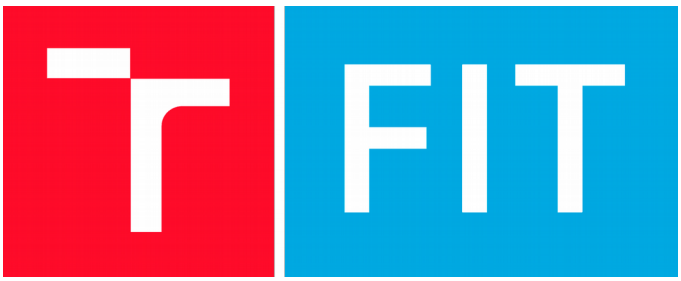


Dokumentace k projektu z ISA

Programování síťové služby

Nástroje moniturující a generující zprávy jednoduchých distance-vektor protokolů

Dne 18. listopadu 2018
Zdeněk Brhel, xbrhel04@stud.fit.vutbr.cz
Fakulta informačních technologií
Vysuké učení technické v Brně

Obsah

1. Úvod

Dokumentace slouží k pochopení protokolu RIP (*Routing Information Protocol*), který slouží na výměnu informací v sítí mezi bránami a koncovými počítači. Dále k pochopení 2 aplikací, které byly vytvořeny na toto téma. První z nich má za úkol odposlouchávat komunikaci těchto paketů na daném rozhraním. Druhý má za úkol podvrhnout zprávu tohoto protokolu a odeslat ji podle daného interface tak, aby ostatní počítače v síti pozměnily své směrování.

2. Uvedení do problematiky

Aby síť mohla řádně fungovat, je třeba, aby počítače věděli, komu mají odesílat své data. Kvůli tomuto problému vzniklo směrování, což označuje určování cest datagramů právě v prostředí těchto sítí. Jelikož ale jsou rozměry internetu obrovské, je v něm směrování realizováno hierarchicky – celý internet je rozdělen na častí s jednotnou směrovací politikou. Jednou z možností, jak tvořit směrování, je protokol RIP - *Routing Information Protocol*, který umožňuje směrovačům mezi sebou komunikovat a reagovat na změny v topologii počítačových sítích. Ačkoliv tento protokol patří mezi nejstarší doposud používané směrovací protokoly v sítích IP, je stále uplatnitelný v menších sítích – je jednoduchý a má nenáročnou konfiguraci.

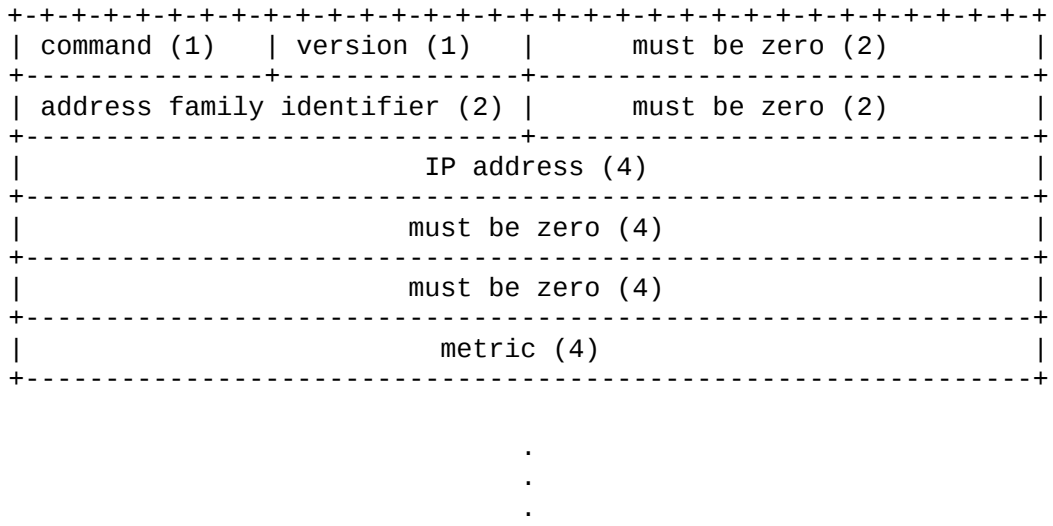
2.1 RIP Protokol

RIP je směrovací protokol typu *distance-vector* (neznají strukturu sítě za svými nejbližšími sousedy – tato informace je vyjádřena v metrice propagovaných směrovacích cest) využívající Bellmanův-Fordův algoritmus pro určení nejkratší cesty v síti. Metrikou směrování je počet skoků k cílové síti (hop-count). V případě RIP je tento limit omezen – maximální možný počet skoků je 15 (*0x0F*). Hodnota 16 (*0x10*) je brána jako nekonečná vzdálenost a používá se k označení nepřístupných a nepoužitelných směrovacích tras.

