Projektová dokumentace

ISA Netflow exporter

Jakub Kuzník

xkuzni04

Úvod

Pro řešení projektu jsem zvolil jazyk C. Snažil jsem se o minimalistické řešení s důrazem na přehlednost kódu a smyslupným členěním zdrojového kódu do jednotlivých souborů. Pro získavání dat z paketů, jsem použil své existující řešení packeté-sniffer, které jsem vytvořil v rámci předmětu ipk: https://github.com/jakubKuznik/VUT-ipk-packet_sniffer

Teorie

Pro pochopení problematiky protokolu netflow, jsem čerpal především ze záznamu přednášky Dr. Grégra, dokumentace Cisco [1] a rfc3954. Velice problematický fakt byl ten, že rfc je napsáno z pohledu exportujícího routeru, který traffic spracovává v reálném čase. V našem případě jsem tedy musel, a to především v časových parametrech, velice improvizovat, abych simuloval činnost routeru.

Flow

Co je to vlastně flow? Pro pochopení problematiky Netflow exporteru je zásadní si uvědomit, co je to vlastně flow. Flow nám v podstatě uchovává informace o množině paketů, které sdílí společné vlastnosti. Společnými vlastnostmi myslíme: (zdrojový interface, zdrojovou ip adresu, cílovou ip adresu, protokol, tos, cílový a zdrojový port.).

Exporter a Collector

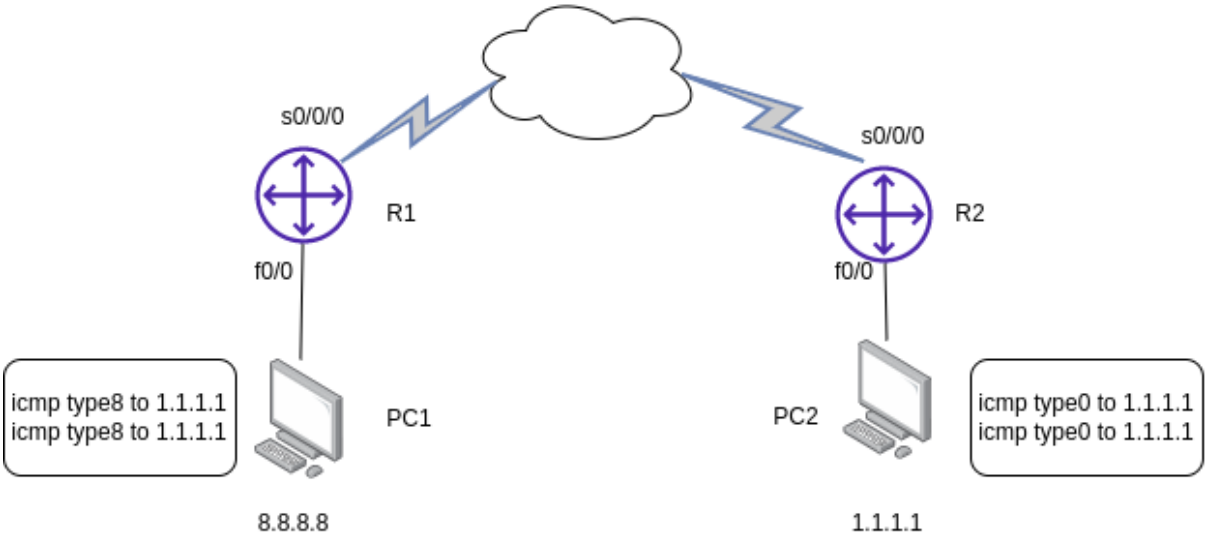
Co je to exporter a collector? Exporter vyrábí a uchovává databází jeho flows, která se dynamicky mění při odeslání flows na kolektor, čí pří příchodu nových neevidovaných paketů. Tyto flows se zapouzdřují do UDP paketů a jako UDP paket se posílají na kolektor.

Kolektor sbírá netflow záznamy a dále s nimi může pracovt. Např pomoci IPfix …

Praktická ukázka

Jak to vypadá prakticky? Ilustrace na jednoduchém příkladu:

Předpokladejme, že router R1 je Netflow exporter.
Počítač PC1 posílá 2 ICMP request packety počítači PC2, ten mu na ně odpoví dvěmi icmp response packety. V klasickém logovacím systému by jsme mohli zjistit, že přes R1 prošly 4 icmp packety bez vetších souvislostí. Jelikož je však R1 netflow exportér, tak dokáže zjistit, že pakety odeslány z PC1 maji veškeré vlastnosti stejné a Pakety odeslány z PC2 mají taky veškeré vlastnosti stejné. Vyrobí tedy pro jednotlivé dvojice flow záznam a ten nasledně odešle na kolektor.



