

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta informačních technologií



Dokumentace k projektu do předmětu ISA

Nástroj monitorování RIP a RIPng

21. Listopadu 2015

Autor: Jakub Stejskal, xstejs24@stud.fit.vutbr.cz
Fakulta informačních technologií
Vysoké Učení Technické v Brně

Obsah

1	Úvod	3
2	Rozbor protokolu	3
2.1	RIPv1.....	3
2.2	RIPv2.....	4
2.3	RIPng.....	5
3	Návrh a implementace	7
3.1	Odchytávač.....	7
3.2	Podvrhávač.....	7
4	Příklady užití a vyhodnocení výsledků	8
5	Závěr	9
A	Metriky kódu	10

1 Úvod

V této dokumentaci jsou shrnuty všechny potřebné informace, které byly potřeba k vytvoření aplikace pro snímání RIPv1, RIPv2 a RIPvng paketů a aplikace pro podvržení RIPv2 RESPONSE paketů.

Dále jsou zde rozebrány možnosti spuštění aplikace a vyhodnocení některých výsledků, kterých aplikace dosahuje.

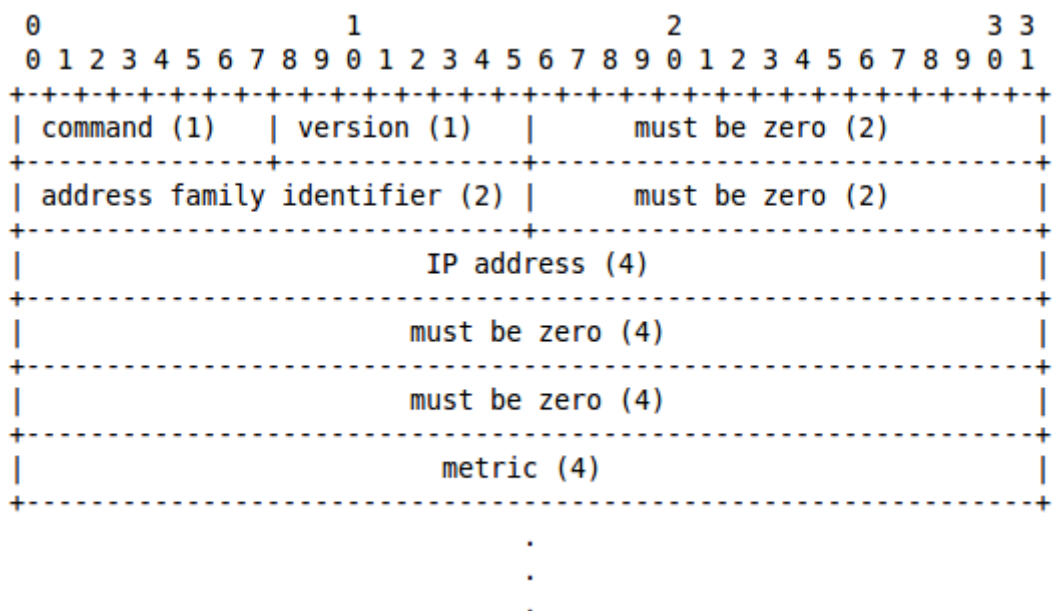
2 Rozbor RIP protokolu

RIP protokol je směrovací protokol, který funguje na základě Bellman-Fordova (distance vector) algoritmu. RIP tento algoritmus využívá pro určení nejkratší cesty v síti. Metrikou je počet skoků k cílové síti, které slouží zároveň jako ochrana proti nekonečným smyčkám v síti. Metrika může nabývat hodnot od 0 do 15, číslo 16 už značí nedosažitelnou síť, počet zařízení je tedy omezen na 16.

Pro vytvoření výsledné aplikace je potřeba znát strukturu celého datagramu, zejména jeho RIP a RIPvng části. RIP a RIPvng mají různou strukturu, ve které uchovávají přenášené informace. Tyto struktury jsou jednotlivě popsány níže. K nastudování RIP a RIPvng datagramu bylo využito *RFC1058* pro RIPv1, *RFC2456* pro RIPv2 a *RFC2080* pro RIPvng.

