VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

GENEROVÁNÍ NETFLOW DAT ZE ZACHYCENÉ SÍŤOVÉ KOMUNIKACE

GENERATION OF NETFLOW DATA FROM CAPTURED NETWORK COMMUNICATION

PROJEKTOVÁ DOKUMENTACE
PROJECT DOCUMENTATION

AUTOR PRÁCE — **TOMÁŠ BÁRTŮ**
AUTHOR

VEDOUCÍ PROJEKTU — **Ing. MATĚJ GRÉGR, Ph.D.**
SUPERVISOR

BRNO 2022

Obsah

Kapitola 1

Úvod

1.1 Obecné informace

Cílem projektu do ISA — Síťové aplikace a správa sítí, bylo implementovat NetFlow exportér, který ze zachycených síťových dat, které jsou uloženy ve formě pcap souboru, vytvoří NetFlow záznamy, jenž jsou následně odeslány na cílový NetFlow kolektor.

1.2 Upřesnění

Zadání specifikuje podporu generování toků pouze z protokolů TCP, UDP a ICMP. Tento projekt podporuje pouze NetFlow ve verzi 5. Informace, které nebudou známé z analyzovaného pcap souboru, budou mít nastavenou nulovou hodnotu jako například informace src_as, input, output a jiné. Při zpracovávání a následném generování záznamu mají být použity časové značky ze zachycené komunikace respektive časové značky ze zdrojového pcap souboru. Následná vyexportovaná data musí být čitelná nástrojem `nfdump`.

Kapitola 2

Teorie

2.1 Monitorování počítačových sítí

V současných sítích se již stalo standardem sledovat a analyzovat síťový provoz a každý správný správce sítě by se měl dozvědět o případných problémech na síti dříve než samotný uživatel této sítě. K tomuto účelu existují určité nástroje a přístupy jako například protokol ICMP. Pro vytváření dlouhodobějších statistik o provozu na síti lze použít například protokoly SNMP nebo RMON, které slouží k získání statistik o provozu na daném zařízení nebo lze pro získání statistických údajů toků lze použít IPFIX, OpenFlow a již zmíněný NetFlow [7].

2.2 Síťový tok

Co je to vlastně síťový tok? Dle definice je tok *posloupnost paketů, které mají společnou vlastnost a které prochází bodem pozorování za určitý časový interval* [4]. Každý ze záznamů obsahuje informace o toku. Jeho obsahem je typicky zdrojová a cílová IP adresa, zdrojový a cílový port, typ použitého protokolu, datum a čas zaznamenání nebo počet přenesených dat jako například počet přenesených bajtů či počet zaznamenaných paketů.

2.3 NetFlow architektura

Původně byl vyvinut formou Cisco avšak později byl standardizován. NetFlow formát umožňuje analyzovat a ukládat informace o provozu na síťovém zařízení. Mezi základní prvky patří exportér, komunikační protokol NetFlow, kolektor a také nástroje, které umožňují zachycená data zobrazit i v grafické formě. [6].

2.4 NetFlow exportér

Jedná se o sondu/router či software, který dokáže monitorovat síťový provoz. Ze zachycené komunikace vytváří záznamy o toku neboli `Flow Records`. Při zachycení komunikace respektive paketu vytvoří nový záznam nebo aktualizuje záznam, který je již uložen ve vyrovnávací pamětí neboli `NetFlow cache`. Aktualizace záznamu je prováděna dle určitých pravidel.

