťová adresa	Maska	Adresa brány	Metrika
172.17.0.0	255.255.0.0	10.0.0.138	1

IPv6 Směrovací tabulka

Aktivní směrování:

Rozhraní	Metrika	Cíl v síti	Brána
1	306	::1/128	Propojené
11	281	fe80::/64	Propojené
12	286	fe80::5efe:10.0.0.2/128	Propojené
11	281	fe80::594d:e8e9:987e:3c80/128	Propojené
1	306	ff00::/8	Propojené
11	281	ff00::/8	Propojené

Trvalé trasy:

Žádné

Síť 224.0.0.0 s maskou 224.0.0.0 označuje všechny adresné oběžníky (včetně rezervy IP adres, tj. IP -adresy tříd D a E).

Výpis obsahu směrovací tabulky na směrovačích CISCO

Na směrovačích CISCO je možné vypsát obsah směrovací tabulky i s neprivilegovaným uživatelem:

```
Router>sho ip route
```

```
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
```

```
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
```

```
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
```

```
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
```

```
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
```

```
U - per-user static route, o - ODR
```

```
Gateway of last resort is 195.47.37.192 to network 0.0.0.0
```

```

    195.47.37.0/27 is subnetted, 1 subnets
C       195.47.37.192 is directly connected, Ethernet0
    10.0.0.0/24 is subnetted, 3 subnets
S       10.4.6.0 [1/0] via 195.47.37.211
S       10.4.4.0 [1/0] via 195.47.37.210
S       10.4.5.0 [1/0] via 195.47.37.210
S*    0.0.0.0/0 [1/0] via 195.47.37.192

```

Dolní část výpisu obsahuje jednotlivé položky směrovací tabulky. Výpis položek směrovací tabulky je rozdělen do sekcí podle jednotlivých sítí. Např. nadpis „10.0.0.0/24 is subnetted, 3 subnets“ nám uvozuje desítkové síť a navíc nám sděluje, že desítková síť je rozdělena na tři subsítě. Velice důležitou informací je první sloupec ve výpisu položky směrovací tabulky. Ten sděluje, jakým způsobem se položka do směrovací tabulky dostala. V našem výpisu máme pouze dva případy:

- ◆ C – položka se do směrovací tabulky dostala z konfigurace rozhraní směrovače.
- ◆ S – položka byla staticky konfigurována atd. Všechny možnosti jsou mj. uvedeny v první části výpisu.

Naplnění směrovací tabulky a rušení položek

Směrovací tabulka se plní:

- ◆ Při konfiguraci síťového rozhraní, kdy říkáme, jakou má síťové rozhraní adresu a masku. V operačním systému UNIX se jedná o příkaz *ifconfig*.
- ◆ Staticky (ručně) příkazem *route*.
- ◆ Dynamicky z ICMP zpráv *redirect*.
- ◆ Dynamicky směrovacími protokoly.

Staticky se směrovací tabulka plní pomocí příkazu *route*. Ve Windows má příkaz *route* následující syntaxi:

```
ROUTE [-f/-p] [command [destination] [MASK netmask] [gateway] [METRIC metric]]
```

-f	Vymaže nejprve obsah směrovací tabulky.
-p	U příkazu ADD zajistí, aby takto přidaná položka zůstala ve směrovací tabulce i po restartu PC, tj. stala se trvalou položkou. U příkazu PRINT způsobí, že se vypíše trvalé položky.
command	Určuje příkaz pro manipulaci se směrovací tabulkou, nabývá následujících hodnot:
PRINT	Vypis obsah směrovací tabulky.
ADD	Přidej položku do směrovací tabulky.
DELETE	Zruš položku ve směrovací tabulce.
CHANGE	Změň položku.
destination	Specifikuje cílovou síť.
netmask	Specifikuje síťovou masku.
gateway	Specifikuje next hop.
METRIC	Specifikuje metriku.

Příklad Windows

```
route -p add 10.0.0.32 mask 255.255.255.240 192.168.1.2
```

Příklad Unix

V operačním systému UNIX je příkaz `route` podstatně bohatší. Nejsou zde trvalé položky směrovací tabulky – po restartu se statické položky do směrovací tabulky vždy plní příkazem `route` (buť automaticky nějakou procedurou).

V operačním systému UNIX je i jiný repertoár příkazů pro manipulaci se směrovací tabulkou. Nebývá zde příkaz `PRINT`, ale naopak bývá příkaz `flush` (vymazání směrovací tabulky) a někdy též příkaz `monitor`, který způsobí živé vypisování změn ve směrovací tabulce na standardní výstup (ukončuje se se `^C`).

```
route add -net 10.0.0.32/28 192.168.1.2
```

Příklad CISCO

Ve směrovačích CISCO staticky přidáváme položky do směrovací tabulky tak, že položku přidáme do konfigurace směrovače. Např.:

```
ip route 10.1.1.0 255.255.255.0 192.169.1.1
```

vloží položku, která bude síť 10.1.1.0/24 směrovat na směrovač (gateway) 192.169.1.1. Obdobně přidáme i default směřující vše ostatní na směrovač 192.168.1.2:

```
ip route 0.0.0.0 0.0.0.0 192.168.1.2
```

Směrovací protokoly

Směrovací protokoly slouží směrovačům, aby si vzájemnou komunikací mezi sebou automaticky naplnily a udržovaly směrovací tabulky.