Původně routery využívající RIP vysílaly aktualizované směrovací tabulky v intervalu 30 sekund, avšak časem (až byly sítě větší) tímto vzniknul problém – síť byla každých 30 sekund zahlcena obrovským množstvím dat. Později bylo toto vyřešeno rozprostřením těchto zpráv v čase, aby nedocházelo k velkým narazovým tokům dat.

Původní verzí protokolu byl RIPv1 (verze 1), novější je RIPv2 (verze 2). Pro IPv6 byl pak definován RIPng (nová generace). RIPv2 měl oproti RIPv1 zabezpečení komunikace mezi směrovačí pomocí šifrovaného hesla a uměl přenášet síťové masky ve zprávách mezi směrovači (což umožňovalo používat podsítě). RIPng má podporu pro IPv6 síťování, podpora autentizací.

2.2 RIPv1 datagram



Část od ***address family identifier (2)*** se může objevit až 25x. Maximální velikost datagramů je 512 bytů.

`command (1)`
Určuje o jaký typ příkazu jde:

- 1 – Request (požadavek na systém, aby poslal celou, nebo část jeho routovací tabulky)
- 2 – Response (zpráva obsahující část, nebo celou routovací tabulku, může být odeslána buď jako odpověď na request, nebo jako aktualizace vygenerovaná odesílatelem)
- 3 – Traceon (zastaralý, ignorovaný)
- 4 – Traceoff (zastaralý, ignorovaný)
- 5 – Reserved (rezervovaná hodnota)

`version (1)`
Verze RIP, hodnota může být jen 1, nebo 2.

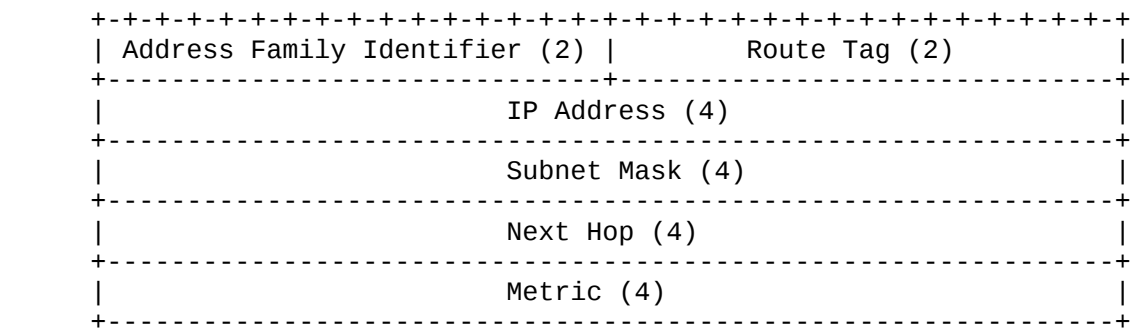
`address family identifier (2)`
Hodnota pro IP je 2. Žádná jiná implementace RIP není dostupná pro jiný typ adresy.

`IP address (4)`
Hodnotou je IP adresa v *network byte order*.

`Metric (4)`
Jak je psáno výše, hodnota musí být mezi 1 – 15, nebo hodnota 16, která indikuje nedostupnou (nekonečnou) cestu.

2.3 RIPv2 datagram

Od RIPv1 se liší pouze záznamem, přičemž ***Address Family Identifier (2)***, ***IP address (4)*** a ***metric (4)*** je stejný.



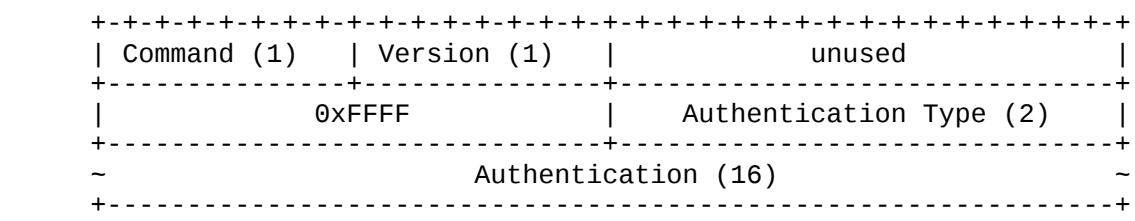
`Route Tag (2)`
Jde o atribut připsaný do cesty, který musí být zachován a znovu přeposlán s cestou. Zamýšlené použití Route Tagu je k poskytnutí metody k oddělení interních a externích RIP cest.