2.1 RIPv1

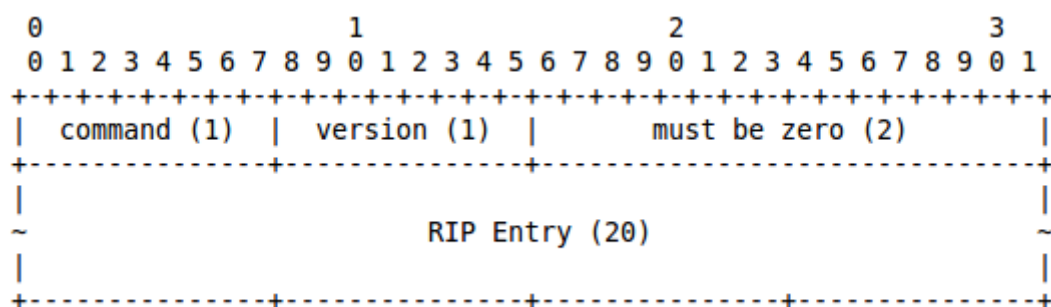
Pro správné odchycení RIPv1 datagramu je potřeba vědět, jak takový datagram vypadá. *Obrázek č. 1* zachycuje strukturu RIPv1 datagramu.



Obrázek č. 1 - RIPv1 datagram

2.2 RIPv2

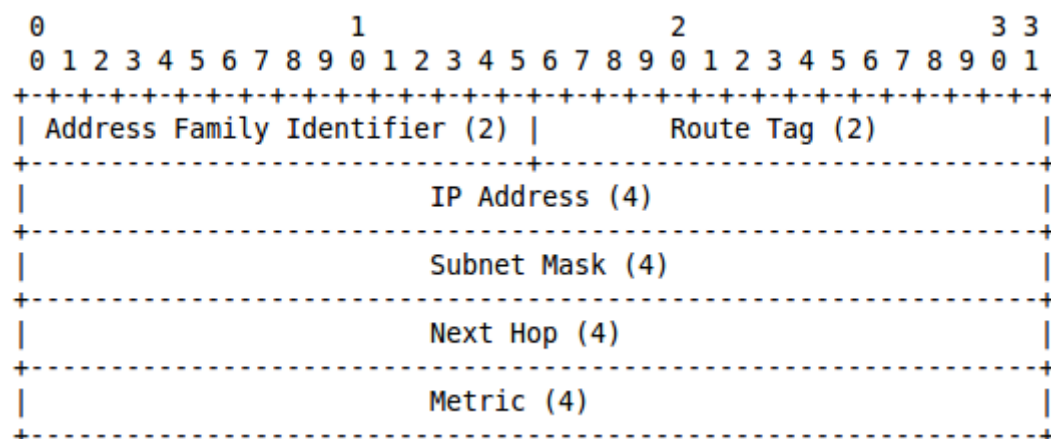
RIPv2 (Obrázek č. 2) je označení pro vylepšení RIPv1. Na rozdíl od RIPv1 umožňuje RIPv2 použít zabezpečení datagramu heslem.



Obrázek č. 2 - RIPv2 datagram

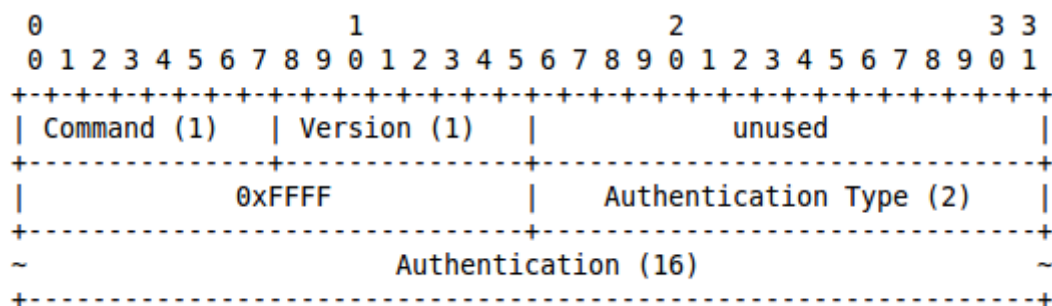
Heslo je přenášeno jako část datagramu (Obrázek č. 4). Autentizační část využívá velikost RIP Entry (Obrázek č. 3). Zda datagram obsahuje heslo se pozná podle hodnoty první "Address Family Identifier" v první RIP Entry části. Pokud je tato hodnota rovna 0xFFFF obsahuje datagram heslo a první RIP Entry je tedy autentizační část.