Máme dvojí, na sobě nezávislé, dělení směrovacích protokolů:

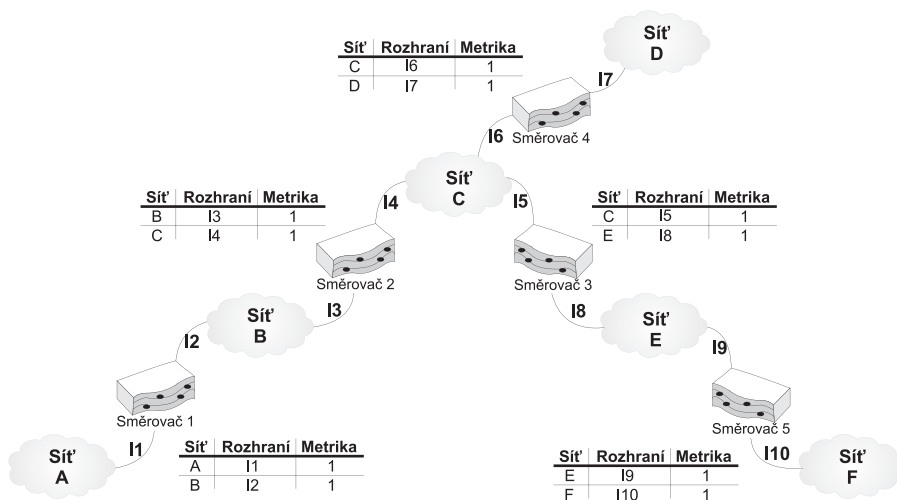
- ♦ Na *Link State Protocols (LSP)* a na *Routing Vector Protocols (RVP)* z hlediska použitého algoritmu.
- ♦ Na *Interior Gateway Protocols (IGP)* a *Exterior Gateway Protocols (EGP)* z hlediska organizačního. IGP jsou určeny pro řešení směrování v rámci autonomního systému (AS). EGP pak pro výměnu směrovacích informací mezi AS.

Většina směrovacích protokolů je určena buď pro IPv4, nebo pro IPv6. Existují však i směrovací protokoly, které umí současně podporovat více síťových protokolů. Jelikož mají protokoly IPv4 a IPv6 každý jinou délku síťové adresy, tak zpravidla každý z nich má i samostatné směrovací tabulky. Na druhou stranu princip dvojice IP adresa + síťová maska je jak v IPv4, tak i v IPv6 stejný. Takže základní principy zůstávají stejné pro oba dva protokoly. Pokud tedy pochopíte princip směrování pro IPv4, pak získáte i základ pro pochopení principu směrování pro IPv6.

Princip Routing Vector Protocols (RVP)

Princip RVP je jednoduchý. Každý směrovač pravidelně odesílá oběžníkem obsah svých směrovacích tabulek. Sousední směrovače si tuto informaci vzájemně přečtou a promítnou ji do svých směrovacích tabulek.

Nejlépe se to demonstruje na příkladu. Představme si, že právě byly zapnuty všechny směrovače na obr. 7.5. Po zapnutí si směrovače naplní své směrovací tabulky ze své startovací konfigurace.



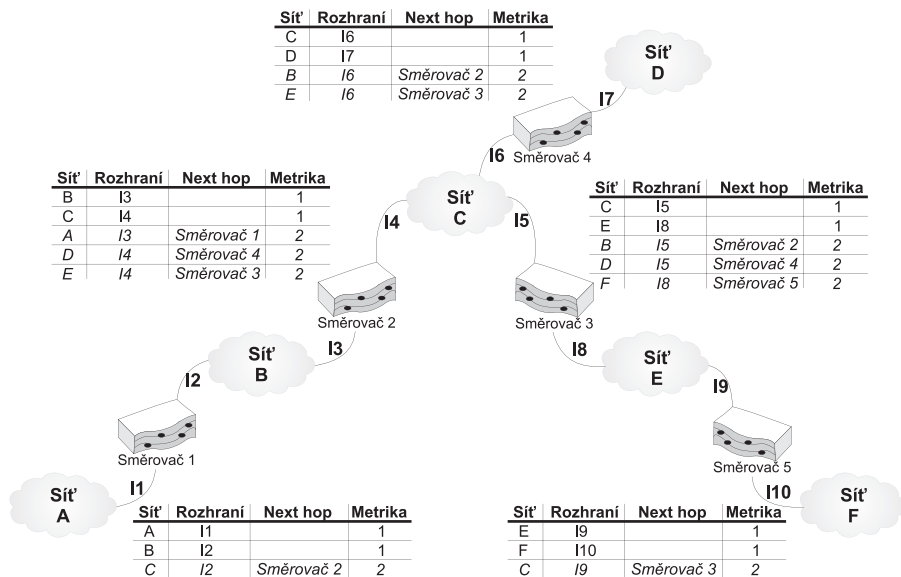
Obrázek 7.5: První krok plnění směrovacích tabulek (zapnutí směrovačů a načtení jejich konfigurace)

Ve druhém kroku (obr. 7.6) si směrovače vzájemně vymění své směrovací tabulky. Sousední směrovací tabulku pak zpracovávají v následujících krocích:

- ◆ Vyhledávají cesty, které buď ve vlastní směrovací tabulce nemají, nebo je mají s menší metrikou.
- ◆ Takto vyhledanou cestu si uloží do své směrovací tabulky, ale s vyšší metrikou. Např. protokoly RIP a RIP2 přičítají k metrice jedničku.
- ◆ V protokolech RVP je vždy stanovena nějaká horní hranice pro metriku. Pokud by položka dosáhla této horní hranice, pak se cesta považuje za nedosažitelnou a položka se do směrovací tabulky nezapiše. Pro mnohé je překvapivé, že tato horní mez není příliš velká. Např. u protokolů RIP a RIP2 bývá 16. Důvodem je skutečně, že při vyšší horní hranici se v praxi ukázalo, že směrovací tabulky začnou oscilovat.

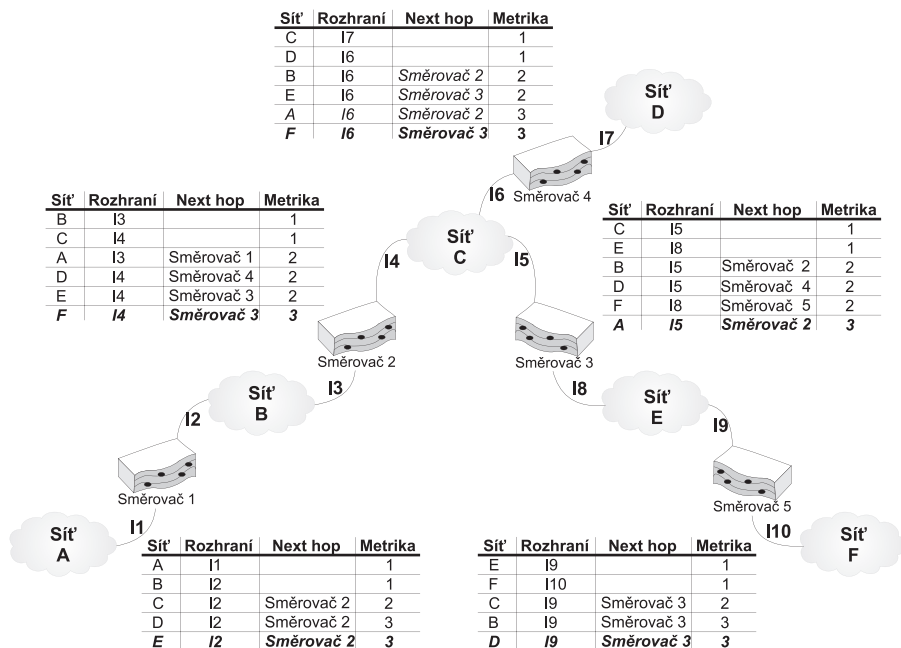
Vezměme si např. směrovací tabulku směrovače 2 z obr. 7.6. V ní jsou:

- ◆ Síť B a C, které se tam dostaly z konfigurace.
- ◆ Síť A, kterou obdržel ze směrovací tabulky směrovače 1 (zvýšil metriku na 2).
- ◆ Síť D, kterou obdržel ze směrovací tabulky směrovače 4 (zvýšil metriku na 2).
- ◆ Síť E, kterou obdržel ze směrovací tabulky směrovače 3 (zvýšil metriku na 2).



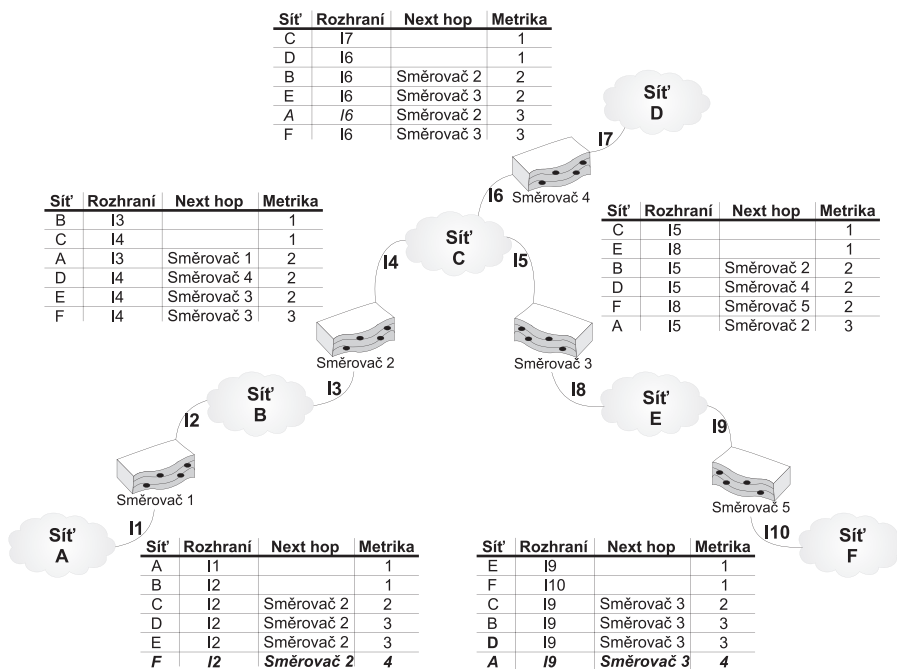
Obrázek 7.6: Druhý krok: doplnění směrovacích tabulek informacemi od sousedů

Ve třetím kroku (obr. 7.7) si směrovače vyměňují směrovací tabulky, které obsahují informace, které získaly ve druhém kroku. Na obr. 7.8 jsou to ty, které mají metriku 3.