2.4.1 Funkcionalita

Jak již bylo zmíněno, tak se při nově zachycené komunikaci vytvoří nový záznam nebo se aktualizuje respektive agreguje s jiným již vytvořeným záznamem. Z komunikace si nejdříve zjistí klíčové parametry potřebné pro odlišení jednotlivých záznamů. Standardně jsou klíčové údaje NetFlow Cisco verze 5 založeny na pětici či sedmici hodnot získané z analyzované komunikace. Tyto hodnoty můžeme jednotně nazvat jako klíč, jenž nám umožňuje rozlišit, zda se má vytvořit nový záznam, či se má zpracovávaný záznam agregovat a aktualizovat hodnoty u již vytvořeného v paměti [5]. Přičemž jednotlivé atributy sedmice jsou:

- Zdrojové rozhraní
- Zdrojová IP adresa
- Cílova IP adresa
- Protokol
- ToS
- Zdrojový port
- Cílový port

Jednotlivé atributy pětice jsou shodné s atributy sedmice. V pětici se ale nevyskytuje atribut ToS a atribut označující zdrojové rozhraní.

2.4.2 Export záznamů

Pokud je záznam označen za ukončený dojde k jeho odstranění z NetFlow cache. K odeslání dochází za podmínek [8]:

- Tok je neaktivní — detekce dle neaktivního časovače
- Tok je v paměti příliš dlouho — detekce dle aktivního časovače
- NetFlow cache je zaplněna
- Byl detekován konec toku — například příznaky FIN nebo RST u TCP komunikace

Pokud nastane nějaká z možností výše, tak dojde k vyexportování takto určených záznamů. Tyto záznamy jsou poslány na kolektor. Hlavička a záznam NetFlow paketu verze 5 je reprezentován tabulkami 2.1 a 2.2.

2.5 Kolektor

Jedná se o software jehož činnost spočívá v přijímání NetFlow paketů z jednoho nebo více exportérů. Z paketů si zjistí jednotlivé záznamy, které pak následně může zobrazit jako grafický výstup na monitor nebo je uloží na disk či do databáze. Je zde možnost agregace jednotlivých záznamů. Open source kolektor je například NFsen nebo NFDUMP.

Bytes	**Contents**	**Description**
0-1	version	NetFlow export format version number
2-3	count	Number of flows exported in this packet (1-30)
4-7	SysUptime	Current time in milliseconds since the export device booted
8-11	unix_secs	Current count of seconds since 0000 UTC 1970
12-15	unix_nsecs	Residual nanoseconds since 0000 UTC 1970
16-19	flow_sequence	Sequence counter of total flows seen
20	engine_type	Type of flow-switching engine
21	engine_id	Slot number of the flow-switching engine
22-23	sampling_interval	First two bits hold the sampling mode; remaining 14 bits hold value of sampling interval

Tabulka 2.1: Formát NetFlow verze 5 hlavičky. Převzato z [3]

Bytes	Contents	Description
0-3	srcaddr	Source IP address
4-7	dstaddr	Destination IP address
8-11	nexthop	IP address of next hop router
12-13	input	SNMP index of input interface
14-15	output	SNMP index of output interface
16-19	dPkts	Packets in the flow
20-23	dOctets	Total number of Layer 3 bytes in the packets of the flow
24-27	First	SysUptime at start of flow
28-31	Last	SysUptime at the time the last packet of the flow was received
32-33	srcport	TCP/UDP source port number or equivalent
34-35	dstport	TCP/UDP destination port number or equivalent
36-37	pad1	Unused (zero) bytes
38	prot	IP protocol type (for example, TCP = 6; UDP = 17)
39	tos	IP type of service (ToS)
40	flags	Cumulative OR of TCP flags
41-43	pad1, pad2, pad3	Unused (zero) bytes
44-48	reserved	Unused (zero) bytes

Tabulka 2.2: Formát NetFlow verze 5 záznamu. Převzato z [3]

Kapitola 3

Implementace

NetFlow exportér je implementován v jazyce C++. Překlad probíhá pomocí standardu 14 respektive tedy C++14. Program podporuje pouze NetFlow ve verzi 5. Samotná implementace exportéru je uložena v následujících souborech:

- flow.cpp, flow.hpp — soubor obsahující funkci main() s voláním podstatných funkcí
- arguments.cpp, argument.hpp — ošetření argumentů programu
- pcap.cpp, pcap.hpp — analýza a zpracování pcap souboru
- exporter.cpp, exporter.hpp — UDP klient odesílající zpracovaná data
- Makefile — překlad programu + mazání přeložených souborů + valgrind