Problém ukončení spojení. Z předchozího příkladu je zjevný jeden problém. Jak exportér pozná, jak dlouho má uchovávat flow, než ho odešle na kolektor a zahodí? U tcp se nabízí jedno logické řešení, a to odeslat flow v případě reset či fin flag. U bezstavových protokolů jako je icmp, čí udp se však musí zavést časovače. Existují 2 druhy časovačů. aktivní a neaktivní. Aktivní nám eviduje dobu od prvního paketu daneho flow, tedy dobu, kdy byl flow vytvořen. Neaktivní sleduje zda-li, do flow už nejakou delší dobu nepřibyl nový paket.

Formát paketu

Netflow je aplikační protokol, který má svou vlastní hlavičku a tělo. V této konkretní implementaci to je hlavička a tělo verze 5. Těchto hlaviček a těl paketů so do jednoho flow může vejít až 30, a to tak, že jsou uspořádaný bezprostředně za sebe. [2] Hlavička má celkem 24 Bajtů a tělo 48 Bajtů

Hlavička		
Bajty	Název	Popis
0-1	version	Verze protokolu netflow
2-3	count	Počet flowu v ramci daného paketu (1-30)
4-7	SysUptime	Čas v milisekundách od nabootovani zařízení
12-15	unix_secs	Čas v sekundách od UTC 1970
16-19	flow_sequence	Čas v nanosekundách od UTC 1970
20	engine_type	typ enginu
21	engine_id	id enginu
22-23	sampling_interval	první 2 bity určují vzorkovací mod a zbývajících 14 hodnotu vzorkovacího intervalu

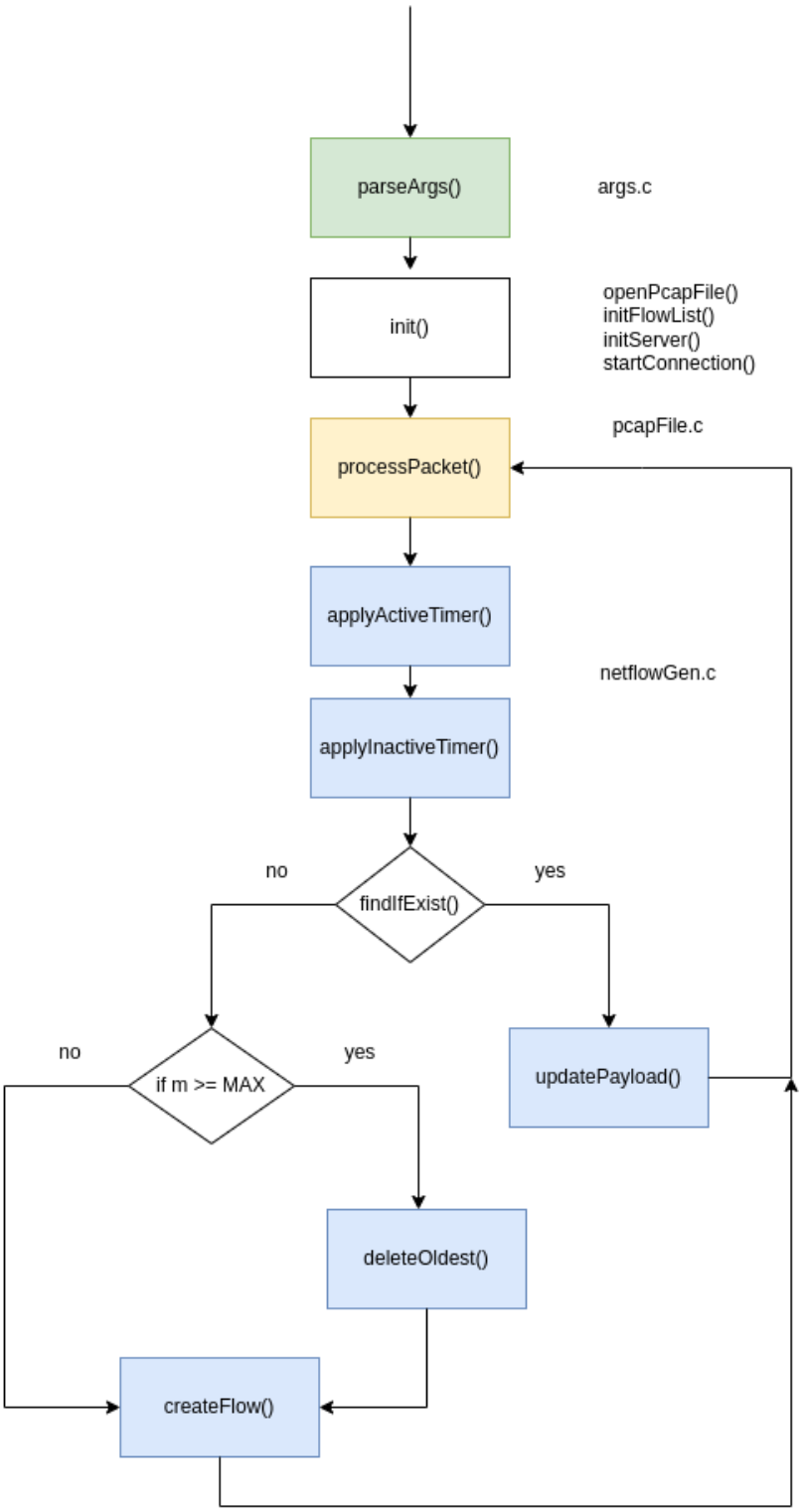
Tělo		
Bajty	Název	Popis
0-3	srcaddr	Zdrojová ip adresa
4-7	dstaddr	Cílová ip adresa
8-11	nexthop	Ip adresa nex-hop routeru
12-13	input	SNMP index vstupního rozhraní
14-15	output	SNMP index výstupního rozhraní
16-19	dPkts	Počet paketů ve flowu
20-23	dOctets	Celková velikost všech paketů ve flowu
24-27	First	Čas prvního paketu flowu
28-31	Last	Čas posledního paketu flowu
32-33	srcport	zdrojový port
34-35	dstport	cílový port
36	pad1	nepožívaný byte