Obrázek 7.7: Třetí krok

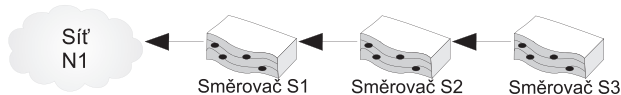
Čtvrtý krok je opět stejná písnička (obr. 7.8). Po čtvrtém kroku ale už všechny směrovače mají informace o všech sítích. Jelikož topologie sítě se může měnit, směrovače takto dynamicky získané informace ve svých směrovacích tabulkách neudržují trvale, ale zpravidla jen 2–5 minut v závislosti na konkrétním směrovacím protokolu.



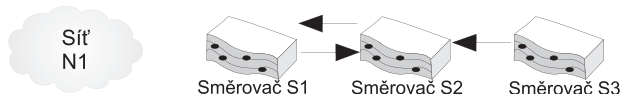
Obrázek 7.8: Čtvrtý krok

Uvedený algoritmus se někdy také označuje jako **Bellman-Fordův algoritmus**. Tento algoritmus má bohužel i některé ne příliš pěkné vlastnosti. Jednou z nich je tzv. „pomalá konvergence“: Směrovač S1 (obr. 7.9) je přímo připojen do sítě N1 s metrikou 1. Periodicky tuto informaci šíří svým sousedům. Nejprve se dozví tuto informaci směrovač S2 a vloží si ji do své směrovací tabulky s metrikou 2. Posléze si tuto informaci vloží do svých směrovacích tabulek i směrovač S3 s metrikou 3.

Směrovače S1, S2 a S3 mají spojení do sítě N1:



Spojení S1 do sítě N1 vypadne:



Obrázek 7.9: Pomalá konvergence

Nyní vypadne spojení směrovače S1 do sítě N1. Směrovač S1 si proto ve směrovací tabulce zruší cestu do sítě N1. Jenomže směrovač S2 má ve svých směrovacích tabulkách tuto cestu s metrikou 2 a v dalším kroku je nabídne směrovači S1. Směrovač S1 v dobré víře, že směrovač S2 zná cestu do sítě N1, si uloží cestu do sítě N1 přes směrovač S2, ale s metrikou 3. Nyní směrovači S2 expiruje tato položka. Jenže směrovač S1 nabízí cestu do N1 s metrikou 3, tak si ji S2 uloží s metrikou 4. Atd. Tento jev nazýváme oscilací, kterou řešíme dvěma postupy:

- ♦ Již zmíněnou horní mezí pro metriku, kdy např. protokoly RIP a RIP2 už cesty s metrikou větší než 15 považují za nedostupné.
- ♦ Technikou „odříznutí horizontu“, kdy se směrovače snaží předcházet zpětnému použití informací, které původně do sítě samy zaslaly.

Další slabinou těchto algoritmů je, že nepodporují rozdělování výkonu (*load balancing*) mezi paralelní linky.

Závěr k RVP je asi takový, že jsou určeny pro méně rozsáhlé WAN. Avšak na menších sítích je vždy otázkou, jestli je opravdu třeba dynamicky plnit směrovací tabulky, jestli nevystačíme se statickými položkami směrovacích tabulek (tj. s ručním naplněním směrovacích tabulek).

RIP, RIP2 a RIPng

Protokoly RIP, RIP2 a RIPng jsou příkladem protokolů RVP. Pro mnohé bude překvapením, že se jedná o aplikační protokoly – své pakety totiž balí do UDP datagramů. Pakety pak obsahují celé nebo části směrovacích tabulek. V Linuxu se tyto protokoly aktivují pomocí směrovacího manažeru *zebra*. Na serverech Windows je lze aktivovat pomocí *Routing and Remote Access*.

Na počátku byl (dnes již historický) protokol RIP (*Routing Information Protocol*). Ten měl však nevýhodu v tom, že nešířil s cílovými sítěmi i jejich masku – předpokládal standardní síťové masky. Proto se do dnešní doby již nehodí. Protokol RIP2 odstranil tento nedostatek. Mezi RIP a RIP2 je tak rozdíl v tom, že RIP šířil své pakety pomocí všeobecného oběžníku (*broadcast*), kdežto RIP2 používá adresný oběžník (*multicast*) o IP adrese 224.0.0.9.

Takto jsou pakety šířeny každých 30 vteřin. Ve směrovacích tabulkách pak směry po 180 vteřinách expirují, tj. pokud se do 180 vteřin neobjeví znovu oběžník s konkrétním směrem, pak je tento směr vymazán ze směrovací tabulky. Metrika 16 je pokládána za nedostupnou síť.

RIPng je modifikací pro IPv6.

Protokoly RIP a RIP2 využívají dobře známý port 520/udp a protokol RIPng využívá dobře známý port 521/udp.