3.1 Začátek běhu programu

Funkce *main()* se nachází v souboru `flow.cpp`. V níž dojde nejdříve k zavolání kontroly argumentů *parse_args()* a následně se ze získanými argumenty zavolá funkce *pcapInit()* pro analýzu zdrojové síťové komunikace. Podstatnou částí je struktura `options` uchovávající jednotlivé argumenty programu, přičemž obsahuje jejich implicitní hodnoty.

```
struct options
{
    const char*     file     {"-"};   // souborový popisovač
    struct hostent* hostent  {};      /* struktura uchovavájící mimo
                                         jiné cílovou IP adresu */
    unsigned long   port     {2055};  // číslo portu
    unsigned long   ac_timer {60};    // aktivní časovač
    unsigned long   in_timer {10};    // neaktivní časovač
    unsigned long   count    {1024};  // velikost cache
};
```

Úsek kódu 3.1: Struktura pro uchování hodnot argumentů programu.

3.2 Zpracování argumentů

Implementace se nachází v souboru `arguments.cpp`. Hlavní funkce pro zpracování argumentů 4.1 je *parse_args()*, kde se pomocí knihovní funkce *getopt()* určí jednotlivé přepínače a jejich parametry se následně uloží do struktury `options` 3.1. Pokud je nějaký z přepínačů zadán vícekrát, tak se uvažuje hodnota posledního zadaného.

3.3 Zpracování vstupního souboru

Klíčovou částí je funkce *pcapInit()*, kde se nejdříve otevře pcap soubor pro zpracování. Následně se vytvoří filtr, aby při následném zpracování byla uvažovány pouze komunikace pomocí protokolů UDP, TCP a ICMP. Následně se začne periodicky vykonávat funkce *pcap_loop()*, která z pcap souboru získává jednotlivé pakety, dokud existuje nějaký další paket k načtení. Při každém načtení nového paketu se pomocí *"callbacku"*[1] volá funkce, ve které dochází k analýze onoho paketu. Po přečtení všech paketů se vyexportují zbylé záznamy z NetFlow cache [2].

3.3.1 Callback

V programu mu odpovídá funkce *handler()*, ve které dochází ke zjištění všech potřebných hodnot z hlaviček jednotlivých protokolů, více viz [1]. Dle 2.4.1 se standardně používá sedmice nebo pětice klíčových hodnot pro rozlišení toků. V této implementaci se uvažuje uvedená pětice + ToS respektive:

- Zdrojová IP adresa
- Cílova IP adresa
- Protokol
- ToS
- Zdrojový port
- Cílový port

Jednotlivé NetFlow záznamy jsou ukládány do mapy, přičemž klíčem je uvedená šestice uložená ve struktuře `tuple` a každému z klíčů odpovídá jeden NetFlow záznam. Ten je reprezentován strukturou `NetFlowRCD`, která odpovídá jednotlivým položkám, viz 2.2.

3.3.2 Vytváření a agregace záznamů

Posléze, co se rozliší protokol (UDP, TCP nebo ICMP) paketu, tak se vytvoří `tuple` se všemi klíčovými hodnotami, který se následně vyhledá v mapě *(mapa simuluje NetFlow cache)*. Pokud se v mapě nenachází, vytvoří se nový záznam v NetFlow cache a pokud se zde klíč nachází, tak dojde pouze k aktualizaci odpovídajících položek a to konkrétně počtu paketů, velikosti celkově zachycených dat v záznamu a času položky *Last*. V případě, že se jedná o protokol TCP, tak se navíc aplikuje na položku uchovávající příznaky bitová

