



“Nástroje monitorující a generující zprávy jednoduchých distance-vector protokolů “

Predmět: ISA

Author: Ksenia Bolshakova

Brno 2018

Obsah

1. Uvod
2. RIPprotokol
 - a. RIPv1
 - b. RIPv2
 - c. RIPvng
3. Implementace Snifferu pro RIPv1, RIPv2 a RIPvng zprávy.
4. Implementace myripresponse
5. Utok
6. Implementace myriprequest
7. Výstup programu
8. Literatura

1.Úvod

Cílem projektu je vytvořit komunikující aplikaci podle konkrétní vybrané specifikace pomocí síťové knihovny BSD sockets v jazyce C/C++.

První úkol byl nastudovat si směrovací protokoly RIP a RIPng.

Druhý úkol je naprogramovat sniffer RIPv1, RIPv2 a RIPng zpráv.

Třetí úkol je naprogramovat podvrhávač falešných RIPng Response zpráv.

Čtvrtý je provést úspěšný útok.

Pátý naprogramovat podvrhávač falešných RIPng Request zpráv (bonusový úkol).

2.RIP protocol

Routing Information Protocol (RIP) je směrovací protokol umožňující směrovačům (routerům) komunikovat mezi sebou a reagovat na změny topologie počítačové sítě. RIP používá UDP (User Datagram Protocol) jako transportní protokol. Na portu 520 běží RIPv1 a RIPv2, na portu 521 běží RIPng. Vzniknul dany protokol v roce 1969.

Timer

Každých 30 vteřin se process RIP probudí a pošle se nevyžádaná zpráva odpovědí, která obsahuje celou RTE.

a) RIP verze 1

Originální specifikace RIPu dle RFC 1058 používá směrování podle původních tříd IPv4 adres A, B nebo C. Periodické aktualizace směrování nezahrnují informace o masce sítě, protože podle původního systému je maska dána příslušností IP adresy do jedné ze tříd. Chybí tak podpora pro CIDR (Classless Inter-Domain Routing), což znemožňuje existenci různě velkých podsítí uvnitř jedné třídy IP adres. Všechny podsítě musí být stejně velké (tj. se stejnou maskou). Neexistuje zde podpora pro vzájemnou autentizaci routerů, a proto je protokol RIPv1 napadnutelný nejrůznějšími útoky.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Address family identifier (2)																Route Tag (2)															
IPv4 address (4)																															
Subnet mask (4)																															
Next hop (4)																															
Metric (4)																															

Autentifikace

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
command (1)								version (1)								must be zero (2)															
0xFFFF																Authentication Type (2)															
Authentication (16)																															

c) RIPng

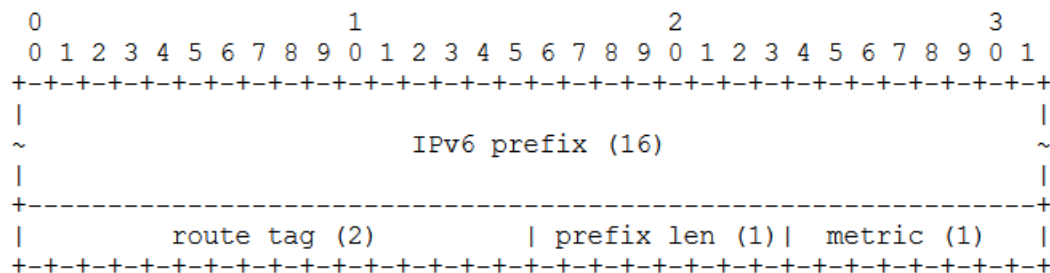
RIPng (RIP další generace), který je definován v [RFC 2080](#), je rozšířením RIPv2 zahrnující podporu IPv6 (Internetového Protokolu další generace). Hlavní rozdíly mezi RIPv2 a RIPng jsou:

- podpora IPv6 síťování
- podpora aktualizovaných autentizací (verifikace ověření identity osoby nebo procesu zabezpečovacím systémem) - RIPng nepodporuje (nahrazeno IPsec)
- připojování libovolných tagů k směrovačům (routerům) - RIPng nepodporuje
- kódování dalších skoků do každého směrovacího záznamu - RIPng vyžaduje specifické kódování na dalším skoku, pro set směrovacích záznamů