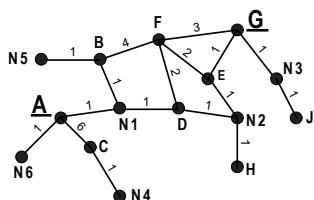
Princip *Link State Protocols* (LSP)

Princip LSP je naprosto odlišný. K pochopení si nejprve musíme udělat krátkou exkurzi do teorie grafů.

Grafem rozumíme množinu vrcholů spojených hranami. Dva vrcholy nazýváme sousedními, existuje-li mezi nimi hrana. Na obr. 7.11 jsou vrcholy A a C sousední, ale vrcholy A a B nikoliv. Pro každý vrchol je možné jednoznačně určit množinu všech jeho sousedů. Např. vrcholu N3 na obr. 7.10 přísluší množina sousedů, která obsahuje uzly G a J.

Dále jako cestu v grafu chápeme sekvenci vrcholů, kde následující uzel je vždy sousedem předchozího uzlu.

Každá hrana je ohodnocena číslem, které nazýváme metrika („délka hrany“). Např. hrana z uzlu A do uzlu C má metriku 6. Můžeme si představit, že délka cesty z uzlu A do uzlu C je 6 jednotek. Délkou celé cesty pak myslíme součet délek všech hran, kterými cesta prochází.



Obrázek 7.10: Graf

Nyní si představme, že směrovač A potřebuje odeslat IP datagram do uzlu G (obr. 7.10). Z hlediska grafu hledáme nejkratší cestu z uzlu A do uzlu G. Nejkratší cesta v grafu se řeší pomocí algoritmu, který nese jméno holandského informatika jménem Edsger Wybe Dijkstra (1930-2002). Tento algoritmus najde z daného uzlu nejkratší cestu ke všem uzlům grafu. Algoritmus však může být zastaven, je-li nalezena nejkratší cesta do požadovaného uzlu grafu.

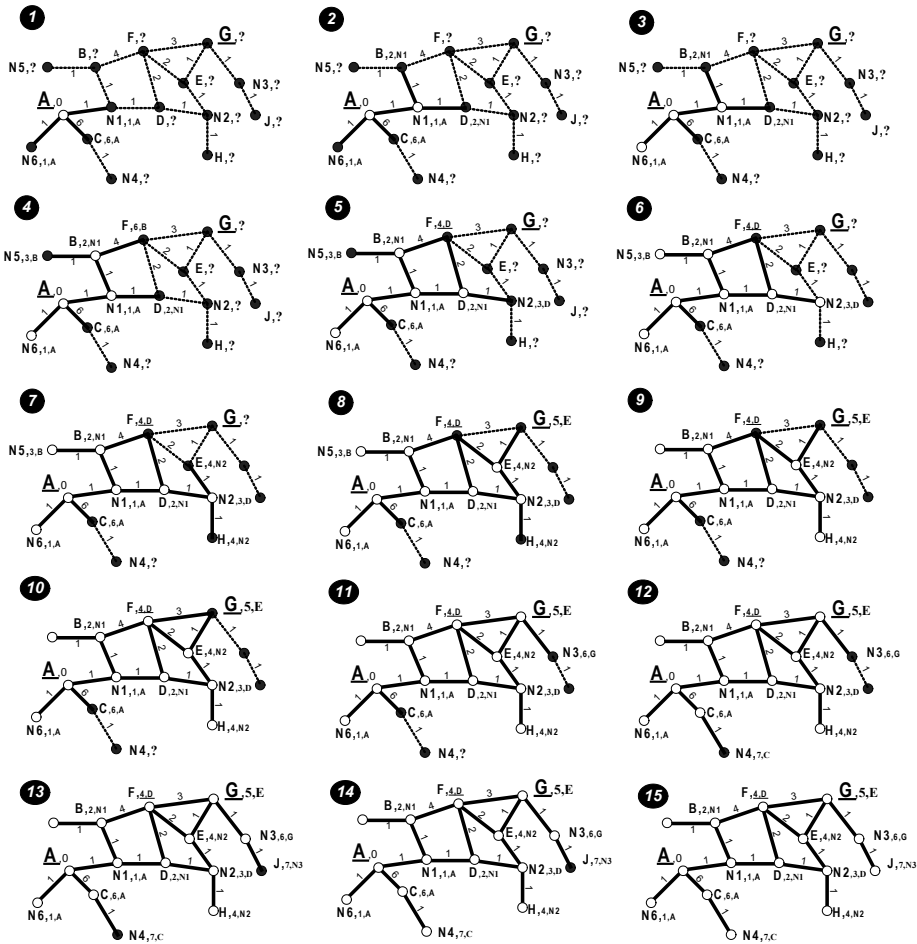
Algoritmus Nejkratší cesty

1. Vytvoříme seznam vzdáleností, seznam předchozích vrcholů, seznam navštívených vrcholů a aktuální vrchol.
2. Všechny hodnoty v seznamu vzdáleností nastavíme na nekonečno, pouze vzdálenost do výchozího (startovacího) uzlu nastavíme na 0.
3. Všechny hodnoty v seznamu navštívených uzlů nastavíme na FALSE.
4. Všechny hodnoty v seznamu předchozích vrcholů nastavíme na -1.
5. Výchozí vrchol nastavíme jako aktuální vrchol.
6. Aktuální vrchol označíme jako navštívený vrchol.
7. Pro všechny uzly, které mohou být dosaženy z aktuálního vrcholu, nastavíme hodnoty v seznamu vzdáleností a v seznamu předchozích vrcholů (pokud je kratší cesta přes vrchol, ve kterém jsme, tak se aktualizuje, jinak se hodnota ponechá).
8. Jako aktuální vrchol nastavíme ještě nenavštívený vrchol, který má nejmenší vzdálenost od startovacího vrcholu.
9. Opakujeme body 6 až 8, dokud nejsou všechny vrcholy označeny jako navštívené.