`Subnet Mask (4)`
Maska sítě pro IP adresu.

`Next Hop (4)`
`IP adresa dalšího skoku, do které mají být předány pakety, které směrují do cíle přes tuto trasu. Pokud je tato položka specifikována, jako 0.0.0.0, tak další skok by měl být přes původce šíření RIP zpráv.`

Autentizace

Autentizace je funkce, kterou umožňuje RIPv2. Je odeslána pouze jednou za zprávu. Určuje se tak, že první (vždy první) záznam začíná s *0xFFFF*. Zbytek záznamu obsahuje antentizaci.

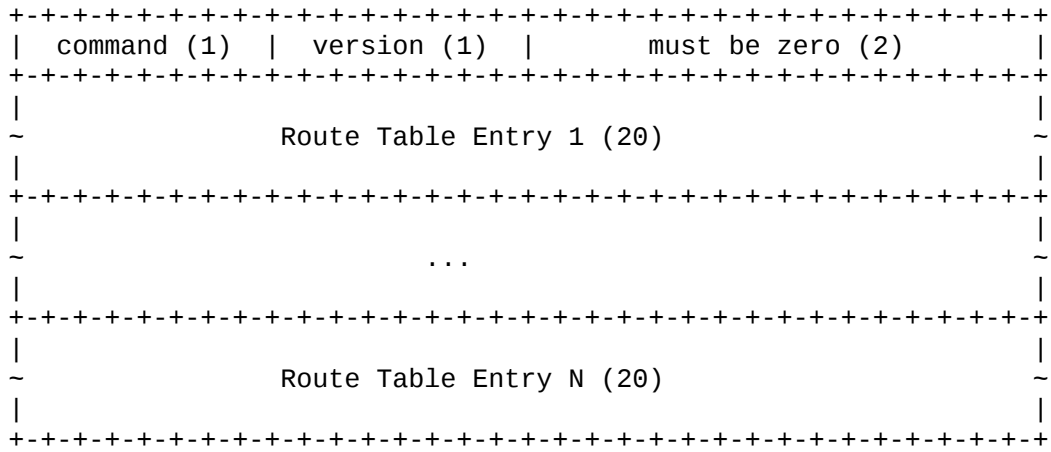


`0xFFFF`
Znamení, že v tomto záznamu jde o autentizaci.

`Authentication Type (2)`
Aktuálně je podporována pouze hodnota 2 – *Simple Password*, což je heslo zapsané v prostém textu.

`Authentication (16)`
16 bytů pro heslo v prostém textu. Pokud je heslo míň jak 16 bytů, musí být zarovnané v levo a doplněné o *0x00* hodnoty do prava.

2.4 RIPng datagram

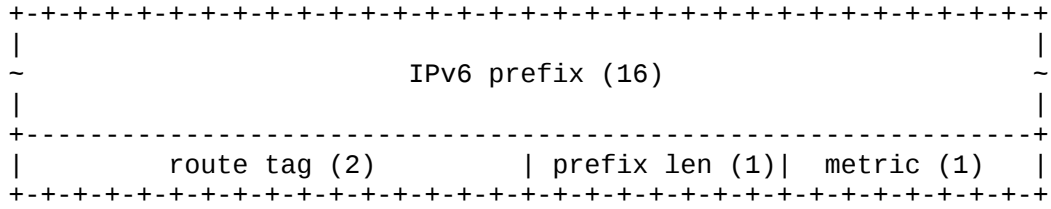


`command (1)`

- 1 (Request)
- 2 (Response)

`version (1)`
Existuje pouze verze 1.

`Route Table Entry 1 (20)`



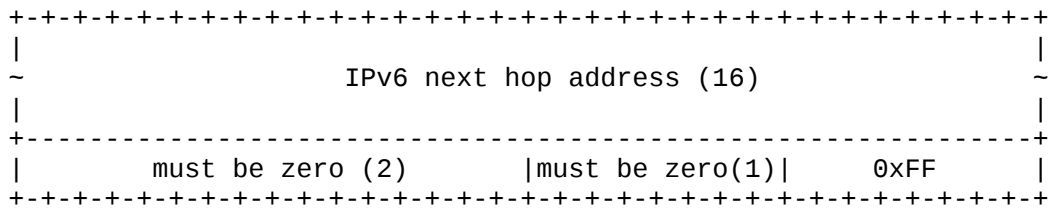
`IPv6 prefix (16)`
IPv6 uložená jako 16 bytů v *network-byte* posloupnosti.