37	tcp_flags	Kumulativní Or tcp příznaků
38	prot	protokol (TCP = 6, UDP = 17)
39	tos	Typ služby, hodnota získaná z hlavičky ip paketu
40-41	src_as	Číslo autonomního systému zdroje
42-43	dst_as	Číslo autonomního systému cíle
44	src_mask	Sítová maska zdroje v prefixovém formátu
45	dst_mask	Sítová maska cíle v prefixovém formátu
46-47	pad2	Nepoúžívaný byte

[2]

Schéma programu

Na následujícím diagramu je zobrazena základní zjednodušená algoritmická logika programu a soubory, které za dané chování zodpovídají. Hlavní konstrukce programu je cyklus, který neustále zpracovává jeden paket za druhým. Uvnitř cyklu se pak do interní struktury flowList ukládají záznamy o těchto paketech a v případě splnění nějaké podmínky se flow exportuje.



Popis implementace

V implementaci je kladen důraz nad přehlednost a jednoduchost, nikoli na optimalizace. Každá funkcionalita je oddělená v samostaném souboru. Za celou implementační logikou programu stojí double linked list, který obsahuje veškeré flows a udržuje si svojí velikost, a uživatel si tedy může dynamicky nastavit velikost paměti cache. V případě optimalizace by bylo vzhledem k neustálému vyhledávání vhodnější použit nějakou formu vyhledávací tabulky, např hash tabulka.

Datová struktura netFlow v sobě obsahuje další dvě struktury hlavičku a tělo paketu, které jsou navrženy přesně podle RFC395.

```
struct netFlow{
        NFheader *nfheader;         // hlavička netflows paketu
        NFPayload *nfpayload;       // tělo
}
struct flowList{
        uint32_t size;              // velikost Linked listu
        node *first;                // první uzel
        node *last;                 // poslední uzel
        node *current;              // aktuální uzel
}
struct node{
        netFlow *data;              // data flowu
        node *next;                 // předchozí flow
        node *prev;                 // nasledující flow
}
```

Souborová struktura

Program je napsán v jazyce C. Každý soubor s příponou .c má svůj hlavičkový soubor se stejným jménem.