Tip: Dijkstrův algoritmus můžete použít i pro plánování nejkratší cesty na dovolené!

Nejllepší je praktická demonstrace: budeme hledat nejkratší cestu z vrcholu A do všech vrcholů grafu z obr. 7.10. Jednotlivé kroky hledání nejkratší cesty jsou rozfázovány na obr. 7.11. (sledujte též obr. 7.12, kde postupně vzniká strom nejkratší cesty):



Obrázek 7.11: Jednotlivé fáze hledání nejkratších cest z výchozího vrcholu A do všech ostatních vrcholů grafu (bílé kolečko = navštívený vrchol)

1. První krok obsahuje dvě fáze:

◆ Inicializační fáze:

- Vytvoří se a inicializují se seznamy.
- Vzdálenost z/do vrcholu A se nastaví na 0.
- Vrchol A je zvolen jako výchozí, aktuální a navštívený.

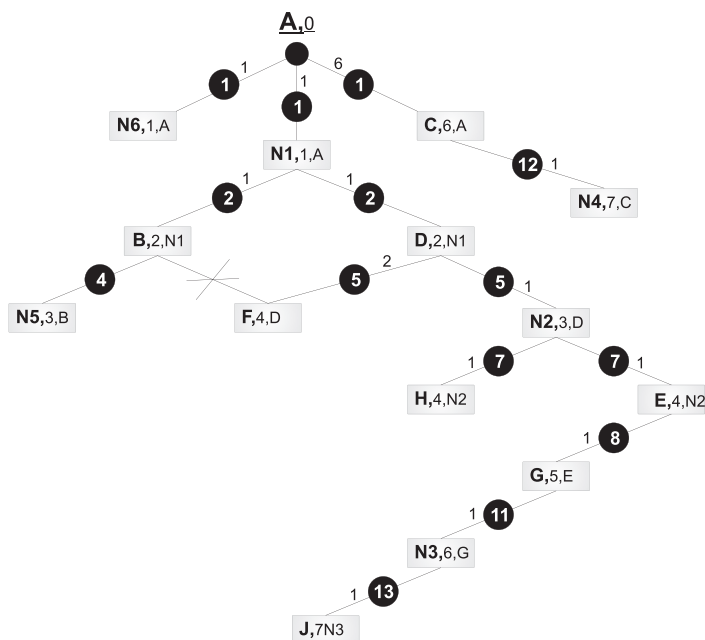
◆ První krok hledání nejkratší cesty, kdy hledáme všechny uzly, které mohou být dosaženy z vrcholu A. Jedná se o uzly:

- N6 se vzdáleností 1 od předchozího uzlu A (délka cesty od uzlu A je 1).
- C se vzdáleností 6 od předchozího uzlu A (délka cesty od uzlu A je 6).
- N1 se vzdáleností 1 od předchozího uzlu A (délka cesty od uzlu A je 1).
- Jako aktuální vrchol označíme vrchol N1 (délka cesty od uzlu A je 1).

2. Ve druhém kroku:
 - ◆ Vrchol N1 označíme jako navštívený.
 - ◆ Hledáme všechny uzly, které mohou být dosaženy z vrcholu N1. Jedná se o uzly:
 - D s délkou hrany 1 od předchozího uzlu N1 (délka cesty od uzlu A je 2).
 - B s délkou hrany 1 od předchozího uzlu N1 (délka cesty od uzlu A je 2).
 - ◆ Jako aktuální vrchol označíme vrchol N6 (délka cesty od uzlu A je 1).
3. Třetí krok nám pouze označí vrchol N6 jako navštívený a jako aktuální nastaví např. vrchol B (délka cesty od uzlu A je 2).
4. Čtvrtý krok:
 - ◆ Vrchol B označíme jako navštívený.
 - ◆ Hledáme všechny uzly, které mohou být dosaženy z vrcholu B. Jedná se o uzly:
 - N5 s délkou hrany 1 od předchozího uzlu B (délka cesty od uzlu A je 3).
 - F s délkou hrany 4 od předchozího uzlu B (délka cesty od uzlu A je 6).
 - ◆ Jako aktuální vrchol označíme D (délka cesty od uzlu A je 2).
5. Pátý krok:
 - ◆ Vrchol D označíme jako navštívený.
 - ◆ Hledáme všechny uzly, které mohou být dosaženy z vrcholu D. Jedná se o uzly:
 - F s délkou hrany 2 od předchozího uzlu D (délka cesty od uzlu A je 4) – opravíme hodnotu získanou ve 4. kroku.
 - N2 s délkou hrany 4 od předchozího uzlu D (délka cesty od uzlu A je 6).
 - ◆ Jako aktuální vrchol označíme N5 (délka cesty od uzlu A je 3).
6. Šestý krok nám pouze označí vrchol N5 jako navštívený a jako aktuální nastaví např. vrchol N2 (délka cesty od uzlu A je 3).
7. Sedmý krok :
 - ◆ Označí vrchol N2 jako navštívený.
 - ◆ Hledáme všechny uzly, které mohou být dosaženy z vrcholu N2. Jedná se o uzly:
 - H s délkou hrany 1 od předchozího uzlu N2 (délka cesty od uzlu A je 4).
 - E s délkou hrany 1 od předchozího uzlu N2 (délka cesty od uzlu A je 4).
 - ◆ Jako aktuální vrchol označíme E (délka cesty od uzlu A je 3).
8. Osmý krok:
 - ◆ Označí vrchol E jako navštívený.
 - ◆ Hledáme všechny uzly, které mohou být dosaženy z vrcholu E. Jedná se o uzly:
 - F s délkou hrany 2, ale s horší cestou od vrcholu A, takže nic neopravujeme.
 - G s délkou hrany 1 od předchozího uzlu E (délka cesty od uzlu A je 5).
 - ◆ Jako aktuální vrchol označíme E (délka cesty od uzlu A je 3).
9. Jako navštívený vrchol označí H.
10. Jako navštívený vrchol označí F.
11. Jako navštívený vrchol označí G a dosaženo bude vrcholu N3.
12. Jako navštívený vrchol označí C a dosaženo bude vrcholu N4.
13. Jako navštívený vrchol označí N3 a dosaženo bude vrcholu J.