`route tag (2)`
Atribut, se stejnými vlastnostmi jako u IPv4 RIPv2.

`prefix len (1)`
Obsahuje délku prefixu o velikosti 0 – 128.

`metric (1)`
Hodnota mezi 1 – 15, nebo hodnota 16 indikující nekonečno – destinace není dostupná.

Next hop



`IPv6 next hop address (16)`
Adresa dalšího skoku. Specifikováním této hodnoty jako :: (samé nuly) indikuje, že další skok by měl být na původce šířitele RIP response zpráv.

3. Návrh aplikací

Zde je popsáno jak byly navrženy a implementovány obě dvě aplikace.

3.1 Návrh a implementace snifferu

V hlavní funkci programu se vytvoří ovladač, který parsuje vstupní parametry. Tyto parametry jsou pak předány snifferu. O samotný sniffer se stará třída *Sniffer,* která je implementovaná v souborech *sniffer.cpp* a *sniffer.h.* Vstupním parametrem programu je rozhraní, na kterém by se měli odposlouchávat pakety. Pomocí knihovny *libpcap,* zavolat funkci `pcap_open_live()`, která poskytne *handle,* díky kterému je možné se dívat na pakety na daném rozhraní. Bylo potřeba nastavit *filtr*, který filtroval pouze pakety, o které jsme měli zájem – pro nás to byly porty 520 (pro RIP) a 521 (pro RIPng). Filtr se sestavil pomocí funkce `pcap_compile()`, a nastavil pomocí `pcap_setfilter()`. Pak už stačilo pouze začít nekonečně odposlouchávat na rozhraní pomocí `pcap_loop()` s druhým argumentem -1, neboť při specifikaci projektu bylo řečeno, že se sniffer bude ukončovat zasláním signálu *SIGTERM*.

V případě, že požadovaný packet prošel přes naše rozhraní, tak byl zachycen callback funkcí `gotPacket()`, která packet parsuje a vypisuje jeho obsah do konzole.

3.2 Návrh a implementace podvrhávače

Stejně jako v bodu 3.1, vytváří se ovladač, který dále parametry předává podvrhávači. O podvrhávač se pak stará třída *Smuggler* (implementována v souborech *smuggler.cpp* a *smuggler.h*). Aby jsme mohli zaslat na naše rozhraní packet, musíme packet vytvořit – o to se stará funkce `craftPacket()`. Funkce vytvoří velikost packetu podle toho, zda-li jde o RIP s jedním záznamem, nebo součástí packetu je i *next hop* záznam. Packet pak naplní požadovanými hodnotami. Funkce `sendPacket()` pak nastaví socket a odešle packet na daný interface; pomocí funkce `if_nametoindex()` zjistíme index interface a `getifaddrs()` nám zjistí lokální IPv6 adresu. Tyto hodnoty pak naplníme do `sockaddr_in6` struktur (jedna slouží pro bind socketu, druhá pro odeslání). Ještě před odesláním je potřeba nastavit socket - *TTL,* rozhraní. Packet byl odeslán na adresu *FF02::9*, což je adresa multicastu pro IPv6.

3.3 Zajímavý kód

Nastavování odchytávání paketů

```
    // Checking opening of device
    if((handle_ = pcap_open_live(interface.c_str(), BUFSIZ, 1, 1000,
errorBuffer_)) == nullptr)
        error("Can't open device " + interface + ".\n"
              + "Error message: " + errorBuffer_ + "\n");

    if(pcap_datalink(handle_) != DLT_EN10MB)
        error(interface + " doesn't provide Ethernet headers.");

    // Filter for RIP and RIPng
    filterSettings_ = "portrange 520-521"; //Set port range we want to sniff
(520 RIP, 521 RIPng)

    // Compiling filter
    if(pcap_compile(handle_, &filter_, filterSettings_.c_str(), 0, net_) == -1)
        error("Can't parse filter " + filterSettings_);

    // Setting up filter
    if(pcap_setfilter(handle_, &filter_) == -1)
        error("Can't set up filter " + filterSettings_);
```

3.4 Spouštění aplikace

Spouštění snifferu:

./myripsniffer -i <rozhraní>, kde význam parametru je následující:
* *-i*: *<rozhraní>* udává rozhraní, na kterém má být odchyt paketů prováděn.