RIPng packet format:

[illegible]

Kde každá Route Table Entry (RTE) má následující formát:

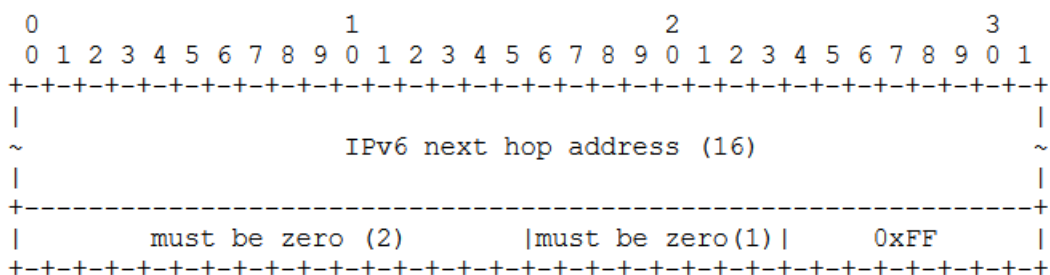


Maximální číslo RTE je definováno dolu.

Ve Version naimplementovány příkazy 1 – request (požadavek, aby reagující systém poslal celou nebo část tabulky), 2 – response (zpráva obsahující všechny nebo části odesílatele směrovací tabulky. Tato zpráva může být zaslána jako odpověď k žádosti nebo může být nevyžádaná směrování aktualizace generované odesílatelem).

RIPng umožňuje specifikovat next hop IPv6 adresu, přes kterou bude směrován paket do cíle.

Next hop Route Table Entry (RTE) format:



Implicitní format next hopu je 0:0:0:0:0:0:0 indikuje, že příští adresa hopu by měla být původcem vizitky RIPng. Adresa specifikovaná jako next hop musí být link-local adresa. Účelem next hop RTE je vyloučit směrování paketů prostřednictvím extra hopů v systému.

3. Implementace Snifferu pro RIPv1, RIPv2, RIPng

Spustí se jako ./myripsniffer -i [rozhraní]. Pokud uživatel bude chtít poradit, spustí program jako ./myripsniffer -h, a vypíše se mu nápověda.

Pro implementace snifferu byla využita knihovna libpcap.

Funkce *getopt()* používá se pro parsování command-line argumentů, která vezme na vstupu parametr -i a jeho argument – název rozhraní a uloží ho do proměnné *dev*.

Na zaklade tutoriálu o knihovně libpcap byl vytvořen sniffer.

Funkce *pcap_lookupnet()* se používá pro určení subnet a netmask rozhraní, v případě chyby vrátí -1.

pcap_open_live() získá handle pro živé zachycení.

pcap_datalink() určí formát paketů dodaných handlem.

Pomocí *pcap_compile()* řetězec filtru je kompilován do programu pseudo-machine-language.

Výsledný program může vytvořit filtr pro handle pomocí *pcap_setfilter()*.

Funkce *ripInfo()* vypisuje všechny potřebné informace pro RIPv1, RIPv2 a RIPv2. Podle verze IP adresy určí, jestli jsou to RIPv2 nebo RIPv1 a RIPv2 protokoly. Na základě formátu dat uložených v protokolu, vypíše verze protokolu, command, metriku a IP adresu.

Pokud je to RIPv2 nebo RIPv2 vypíše autentifikaci. Když typ autentifikace je 02 (Simple Password) u RIPv2, vypíše heslo. Když je typ autentifikace je 03 vypíše "MD5" (šifrované heslo). Pokud je na bytech 48 a 49 uloženo něco jiného, vypíše "Other". Dále vypíše zdrojovou a cílovou IP adresu.

V cyklu se vypisuje Route Table Entry pro RIPv2, RIPv2.

Funkce *pcap_loop(handle, 0, ripInfo, NULL)*; spouští do nekonečna funkci *ripInfo()*, dokud uživatel neukončí ten program. Druhý parametr "0" umožňuje toto nastavení.