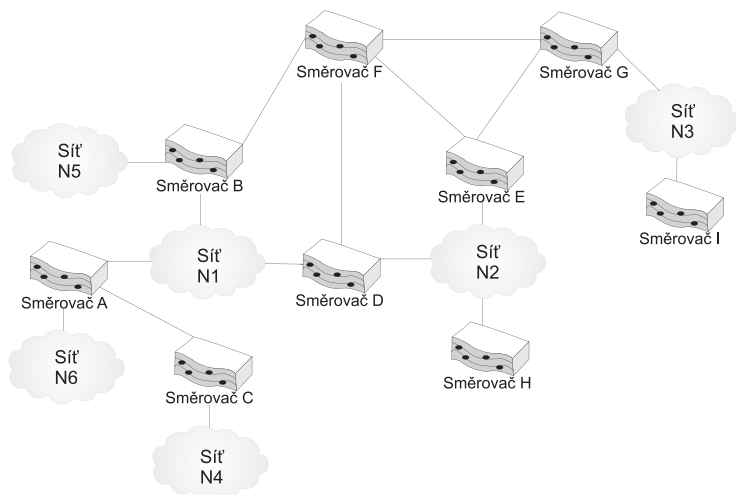
14. Jako navštívený vrchol označí N4.

15. Jako navštívený vrchol označí J.



Obrázek 7.12: Strom nejkratší cesty z vrcholu A do všech ostatních vrcholů grafu

Dosud jsme uvažovali jen obecný graf z obr. 7.10. Nyní si pod ním představíme rozlehlou síť z obr. 7.13.



Obrázek 7.13: Topologie sítě

Na základě stromu nejkratší cesty pak již snadno vytvoříme směrovací tabulku směrovače A:

Síť	Rozhraní	Next hop	Metrika
N1	I3	lokální	1
N2	I3	D	3
N3	I3	D	6
N4	I2	C	7
N5	I3	B	3
N6	I1	1	lokální

Obdobně si všechny směrovače naší sítě vytvoří své směrovací tabulky. Co k tomu potřebují? Nepotřebují nic víc, ani nic méně, než znát topologii sítě, tj. znát seznam všech vrcholů a všech hran (včetně metrik) našeho grafu.

Jak směrovač získá kompletní topologii sítě? Topologii získá ve dvou krocích:

1. V prvním kroku každý směrovač odesílá dotazy všem svým sousedům, jsou-li připojeny („pingne“ na sousedy). Jestliže směrovač v zadaném časovém intervalu obdrží alespoň k z n odpovědí od souseda ($1 \leq k \leq n$), pak považuje souseda za připojeného (*is up*). Cena (metrika) spoje k sousedovi může být buď nastavena ručně, nebo spočtena (např. z podílu k/n , z doby odezvy souseda pod.).
2. Nyní již každý směrovač ví o svých připojených sousedech, a tak v druhém kroku každý směrovač odešle informaci o svých sousedech ostatním směrovačům – směrovače „zaplavují“ sítí těmito informacemi.

Všechny směrovače nyní znají všechny vrcholy a hrany grafu, takže každý z nich si může spustit algoritmus nejkratší cesty a vytvořit své směrovací tabulky.