[1] Úsek spustitelného kódu, který je předán jako argument jinému kódu, který ve vhodnou dobu spustí onen předaný úsek

operace *OR*. Pro ICMP komunikaci je nastaven zdrojový port jako nulový a do položky cílového portu se ukládá zakódovaná hodnota ICMP typu a kódu, které se počítá jako:

$$cilovyPort = TypICMP * 256 + KodICMP$$

3.3.3 Označení záznamů k exportování

Při každém začátku funkce *handler()* dojde ke kontrole, zde není NetFlow cache zaplněná nebo zda mapa neobsahuje záznamy, které by měly být označeny k exportování, více viz 2.4.2. Tato kontrola probíhá ve funkcích *chechkSize()* a *checkTimers()*. První zmíněná funkce slouží ke kontrole, zda je NetFlow cache zaplněná. Pokud ano, tak je označen k exportování nejstarší záznam z mapy. Tato funkce je volána pokaždé, než se má vložit nový záznam do mapy. Kontrola jednotlivých časovačů ve druhé ze zmíněných funkcí probíhá následovně:

```
for (auto &iterator: m) {                    // kde m je NetFlow cache
    if (getUptimeDiff(h.ts) - ntohl(iterator.second.First)
                >= options.ac_timer * 1000) { // aktivní časovač
        queue.emplace_back(iterator);         // označení k exportu
    } else if (getUptimeDiff(h.ts) - ntohl(iterator.second.Last)
                >= options.in_timer * 1000) { // neaktivní časovač
        queue.emplace_back(iterator);         // označení k exportu
    }
}
```

Úsek kódu 3.2: Kontrola aktivního a neaktivního časovače vůči jednotlivám záznamům.

Kde první podmínka reprezentuje aktivní časovač (v milisekundách):

$$SysUptimeAktualihoPktu - SysUptimePosledniAktualizacePktu >= aktivniCasovac$$

a druhá podmínka reprezentuje neaktivní časovač (v milisekundách):

$$SysUptimeAktualnihoPktu - SysUptimePrvnihoVyskytuPktu >= neaktivniCasovac$$

Přičemž označením `SysUptime` rozumíme čas od startu systému do určitého času. V našem případě start systému odpovídá zpracování úplně prvního paketu pcap souboru a jeho `timeval` struktuře nacházející se v pcap hlavičce. Čas reprezentovaný touto strukturou se pak následně používá jako referenční hodnota ve funkci *getUptimeDiff*. V této funkci se vypočítá `SysUptime` z již zmíněné referenční hodnoty a ze skutečného parametru funkce, který odpovídá času právě zpracovávaného paketu. Při výpočtu dochází k případnému zaokrouhlení.

Záznam je také označen k exportování pokud se u TCP komunikace detekuje příznak `RST` nebo `FIN`.

3.3.4 Odesílání dat exportéru

Záznamy označené k odeslání jsou zpracovávány a následně odeslány k exportování ve dvou funkcích a to v *export_queue_flows()* a *export_rest_flows()*. V první z nich se odesílají data na exportér na základě splnění určité podmínky, viz 3.3.3. Ve druhé na základě detekce konce pcap souboru, tj. ukončení *pcap_loop()*. U obou z nich dochází k postupnému odebírání NetFlow záznamů z určité struktury. U první možnosti se to týká mapy a fronty. Ve frontě jsou uloženy záznamy označené k exportování. Z této fronty se získávají klíče do mapy, které určují, jaké záznamy mají být odeslány na exportér a odstraněny z NetFlow cache. Druhou možností je konec zpracovávaného souboru, kde se z NetFlow cache odstraní všechny záznamy tím, že se odešlou na exportér. V obou funkcích se pak následně vytvoří NetFlow hlavička, viz 2.1 a společně se záznamy se spojí do struktury jednoho paketu.

```
#define NETFLOW_MAX_EXPORTED_PACKETS 30

struct NetFlowPacket{
    NetFlowHDR netFlowHdr;
    NetFlowRCD netFlowRcd[NETFLOW_MAX_EXPORTED_PACKETS];
};
```

Úsek kódu 3.3: Struktura NetFlow verze 5 paketu

Ze struktury je patrné, že obě z funkcí realizují odesílání paketů na exportér, umožňují odeslat `maximálně 30` záznamů v jednom NetFlow paketu.