4. Implementace myripresponse

Program se spouští:

```
./myripresponse -i <interface> -r <IPv6>/[16-128] {-n <IPv6>} {-m [0-16]} {-t [0-65535]}
```

Kde

- * *-i: <rozhraní>* udává rozhraní, ze kterého má být útočný paket odeslán
- * *-r: v <IPv6>* je IP adresa podvrhávané sítě a za lomítkem číselná délka masky sítě
- * *-m:* následující číslo udává RIP Metriku, tedy počet hopů, implicitně 1
- * *-n: <IPv6>* za tímto parametrem je adresa next-hopu pro podvrhávanou routu, implicitně ::
- * *-t:* číslo udává hodnotu Router Tagu, implicitně 0.

Nebo se spouští:

```
./myripresponse -h
```

Aby se vypsala nápověda.

V souboru *ripresponse.cpp*:

Pomocí funkce *getopt()* načte argumenty z command-line, rozparsuje je a uloží do odpovídajících proměnných, zkontroluje, aby byli všechny povinné argumenty nastaveny.

Okamžitě se kontroluje, jestli jsou zadané hodnoty, jsou v zadaném rozsahu.

Pomocí funkce *inet_pton()* konvertujeme zadanou uživatelem IP adresy podvrhávané sítě a next hop do struktury *in6_addr*, která se používá pro IPv6 adresy.

Funkce `if_nametoindex(optarg)` konvertuje uživatelem zadaný interface na index.

Do struktury `source_addr` nastaví se všechny parametry pro source address (adresu, port, address family), to stejné se udělá pro strukturu `dest_addr`, která bude přijímat socket.

Funkce `socket();` vytváří socket s určitými parametry.

Funkce `setsockopt(socket_of_client, IPPROTO_IPV6, IPV6_MULTICAST_IF, &index, sizeof(index))` nastavuje parametry pro vytvořený socket, kde index je konvertované rozhraní.

Funkce `bind(socket_of_client, (struct sockaddr *) &source_addr, sizeof(source_addr))` napojuje socket na zadaný port.

Funkce `setsockopt(socket_of_client, IPPROTO_IPV6, IPV6_MULTICAST_HOPS, &max_hop, sizeof(max_hop))` nastavuje maximální počet hopů pro daný socket.

Potom se vytváří nový RIPng packet s parametry, zadanými uživatelem.

Funkce `sendto(socket_of_client, ripngPacket->packet, ripngPacket->length, 0, (struct sockaddr*)&dest_addr, sizeof(dest_addr))` odesílá vytvořený socket.

V souboru `ripngpacket.cpp`

V tomto souboru se vytváří RIPng packet. Používá se a i pro ripresponse, a i pro riprequest.

Na začátku se alokuje paměť pro packet, zakládají se pole pro ukládání adresy next hopu, proměnná pro ukládání délky packetu.

Do packetu nejprve uložíme Command, pomocí funkce `memset()`, pro Verzi, 3 a 4 bity musí mít hodnotu "0". Dále nakopírujeme IPv6 adresu, pomocí `memcpy()`, kterou zadal uživatel. Zatím se uloží Route Tag, délka prefixu, a metrika. Dále se uloží next hop a 0xff uložené na konci.

5. Útok

Pomocí aplikace `myripresponse` byl podvrhnout SW směrovači zprávou RIP Response routu `2001:db8:0:abcd::/64`. Úspěch podvrhnutí byl ověřen (ve virtuálním počítači pomocí příkazu `show ipv6 route`, byla zobrazena aktuální směrovací tabulka, kde byla zobrazena poslána zpráva.

Útok byl úspěšně proveden, výsledek je vidět v sekci Výstup programu.

6. Implementace myriprequest

Implementace je podobná jako u `myripresponse` - jediné, co se změní, je Command.