Položka "Authentication type" značí použití simple password. Hodnota této položky je 2. Zbýlých 16 bytů ("Authentication") zabírá samotné heslo.



Obrázek č. 3 - RIPv2 Entry datagramu

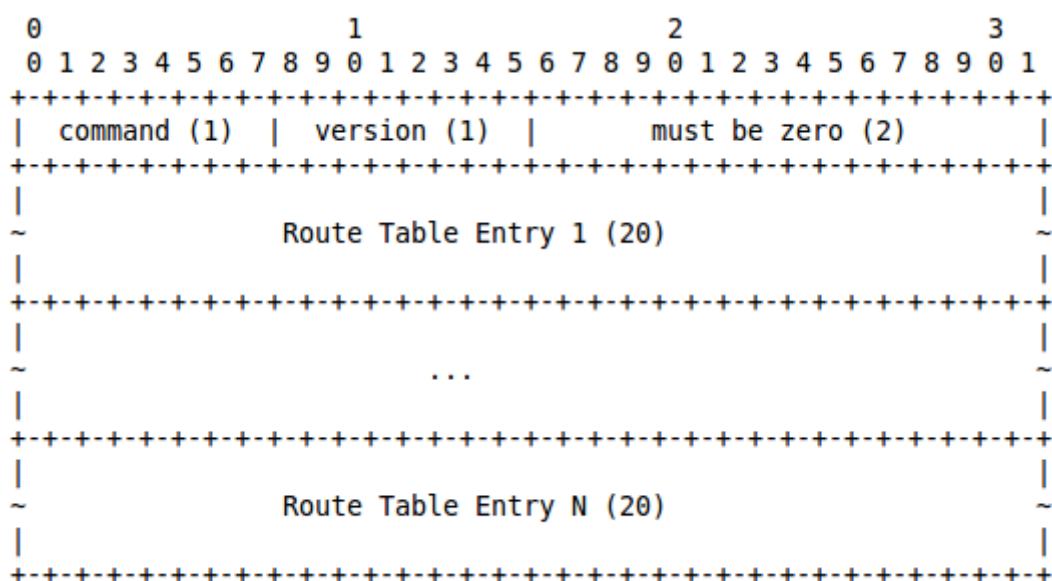
Položka "Subnet Mask" (Obrázek č. 3) ukazuje, že v datagramu se přenáší i maska podsítě na rozdíl od RIPv1, který masku podsítě neposílá. Toto vylepšení povoluje využít ve směrování podsítě.



Obrázek č. 4 - RIPv2 Authentication část RIP datagramu

2.3 RIPv2

Jedná se o rozšíření RIPv2 o podporu zahrnující IPv6 adresy. RIPv2 tedy navíc od RIPv2 umožňuje podporu IPv6 síťování. Na druhou stranu nepodporuje autentizaci (nahrazeno Ipsec) a připojování libovolných tagů ke směrovačům. RIPv2 datagram je vyobrazen na Obrázku č. 5.



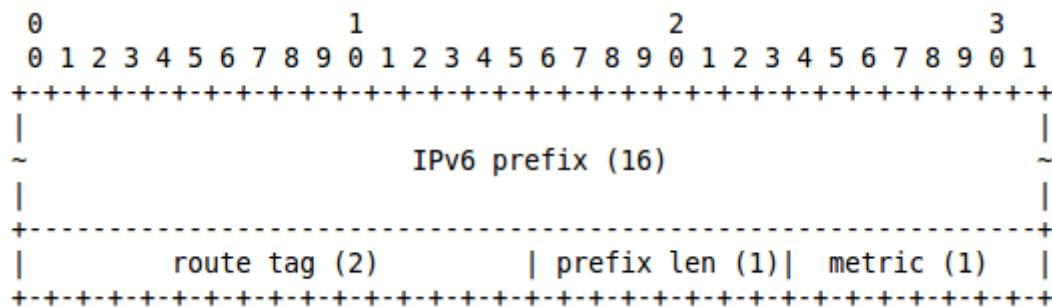
Obrázek č. 6 - RIPv2 datagram

Každý RIPv2 datagram obsahuje 1 až N entry tabulek. Číslo N se spočítá podle vzorce (Obrázku č. 7).

$$\#RTEs = \text{INT} \left(\frac{MTU - \text{sizeof}(IPv6_hdrs) - UDP_hdrlen - RIPv2_hdrlen}{RTE_size} \right)$$