Jednotlivé implementace směrovacích protokolů LSP mají různé modifikace uvedeného algoritmu. Zmíňme se o dvou:

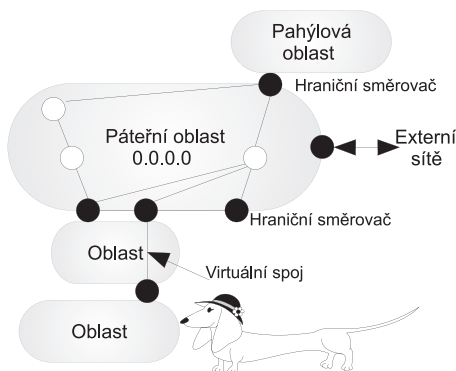
1. Není dobré zaplavovat směrovacími informacemi příliš velké sítě. Proto síť zpravidla rozdělujeme na oblasti (směrovací domény). Směrování pak řešíme v rámci těchto oblastí. Směrovače pak vyměňují směrovací informace mezi oblastmi.
2. Jestliže je na jedné společné síti více směrovačů (např. na LAN), pak se vzájemně dohodnou na jednom z nich (*designated router*), který řeší směrování na této síti jménem ostatních.

LSP-protokoly jsou podstatně stabilnější než protokoly RVP. A hlavně jsou použitelné i na velkých rozlehlých sítích. Nevýhodou je, že je nestačí na směrovačích pouze aktivovat, ale musíme mít experta, který je nakonfiguruje. Jinak je možné, že nám síť nebude chodit tak, jak očekáváme.

OSPF

Asi nejrozšířenějším příkladem protokolu LSP je protokol OSPF (*Open Shortest Path First*). OSPF implementuje řadu nových vlastností: vzájemnou autentizaci komunikujících směrovačů, load balancing na paralelních linkách o stejné metrice, podporu QoS, redistribuci směrovacích informací získaných mimo AS atd. Existuje i varianta pro protokol IPv6.

OSPF vyžaduje, aby síť byla zvláštním způsobem rozdělena do oblastí. Jádrem sítě je tzv. páteřní oblast, jejíž směrovače mají kompletní směrovací informaci. Ostatní oblasti musí být buď přímo, nebo virtuálně připojeny k páteřní oblasti.



Obrázek 7.14: Topologie sítě pro OSPF

Zvláštním případem oblasti je pahýlová oblast. Do této oblasti se nešíří směrovací informace ostatních sítí, protože cesta z této oblasti do ostatních sítí může být snadno udělána pomocí položky *default* směřující do páteřní oblasti. Aby to ale správně chodilo, tak pahýlová oblast může být k páteřní oblasti připojena jen jednou linkou v jednom bodě.

OSPF nepodporuje příčné spoje mezi oblastmi. Výjimkou je spojení oblasti skrze jinou tzv. tranzitní oblast virtuálním spojením.

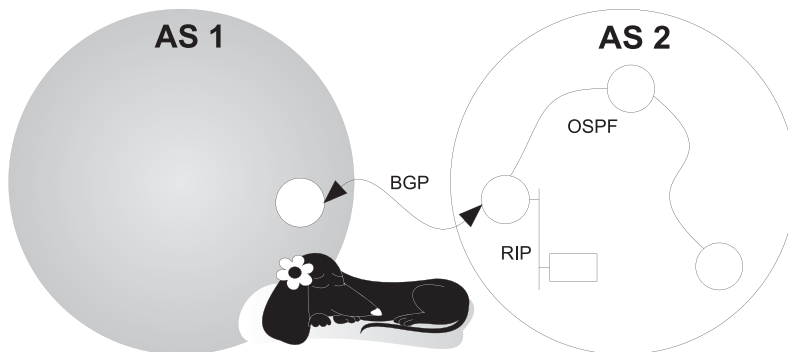
Každá oblast OSPF má svou jedinečnou čtyřbajtovou identifikaci, která se napohled podobá IP adrese (nezaměňovat!). Páteřní oblast má identifikaci 0.0.0.0. Zajímavé mj. jsou hraniční směrovače, tj. směrovače, které mají síťová rozhraní do dvou oblastí. Pak vlastně pro každou oblast jedou zvlášť algoritmus nejkratší cesty.

Redistribuce

Na obrázku 7.15 je otazníkem označen směrovač, který si vyměňuje současně směrovací informace protokoly BGP, OSPF, RIP a k tomu má možná ve směrovací tabulce několik statických položek.

Otázka je, zda se mají položky směrovací tabulky získané jedním směrovacím protokolem propagovat do ostatních směrovacích protokolů, tj. má-li se provést redistribuce. Redistribucí tak rozumíme přenos směrovacích informací získaných jedním směrovacím protokolem jinému směrovacímu protokolu.

Položka ve směrovací tabulce musí v sobě nést tedy také informaci, jakým protokolem byla vytvořena (statické a konfigurační položky tvoří samostatnou skupinu). To je ovšem pohled administrátora. Ve skutečnosti, pokud na jednom směrovači běží více směrovacích protokolů, tak každý má svou část směrovací tabulky, tj. jako bychom měli více samostatných směrovacích tabulek. Redistribuce pak je přenos informací mezi nimi.



Obrázek 7.15: Redistribuce

Domácí cvičení

Na svém počítači/serveru:

- ◆ Vypište obsah směrovacích tabulek.
- ◆ Do směrovacích tabulek vložte cestu do sítě 169.254.0.0/8 přes váš nejbližší směrovač.
- ◆ Zrušte cestu do sítě 169.254.0.0/8 vloženou v předchozím bodu.



Upozornění: Neodborné vložení/zrušení položky ve směrovacích tabulkách je jednou z nejzákladnějších chyb! Po každé manipulaci se směrovacími tabulkami se nezapomeňte přesvědčit, že máte spojení s nějakým vzdáleným serverem.