7. Výstup programů:

Myripsniffer

RIPv1, RIPv2

```
Version: RIPv2
Authentication type: Simple password(2).
Authentication: ISA>28813508880
```

```
Route Entry.
Address Family: 2
Route tag: 0
IP address: 10:0:0:0
Netmask: 255:255:255:0
Next hop: 0:0:0:0
Metric: 1
```

```
Address Family: 2
Route tag: 0
IP address: 10:48:48:0
Netmask: 255:255:255:0
Next hop: 0:0:0:0
Metric: 1
```

```
Address Family: 2
Route tag: 0
IP address: 10:108:206:0
Netmask: 255:255:255:0
Next hop: 0:0:0:0
Metric: 1
```

```
Address Family: 2
Route tag: 0
IP address: 10:111:115:0
Netmask: 255:255:255:0
Next hop: 0:0:0:0
Metric: 1
```

```
Address Family: 2
Route tag: 0
IP address: 10:226:104:0
Netmask: 255:255:255:0
Next hop: 0:0:0:0
Metric: 1
```

RIPng

```
RIPpacket
Version: RIPng
Command: Response (2)

Route Table Entry:
IPv6 Prefix: fd00:0000:0000:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 64
Metric: 1

IPv6 Prefix: fd00:00d4:30c0:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 64
Metric: 1

IPv6 Prefix: fd00:0112:2c06:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 64
Metric: 1

IPv6 Prefix: fd00:0900:14d0:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 64
Metric: 1

IPv6 Prefix: fd00:0948:0062:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 64
Metric: 1

IPv6 Prefix: 0000:0000:0000:0000:0000:0000:0000
Route Tag: 0
Prefix Length: 0
Metric: 0

Source IPv6 address: fe80:0000:0000:0000:0a00:27ff:fe0e:ede7
Destination IPv6 address: ff02:0000:0000:0000:0000:0000:0000:0009
```

Myripresponse

No.	Time	Source	Destination	Protocol	Length	Info
52	119.7132009	fe80::800:27ff:fe00:0	ff02::9	RIPng	86	Command Response, Version 1
53	124.4903463	fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
54	124.4903527	fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
55	125.0906229	CadmusCo_0e:ed:e7	Broadcast	ARP	42	Who has 192.168.56.100? Tell 192.168.56.101
56	125.0906352	CadmusCo_0e:ed:e7	Broadcast	ARP	42	Who has 192.168.56.100? Tell 192.168.56.101
57	125.0999277	192.168.56.100	255.255.255.255	DHCP	590	DHCP ACK - Transaction ID 0xa95785f5
58	125.0999353	192.168.56.100	255.255.255.255	DHCP	590	DHCP ACK - Transaction ID 0xa95785f5
59	134.6454317	fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
60	134.6454406	fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
61	135.0509145	fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1
62	135.0509259	fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1
63	136.0715948	10.0.0.1	224.0.0.9	RIPv2	146	Response
64	136.0716029	10.0.0.1	224.0.0.9	RIPv2	146	Response
65	136.0738360	192.168.56.101	224.0.0.9	RIPv2	166	Response
66	136.0738414	192.168.56.101	224.0.0.9	RIPv2	166	Response

▶ User Datagram Protocol, Src Port: ripng (521), Dst Port: ripng (521)

▼ RIPng

- Command: Response (2)
- Version: 1
- Reserved: 0000
- ▼ Route Table Entry: IPv6 Prefix: 124:123:1232:1232::/64 Metric: 1
 - IPv6 Prefix: 124:123:1232:1232:: (124:123:1232:1232::)
 - Route Tag: 0x0000
 - Prefix Length: 64

```
3000 33 33 00 00 00 09 0a 00 27 00 00 00 86 dd 60 00 33.....'.....
3010 00 00 00 20 11 01 fe 80 00 00 00 00 00 00 00 00 .....
3020 27 ff fe 00 00 00 ff 02 00 00 00 00 00 00 00 00 .....
3030 00 00 00 00 00 09 02 09 02 09 00 20 67 62 02 01 ..... gb..
3040 00 00 01 24 01 23 12 32 12 32 00 00 00 00 00 00 ...$.#.2 .2.....
3050 00 00 00 00 40 01 .....@.
```

```
Reserved: 0000
▼ Route Table Entry: IPv6 Prefix: 124:123:1232:1232::/64 Metric: 6
    IPv6 Prefix: 124:123:1232:1232:: (124:123:1232:1232::)
    Route Tag: 0xe8fd
    Prefix Length: 64
    Metric: 6
▼ Route Table Entry: IPv6 Prefix: 1212:1313:7878:0:1111:2222:0:1/0 Metric: 255
    IPv6 Prefix: 1212:1313:7878:0:1111:2222:0:1 (1212:1313:7878:0:1111:2222:0:1)
    Route Tag: 0x0000
```