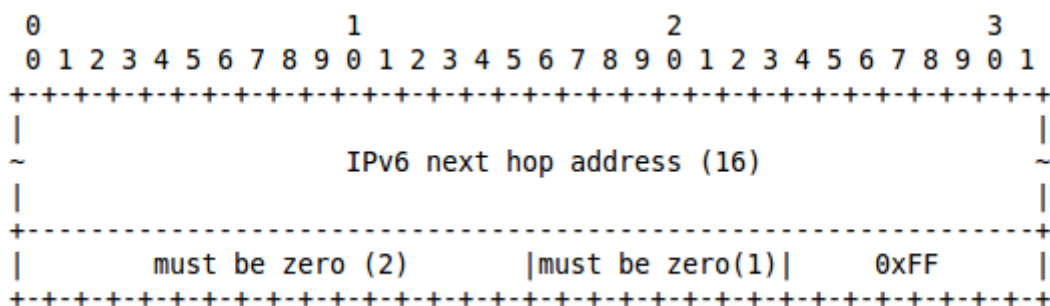
Obrázek č. 7 - Vzorec pro výpočet počtu entry tabulek

Formát RIPng entry tabulky je zobrazen na *obrázku č. 8*. Pokud je ovšem v části datagramu, kde se nachází metrika hodnota 0xFF nejedná se o entry tabulku, ale o Next Hop Entry tabulku (*Obrázek č. 9*).



Obrázek č. 8 - RIPng entry tabulka

Next Hop Entry tabulka uchovává informace o následujícím hopu, který se v síti nachází.



Obrázek č. 9 - RIPng RTE table (tabulka dalšího skoku)

3 Návrh a implementace

Pro implementaci projektu byl zvolen jazyk C++, ale bez jeho objektových vlastností. Hlavní využití spočívalo ve zpracování textu a jednodušší práci (alespoň pro mě) než s jazykem C.

Návrh výsledného výpisu aplikace byl převzat z aplikace Wireshark, kde nejdříve bylo odchyceno několik RIP a RIPv2 datagramů z připraveného virtuálního systému pro generování potřebných datagramů, a poté přišlo na řadu zpracování datagramů do takové formy, aby mohli být výsledky lehce prezentovány uživateli.

3.1 Odchytávač

Program pro odchyťávání RIP datagramů (myripsniffer) má jeden povinný parametr, kterým je jméno interface, na kterém se mají datagramy odchytávat. Spuštění programu tedy proběhne takto - `./myripsniffer -i "interface"`.

Pokud je zadáno neexistující rozhraní se program ukončí s výpisem na chybový výstup s popisem chyby.

Parametry aplikace jsou zpracovány pomocí funkce `getopt()` ve funkci `getParams()`. Zpracované parametry jsou ukládány do struktury pro další využití. Pro spuštění odchyťávání se spustí nejprve funkce `openInterface()`, která zjistí adresu interface zpřístupní interface funkci `capturePacket()`. Tato funkce pot odchyťává datagramy s portem 520(RIP) a 521(RIPv2), dokud není ukončen celý program.

Srdcem aplikace je funkce `parsePacket()`, která slouží pro zpracování příchozího datagramu. Zpracuje nejprve Ethernet header, IP header, UDP header a poté, podle informací získaných z těchto částí datagramu, se plní struktury pro RIP datagram, které jsou definovány v souboru `tables.h`.

3.2 Podvrhávač

Podvrhávač falešných RIPv2 RESPONSE zpráv funguje podobně jako odchytávač. Povinný parametr je `-r "ip-address/mask"`, který udává adresu podvrhované sítě a za lomítkem je zadána číselná délka masky. Nepovinnými parametry jsou:

- `-i "interface"` - udává interface, ze kterého bude falešný datagram odeslán (pokud není zadán jsou odeslány datagramy ze všech rozhraní, které mají přidělenou IP adresu)

- `-m "číslo metriky"` - udává počet hopů 1-15 (implicitně 1)

- `-n "adresa dalšího hopu"` - udává adresu dalšího hopu (implicitně 0.0.0.0)

- `-t "route tag"` - hodnota Router Tagu (implicitně 0)

- `-p "heslo"` - pokud je použit tento přepínač je datagram zabezpečen pomocí simple-password autentizace.

Příklad spuštění bude uveden v sekci 4.