Spouštění podvrhávače:

./myripresponse -i <rozhraní> -r <IPv6>/[16-128] {-n <IPv6>} {-m [0-16]} {-t [0-65535]}, kde význam parametrů je následující:
* *-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.

4. Použití vytvořených programů

Všechny zachycené packety:

RIPng

Destination: 33:33:00:00:00:09
Source: 08:00:27:94:99:12
IPv6 packet
IPv6 source: fe80:0000:0000:0000:0a00:27ff:fe94:9912
IPv6 destination: ff02:0000:0000:0000:0000:0000:0000:0009
Source port: 521
Destination port: 521
----RIPng----
Command: Response
Version: 1
----Entry no.0----
IPv6 prefix: fd00:0000:0000:0000:0000:0000:0000:0000
Route Tag: 0x00
Prefix Lenght: 64
Metric: 1
----Entry no.1----
IPv6 prefix: fd00:00d4:32a0:0000:0000:0000:0000:0000
Route Tag: 0x00
Prefix Lenght: 64
Metric: 1
----Entry no.2----
IPv6 prefix: fd00:0107:26aa:0000:0000:0000:0000:0000
Route Tag: 0x00
Prefix Lenght: 64
Metric: 1
----Entry no.3----
IPv6 prefix: fd00:08f0:0062:0000:0000:0000:0000:0000
Route Tag: 0x00
Prefix Lenght: 64
Metric: 1
----Entry no.4----
IPv6 prefix: fd00:09c0:1728:0000:0000:0000:0000:0000
Route Tag: 0x00
Prefix Lenght: 64
Metric: 1

RIPv1

Destination: 01:00:5e:00:00:09
Source: 08:00:27:94:99:12
IPv4 packet
IPv4 source: 10.0.2.4
IPv4 destination: 224.0.0.9
Source port: 520
Destination port: 520
----RIP----
Command: Response
Version: RIPv2
----Entry no.0----
Authentication type: Simple password
Authentication password: **ISA>27d15c08448**
----Entry no.1----
Address family: IP
Route tag: 0x0000
IP address: 10.0.0.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.2----
Address family: IP
Route tag: 0x0000
IP address: 10.48.52.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.3----
Address family: IP
Route tag: 0x0000
IP address: 10.104.202.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.4----
Address family: IP
Route tag: 0x0000
IP address: 10.114.101.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.5----
Address family: IP
Route tag: 0x0000
IP address: 10.215.108.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1

RIPv1

Destination: 01:00:5e:00:00:09
Source: 08:00:27:94:99:12
IPv4 packet
IPv4 source: 10.0.0.1
IPv4 destination: 224.0.0.9
Source port: 520
Destination port: 520
----RIP----
Command: Response
Version: RIPv2
----Entry no.0----
Authentication type: Simple password
Authentication password: **ISA>27d15c08448**
----Entry no.1----
Address family: IP
Route tag: 0x0000
IP address: 10.48.52.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.2----
Address family: IP
Route tag: 0x0000
IP address: 10.104.202.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.3----
Address family: IP
Route tag: 0x0000
IP address: 10.114.101.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1
----Entry no.4----
Address family: IP
Route tag: 0x0000
IP address: 10.215.108.0
Netmask: 255.255.255.0
Next hop: 0.0.0.0
Metric: 1

Při spočítání unikátních cest tedy dojdeme k závěru, že jsme odchytili různých 10 cest.

Zjištěné heslo pomocí snifferu: ISA>27d15c08448

Důkaz o úspěšném podvrhnutí:

5. Literatura

Routing Information Protocol. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-11-19]. Dostupné z: https://cs.wikipedia.org/wiki/Routing_Information_Protocol#Omezen%C3%AD

Routing Information Protocol: version 1 [online]. Rutgers University: Network working group, 1988 [cit. 2018-11-19]. Dostupné z: https://tools.ietf.org/html/rfc1058

Routing Information Protocol: version 2 [online]. Rutgers University: Network working group, 1998 [cit. 2018-11-19]. Dostupné z: https://tools.ietf.org/html/rfc2453

Routing Information Protocol: version 2 [online]. Ipsilon Networks, 1997 [cit. 2018-11-19]. Dostupné z: https://tools.ietf.org/html/rfc2080

Packet capture library. In: *Wireshark wikipedia* [online]. 1997 [cit. 2018-11-19]. Dostupné z: https://wiki.wireshark.org/libpcap