Utok

203 499.9410154:192.168.56.101	224.0.0.9	RIPv2	166 Response
204 503.6204511:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
205 503.6204500:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
206 510.1401261:cadmusCo.0e:ede7	Broadcast	ARP	42 Who has 192.168.56.100? Tell 192.168.56.101
207 510.1401328:cadmusCo.0e:ede7	Broadcast	ARP	42 Who has 192.168.56.100? Tell 192.168.56.101
208 510.1430201:192.168.56.100	255.255.255.255	DHCP	590 DHCP ACK - Transaction
209 510.1430247:192.168.56.100	255.255.255.255	DHCP	590 DHCP ACK - Transaction
210 513.7298282:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
211 513.7298345:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
212 523.8304110:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
213 523.8304173:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198 Router Advertisement
214 523.9543706:10.0.0.1	224.0.0.9	RIPv2	146 Response
215 523.9543825:10.0.0.1	224.0.0.9	RIPv2	146 Response
216 523.9586769:192.168.56.101	224.0.0.9	RIPv2	166 Response
217 523.9586850:192.168.56.101	224.0.0.9	RIPv2	166 Response

Ethernet II, Src: 0a:00:27:00:00:00 (0a:00:27:00:00:00), Dst: IPv6mcast.00:00:00:09 (33:33:00:00:00:09)
Internet Protocol Version 6, Src: fe80::800:27ff:fe00:0 (fe80::800:27ff:fe00:0), Dst: ff02::9 (ff02::9)
User Datagram Protocol, Src Port: ripng (521), Dst Port: ripng (521)
RIPng
Command: Response (2)
Version: 1
Reserved: 0000
Route Table Entry: IPv6 Prefix: 2001:db8:0:abcd::/64 Metric: 1
Route Table Entry: IPv6 Prefix: ::/0 Metric: 255

00 33 33 00 00 00 09 0a 00 27 00 00 00 06 dd 60 00	33.....
10 00 00 00 34 11 ff fe 80 00 00 00 00 00 08 00	...4.....
20 27 ff fe 00 00 00 ff 02 00 00 00 00 00 00 00	...7fffe... ..
30 00 00 00 00 00 09 02 09 02 09 00 34 03 5f 02 019... ..
40 00 00 20 01 0d b8 00 00 ab cd 00 00 00 00 00 00ab... ..
50 00 00 00 40 01 00 00 00 00 00 00 00 00 00 0001... ..
60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00f.....

```

ISA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Hello, this is Quayga (version 0.99.16).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification
Password:
Routing? show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng, O - OSPFv3,
I - ISIS, B - BGP, * - FIB route.

R>* ::/96 via ::1, lo0, rej
C>* ::1/128 is directly connected, lo0
K>* ::ffff:0.0.0.0/96 via ::1, lo0, rej
R>* 2001:db8:0:abcd::/64 [120/21] via fe80::800:27ff:fe00:0, em0, 00:02:32
C>* fd00::/64 is directly connected, em0
C>* fd00:d4:30c0::/64 is directly connected, lo0
C>* fd00:112:2c06::/64 is directly connected, lo0
C>* fd00:900:14d0::/64 is directly connected, lo0
C>* fd00:948:62::/64 is directly connected, lo0
R>* fe80::/10 via ::1, lo0, rej
C>* fe80::/64 is directly connected, lo0
C>* fe80::/64 is directly connected, em0
K>* ff02::/16 via ::1, lo0, rej
Routing?