Jméno souboru	Popis
flowc	Soubor s funkci main()
args.c	Má zodpovědnost za zpracování vstupních parametrů.
pcapFile.c	Zpracovává pakety a ziskavá z nich informace, které ukládá do struktury packetInfo.
udp.c	Zajištuje odesílání paketů na collector.

netflowGen.c	Pracuje s flows, čili zajišťuje ukládání a editaci v rámci flow datových struktur.
structurik.h	Obsahuje veškeré datové struktury programu.
Makefile	Makefile projektu
manual.pdf	Tento soubor, obsahuje dokumentaci projektu.
flow.1	Manualova stranka

Knihovny

Přehled využívaných knihoven se nachází v souboru **flow.h**. Veškeré informace o jednotlivých knihovnách jsou dostupné v rámci zdrojových kódu daných knihoven či manuálových stránek. [16]

Jméno knihovny	Využití v projektu
pcap.h	Funkce pro načítání rámce z rozhraní. Pomocí **pcap_next()** se vždy načte jeden rámec a ten se dále zpracuje. Dále struktura **pcap_header**, z nichž jednoduše zjistíme základní informace o daném rámci. Funkce, která najde veškeré síťové rozhraní **pcap_findalldevs()** a vrátí jejich seznam, či **pcap_open_live()** ta otevře zadané rozhraní. Pomocí **pcap_datalink()** zjistíme zda-li dané rozhraní podporuje rámce typu ETHERNET.
arpa/inet.h	Poskytuje datové typy pro ip adresy, ale především funkce **ntohs()** či **ntohl()** ty zamezí problémům v případě odlišných počítačových architektur (little endian, big endian)
netinet/if_ether.h	Obsahuje datovou strukturu **ether_header** pro práci s hlavičkou ethernetových rámců a makra, která pomáhají s práci s daty obsaženými v dané hlavičce.
netinet/ip_icmp.h	Obsahuje datovou strukturu **icmp** pro práci s hlavičkou icmp a makra, která pomáhají s práci s daty obsaženými v dané hlavičce. Například makra pro rozpoznání typu zprávy.
netinet/tcp.h	Obsahuje datovou strukturu **tcphdr** pro práci s TCP hlavičkou
netinet/udp.h	Obsahuje datovou strukturu **udphdr** pro práci s UDP hlavičkou
netinet/arp.h	Obsahuje datovou strukturu **arphdr** pro práci s hlavičkou arp a makra, která pomáhají s práci s daty obsaženými v dané hlavičce. Například makra pro rozpoznání typu zprávy.

Návod na použití

./flow [-f <file>] [-c <netflow_collector>[:<port>]] [-a <active_timer>] [-i <inactive_timer>][-m <count>]

Veškere parametry jsou volitelné v případě nezadaní vstupního souboru program čte ze standartního vstupu.

- -f <file>
    - jméno analyzovaného souboru nebo STDIN
- -h –help
    - Vypíše nápovědu
- -c <netflow_collector:port>
    - IP adresa, nebo hostname NetFlow kolektoru. volitelně i UDP port (127.0.0.1:2055, pokud není specifikováno),
- -a <active_timer>
    - Zapnutí protokolu tcp - pakety tcp budou zobrazeny a nebudou přeskočeny
- -i <seconds>
    - interval v sekundách, po jehož vypršení se exportují neaktivní záznamy na kolektor (10, pokud není specifikováno),
- -m <count>
    - velikost flow-cache. Při dosažení max. velikosti dojde k exportu nejstaršího záznamu v cachi na kolektor (1024, pokud není specifikováno).

Zdroje:

[1] NetFlow Export Datagram Format - Cisco. *Cisco - Networking, Cloud, and Cybersecurity Solutions* [online]. Dostupné z: https://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/3-6/user/guide/format.html
[2] rfc3954
Zdroje pro sestrojeni packet-snifferu:
[10] Ethernet frame [online]. [cit 22.4.2022] Dostupné z: https://en.wikipedia.org/wiki/ethernet_frame odstavec ethernet II
[11] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 4.4.2 ipv4 Addresing
[12] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 4.4.4 ipv6
[13] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 3.5.2 TCP Segment Structure
[14] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 3.3.1 UDP Segment Structure
[14] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 5.4.1 Link-Layer Addressing and ARP
[15] Kurose, James F. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. p. cm. Includes bibliographical references and index. ISBN-13: 978-0-13-285620-1 ISBN-10: 0-13-285620-4 4.4.3 Internet Control Message Protocol.
[16] <netinet/>. *The Open Group Publications Catalog* [online]. Copyright © 1997 The Open Group [cit. 22.04.2022]. Dostupné z: https://pubs.opengroup.org/onlinepubs/7908799/xns/netinetin.h.html