Parametry jsou zpracovány pomocí `getopt()` ve funkci `getParams()`. Zpracované parametry jsou ukládány do struktury pro další využití. Po zpracování parametrů jsou ve funkci `main()` naplněny tabulky `RIP_HEADER`, `RIP_EXT` a pokud je uveden přepínač `-p` tak i `RIP_AUTH`. Potřebné informace se řadí za sebou v paměti, na kterou je uložen ukazatel.

Po vytvoření datagramu v paměti je zavolána funkce `sendPacket()`, která vyplní poslední potřebné informace jako jsou port, zdrojová a cílová adresa a poté pošle packet pomocí funkce `bindANDSend()`. V případě nezadaného parametru `-i` se nejprve

zjistí všechny interface počítače, které mají přiřazenou IP adresu. Tyto informace jsou poté uloženy ve struktuře, která se prochází a na každou nalezenou IP adresu je zaslán požadovaný datagram.

4 Příklady užití a vyhodnocení výsledků

Příklad užití odchytače:

./myripsniffer -i wlan0

Odchycený RIPv2 datagram.

Získané heslo: **ISA>294172a8c9d**

Typ: **RESPONSE**

Odchycené routy: **10.50.52.0, 10.101.216.0, 10.116.106.0, 10.222.115.0**

+++++[RIPv2]+++++

Time: Sun Nov 22 17:04:23 2015

UDP-RIP-IPv4 10.0.0.1 -> 224.0.0.9

PortNumber: 520

RIP packet type: RESPONSE

Password: ISA>294172a8c9d

```
=====
IP address      Netmask          NextHop          Metrika    RouteTag    Address Family
10.50.52.0      255.255.255.0    0.0.0.0          1           0           2
10.101.216.0    255.255.255.0    0.0.0.0          1           0           2
10.116.106.0    255.255.255.0    0.0.0.0          1           0           2
10.222.115.0    255.255.255.0    0.0.0.0          1           0           2
=====
+++++
```

Odchycený RIPv6 datagram.

Typ: **RESPONSE**

Odchycené routy: **fd00::, fd00:d8:35e8::, fd00:110:2f9e::, fd00:1f9:73::, fd00:a28:1790::**

+++++[RIPv6]+++++

Time: Sun Nov 22 17:03:58 2015

UDP-RIPv6 fe80::a00:27ff:fe01:2436 -> ff02::9:

PortNumber: 521

RIP packet type: RESPONSE

```
=====
Prefix          RouteTag          PrefixLen        Metrika
fd00::          0                 64               1
fd00:d8:35e8::  0                 64               1
fd00:110:2f9e:: 0                 64               1
fd00:1f9:73::   0                 64               1
fd00:a28:1790:: 0                 64               1
=====
+++++
```

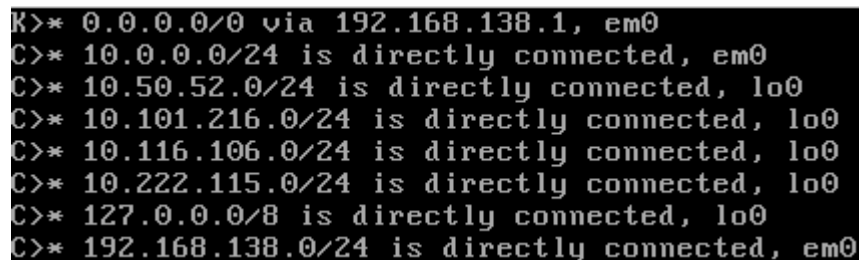
Příklad užití odchytače:

./myripresponse -i wlan0 -r 10.10.10.0/24 -m 5 -t 666 -p "ISA>294172a8c9d"

Po podvržení datagramu se tento datagram objeví ve výpisu odchytače:

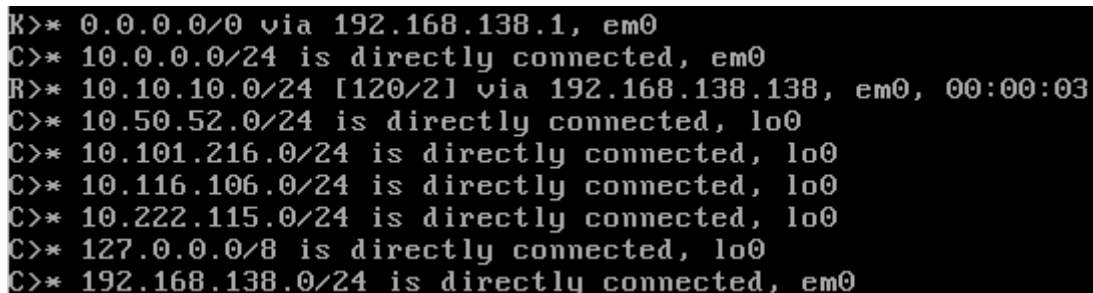
```
+++++[RIPv2]+++++
Time: Sun Nov 22 17:18:21 2015
UDP-RIP-IPv4 192.168.138.138 -> 224.0.0.9
PortNumber: 520
RIP packet type: RESPONSE
Password: ISA>2a431d693baa
=====
IP address      Netmask          NextHop          Metrika          RouteTag          Address Family
10.10.10.0      255.255.255.0    0.0.0.0          5                 666               2
=====
+++++
```

O správnosti podvržení se můžeme přesvědčit v připraveném virtuálním stroji. Při odeslání toho datagramu je upravena routovací tabulka. Tabulku před odesláním datagramu zachycuje *Obrázek č. 10* a tabulku po odeslání datagramu zachycuje *Obrázek č. 11*.



```
K>* 0.0.0.0/0 via 192.168.138.1, em0
C>* 10.0.0.0/24 is directly connected, em0
C>* 10.50.52.0/24 is directly connected, lo0
C>* 10.101.216.0/24 is directly connected, lo0
C>* 10.116.106.0/24 is directly connected, lo0
C>* 10.222.115.0/24 is directly connected, lo0
C>* 127.0.0.0/8 is directly connected, lo0
C>* 192.168.138.0/24 is directly connected, em0
```

Obrázek č. 10 - Routovací tabulka před odesláním datagramu



```
K>* 0.0.0.0/0 via 192.168.138.1, em0
C>* 10.0.0.0/24 is directly connected, em0
R>* 10.10.10.0/24 [120/21] via 192.168.138.138, em0, 00:00:03
C>* 10.50.52.0/24 is directly connected, lo0
C>* 10.101.216.0/24 is directly connected, lo0
C>* 10.116.106.0/24 is directly connected, lo0
C>* 10.222.115.0/24 is directly connected, lo0
C>* 127.0.0.0/8 is directly connected, lo0
C>* 192.168.138.0/24 is directly connected, em0
```

Obrázek č. 11 - Routovací tabulka po odeslání datagramu

5 Závěr

Program *myripsniffer* funguje jako snímač RIP a RIPv2 datagramů na určitém rozhraní, které si definuje uživatel. Výsledky pro uživatele vypisuje na stdout, ale je možné přeměrovat výstup do souboru. Program je ukončen až po zaslání ukončovacího signálu.

Program *myripresponse* funguje jako podvrhávač RIPv2 RESPONSE zpráv. Kam a jaký datagram je odeslán definuje uživatel volbou příslušných parametrů.

Zdrojové programy obou programů je možné přeložit pomocí souboru *Makefile*. Stačí v terminálu zadat příkaz "make".

Programy byly vyvíjeny pod systémem Linux Mint 17.2. Testován byl pod systémem Linux Mint 17.2 a Ubuntu 14.04 LTS a splnil všechna potřebná testovací kritéria.

A Metriky kódu

Počet řádků: 1013

Počet souborů: 3

Velikost spustitelných souborů:

myripsniffer - 28445B

myripresponse - 18299B

Reference

[1] *Wikipedie: Otevřená encyklopedie: Routing Information Protocol* [online]. c2014 [citováno 22. 11. 2015]. Dostupný z WWW: <https://cs.wikipedia.org/w/index.php?title=Routing_Information_Protocol&oldid=11523945>

[2] IETF Tools. RFC 1058 - Routing Information Protocol [online]. ©1988 [cit. 2015-11-22]. Dostupné z: <https://tools.ietf.org/html/rfc1058>

[3] IETF Tools. RFC 2453 - RIP Version 2 [online]. ©1998 [cit. 2015-11-22]. Dostupné z: <https://tools.ietf.org/html/rfc2453>

[4] IETF Tools. RFC 2080 - RIPng for IPv6 [online]. ©1998 [cit. 2015-11-22]. Dostupné z: <https://tools.ietf.org/html/rfc2080>