```

```

34 Files: riprequest.cpp, ripngpacket.cpp, ripngpacket.h
35
36 -----
37
38 Examples how
39
40 ./myripsniff
Routing> [J]
41 Routing> [J]
42 ./myriprespon
Routing> [J]
43 ./myriprespon
Routing> [J]
44 ./myriprespon
Routing> show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng, O - OSPFv3,
I - ISIS, B - BGP, * - FIB route.
45
46 ./myriprespon
K>* ::/96 via ::1, lo0, rej
47 C>* ::1/128 is directly connected, lo0
48 K>* ::ffff:0.0.0.0/96 via ::1, lo0, rej
R>* 1233:1238:0:abcd::/64 [120/21] via fe80::800:27ff:fe00:0, em0, 00:00:56
C>* fd00::/64 is directly connected, em0
C>* fd00:d4:30c0::/64 is directly connected, lo0
C>* fd00:112:2c06::/64 is directly connected, lo0
C>* fd00:900:14d0::/64 is directly connected, lo0
C>* fd00:948:62::/64 is directly connected, lo0
K>* fe80::/10 via ::1, lo0, rej
C>* fe80::/64 is directly connected, lo0
C>* fe80::/64 is directly connected, em0
K>* ff02::/16 via ::1, lo0, rej
Routing>

```

```

[root@localhost ISAprjekt]# make
make: Nothing to be done for 'all'.
[root@localhost ISAprjekt]# ./myripresponse -i vboxnet0 -r 1233:1238:0:abcd::/64
Interface: vboxnet0
IP address: 64, netmask: 64
IP address of next hop: ::
Metric: 1
Router tag: 0
index 5
[root@localhost ISAprjekt]#

```

Myriprequest

Time	Source	Destination	Protocol	Length	Info
248	575.948227:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
249	579.9745998:fe80::800:27ff:fe00:0	ff02::9	RIPng	86	Command Request, Version 1
250	579.9772950:fe80::a00:27ff:fe0e:ede7	fe80::800:27ff:fe00:0	RIPng	86	Command Response, Version 1
251	579.9773412:fe80::800:27ff:fe00:0	fe80::a00:27ff:fe0e:ede7	ICMPv6	134	Destination Unreachable (Administratively prohibited)
252	580.3183462:fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1
253	580.3183526:fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1
254	584.0674134:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
255	584.0674241:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
256	594.1687635:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
257	594.1687754:fe80::a00:27ff:fe0e:ede7	ff02::1	ICMPv6	198	Router Advertisement
258	595.2699683:10.0.0.1	224.0.0.9	RIPv2	146	Response
259	595.2699793:10.0.0.1	224.0.0.9	RIPv2	146	Response
260	595.2724410:192.168.56.101	224.0.0.9	RIPv2	166	Response
261	595.2724469:192.168.56.101	224.0.0.9	RIPv2	166	Response
262	602.3289822:fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1
263	602.3289921:fe80::a00:27ff:fe0e:ede7	ff02::9	RIPng	166	Command Response, Version 1

► User Datagram Protocol, Src Port: ripng (521), Dst Port: ripng (521)

▼ RIPng

- Command: Request (1)
- Version: 1
- Reserved: 0000
- ▼ Route Table Entry: IPv6 Prefix: ::/0 Metric: 1
 - IPv6 Prefix: :: (::)
 - Route Tag: 0x0000
 - Prefix Length: 0

0000	33 33 00 00 00 09 0a 00	27 00 00 00 86 dd 60 00	33..... '.....'
0010	00 00 00 20 11 01 fe 80	00 00 00 00 00 00 08 00 00 00 08 00
0020	27 ff fe 00 00 00 ff 02	00 00 00 00 00 00 00 00	'..... 00 00 00 00
0030	00 00 00 00 00 09 02 09	02 09 00 20 cf 0d 01 01 00 0d 01 01
0040	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00
0050	00 00 00 00 00 01	

8.Literatura

https://cs.wikipedia.org/wiki/Routing_Information_Protocol

<https://tools.ietf.org/html/rfc1058>

<https://tools.ietf.org/html/rfc2453>

<https://tools.ietf.org/html/rfc2080>

<http://www.tcpdump.org/>