ISA projekt

Varianta

Nástroje monitorující a generující zprávy jednoduchých distance-vector protokolů

2018/2019

Konzultuje: Ing. Vladimír Veselý, Ph.D.
Riešiteľ: Jakub Senčák
Login: xsenca00

Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

Dátum: 19.11.2018

Obsah

---

1 O projekte

Našou úlohou bolo vytvoriť sniffer na RIP pakety, ktoré má za úlohu vhodnou formou zobrazovať používateľovi. Následne sme mali vytvoriť program, ktorý podvrhne RIP response paket a pomocou neho si vyskúšať podvrhnúť tento paket smerovaču. A ako bonus vytvoriť program, ktorý vztvorí RIP request paket.

---

2 Protokoly RIP(ng)

2.1 Krátky úvod do problematiky

Routing Information Protocol (RIP) je protokol založený na tzv. “Distance Vector algoritmoch”, ktorý používa ako hlavnú metriku počet skokov medzi smerovačmi, pričom maximálnou metrikou je 15 skokov (označované ako nekonečno). Smerovače si každých 30 sekúnd vzájomne vymieňajú smerovacie tabuľky, ktoré si udržujú po 180 sekúnd.

2.1.1 RIP

Prvá verzia (z roku 1988), ktorá sa používala bola popísaná v dokumente RFC 1058 a vychádzala z vtedy používaných techník. Na rozosielanie smerovacích tabuliek používa broadcast.

2.1.2 RIPv2

Druhá verzia protokolu RIP z roku 1998 popísaná v RFC 2453 sa snaží odstrániť nedostatky prvej verzie a zachováva spätnú kompatibilitu. Pridáva napr. možnosť autentifikácie a rozposiela multicastom.

2.1.3 RIPng

V krátkosti doplnila podporu pre IPv6 adresovanie.

---

3 Popis riešenia

3.1 Stručný popis návrhu aplikácie

Pred návrhom aplikácie som si naštudoval RFC týkajúce sa problematiky. Vyskúšal som si aj tutoriál na stránke https://www.tcpdump.org/pcap.html, z ktorého som na začiatku vychádzal. Pri návrhu som sa silne inšpiroval aj programami, ktoré sme mali k dospozícii ku predmetu ISA, najmä `/examples/pcap/sniffer-filter.c` a `/examples/pcap/sniffer.c`.

3.2 Popis implementácie

Pri implementácii využívam štruktúry nachádzajúce sa v hlavičkových súboroch `<netinet/*.h>` a `<pcap.h>`, ktoré využívam na ukladanie dát a navigáciu v rámci paketov.
V rámci funkcie main spracovávam parametre na vstupe.
`sniff(char *device, char *errbuf)` Pripravuje všetko potrebné na pre odpočúvanie na zariadení `device`. Na otvorenie rozhrania, nastavenie filtra a preberanie paketov využívam knižnicu libpcap.

Funkcia `pcap_loop` callback-om volá funkciu `process_packets`, ktorá pakety spracováva a vypisuje na štandatdný výstup.

3.3 Nedokončené

sniffer - mal by byť kompletný Response - viď. README Bonus - viď. README

---

4 Návod na použitie

4.1 Inštalácia

Stiahnite a rozbaľte balíček .tar.
spustite v príkazovej riadke `make`
Alternatívne, ak chcete vidieť nedokonalosti v implementácii použite `make hard` - na vlastné riziko, pravdepodobne nepreložiteľné :P.

4.2 myripsniffer

Program sa spúšťa pomocou príkazu:

```
./myripsniffer –i <rozhranie>
```

-i <rozhranie> je špecifikovanie, na ktorom rozhraní sa bude komunikácia odpočúvať.

POZOR: Je možné, že budete potrebovať práva správcu (`sudo`).

Aplikáciu je možné spúšťať aj bez parametru -i, v tom prípade sa ako rozhranie použije rozhranie, ktoré vráti funkcia `pcap_lookupdev( )`.
Pri testovaní som používal aj .pcap súbory, ktoré som načítal namiesto rozhrania. Pre takéto testovanie, je nutné odkomentovať časť kódu vo funkcii `sniff( )` [pozor na koncovú zátvorku!!!]. V tomto prípade stačí znovu aplikáciu preložiť a namiesto rozhrania použiť súbor.

```
./myripsniffer -i file.pcap
```

Pri načítaní zo súboru uloženého aplikáciou Wireshark sa výpis môže "pokaziť" - pravdepodobne načítam do konca paketov, v ktorých už nič nie je. Funkcionalitu v bežnej prevádzke to nijako neovplyvňuje.

Príklad výstupu:

```
\#1 at Mon Nov 19 22:21:58 2018
IP src = 10.0.0.1 at port 520
RIPv2   Request command
AddressFId:     0
Route tag:      0
Metric:         16
...
\---------------------------------
\#3 at Mon Nov 19 22:21:58 2018
IP src = fe80::a00:27ff:fe65:12f5 at port 521
RIPng   Request command
Prefix:         ::
Route tag:      0
Prefix length:  0
Metric:         16
\---------------------------------
...
\#5 at Mon Nov 19 22:21:59 2018
IP src = 10.0.0.1 at port 520
RIPv2   Response command
Auth. passwd: ISA>2811200801e
\---------------------------------
AddressFId:     2
Route tag:      0
Address:        10.48.48.0
Mask:           255.255.255.0
Next-hop:       0.0.0.0
Metric:         1
\---------------------------------
```

4.3 myripresponse

Program sa spúšťa pomocou príkazu:

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

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

---

5 Literatura

MALKIN, G. Routing Information Protocol. RFC 1058. https://tools.ietf.org/html/rfc1058
MALKIN, G. RIP Version 2. RFC 2453. https://tools.ietf.org/html/rfc2453
MALKIN, G. RIPng for IPv6. RFC2453. https://tools.ietf.org/html/rfc2080

GARCIA, L. M. Programming with Libpcap - Sniffing the Network From Our Own Application. Hacking magazine. http://recursos.aldabaknocking.com/libpcapHakin9LuisMartinGarcia.pdf ISSN 1733-7186

Li Q. IPv6 Advanced Protocols Implementation. [2007]

5.1 Ďalšie zdroje

Súbory na stránkach predmetu ISA: *suborypredmetuisa/examples/pcap/sniff.c*
tutoriály na stránkach tcpdump.org https://www.tcpdump.org/pcap.html
https://www.devdungeon.com/content/using-libpcap-c
A rozličné iné, z ktorých som čerpal minimálne.

5.2 Tutoriály

https://www.tcpdump.org/pcap.html
https://www.devdungeon.com/content/using-libpcap-c
http://www.omnisecu.com/cisco-certified-network-associate-ccna/difference-between-ripv1-and-ripv2.php

---

THE END