3.4 Exportér

Funkcionalita exportéru je implementovaná v souboru `exporter.cpp`. Jedná v jednoduchosti o UDP klienta odesílajícího nějaká data (zde NetFlow pakety). Podstatnou částí, která nám zajistí to, aby byly správně odeslány všechny *toky* na kolektor, je samotný výpočet odesílaných dat, který je počítán jako:

$$velikostNetFlowHlavicky + pocetOdesilanychFlowu * velikostNetFlowZaznamu$$

Struktura NetFlow paketu určeného k odeslání se pomocí funkce *memcpy()* vloží do bufferu. Velikost bufferu, tedy maximální velikost přenesených dat přes protokol NetFlow verze 5, jde spočítat jako maximální počet toků v jednom paketu krát velikost jednoho toku plus velikosti hlaviček NetFlow, IPv4 a Ethernetu (uvažujeme maximální velikost IP hlavičky). Výsledná velikost odpovídá tomuto výpočtu: $30 * 48 + 24 + 60 + 14 = 1538$. Velikost bufferu musí být tedy větší než spočtená hodnota. V programu tato velikost odpovídá nejbližší větší mocnině dvou tedy velikosti 2048 bajtů. Následně vytvoříme soket typu `SOCK_DGRAM`, nastaví se UDP schránku pomocí funkce *connect()* a NetFlow paket se posléze odešle na kolektor. Jelikož se neví, jestli se ještě budou odesílat nějaké další pakety na kolektor, tak uzavřeme soket [9].

Kapitola 4

Návod na použití

4.1 Softwarové požadavky

- GNU překladač g++ (použitá verze 11.3.0)
- GNU Make (použitá verze 4.3)

4.2 Překlad programu

Soubory projektu lze přeložit následujícím příkazem.

```
$ make
```

V souboru Makefile jsou dále k dispozici cíle *clean* a *valgrind*. Přičemž cíl *clean* vymaže z adresáře všechny soubory vytvořené překladem. Cíl *valgrind* zobrazí na standardní výstup svůj výpis z něho puštěného programu *flow*, přičemž se použijí implicitní hodnoty. Tudíž je vhodné například zdrojový soubor přesměrovat na standardní vstup spuštěného programu.

Vymazání přeložených souborů.

```
$ make clean
```

Spuštění valgrindu se zdrojovým souborem *file.pcap*.

```
$ make valgrind < file.pcap
```

4.3 Argumenty programu

V tabulce jsou zobrazeny jednotlivé přepínače a jejich popis společně s jejich základními hodnotami.

Argument	Popis
-f <soubor>	Jméno analyzovaného souboru nebo standardní vstup. Implicitní hodnota: STDIN.
-c <kolektor:<port»	IP adresa nebo doménové jméno NetFlow kolektoru. Volitelně i číslo portu. Implicitní hodnota: 127.0.0.1:2055.
-a <aktivní časovač>	Interval v sekundách, po kterém se exportují aktivní záznamy na kolektor. Implicitní hodnota: 60.
-i <neaktivní časovač>	Interval v sekundách, po jehož vypršení se exportují neaktivní záznamy na kolektor. Implicitní hodnota: 10.
-m <počet>	Velikost flow-cache. Při dosažení maximální velikosti dojde k exportu nejstaršího záznamu v cachi na kolektor. Implicitní hodnota: 1024.

Tabulka 4.1: Výpis argumentů programu s jejich významem.

4.4 Syntax pro spuštění

Program při spouštění podporuje následující syntax. Jednotlivé přepínače jsou blíže popsány v sekci 5.3.

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

4.5 Příklady spouštění

Program lze spustit dle 5.4. Několik ze způsobů spuštění je názorně zobrazeno a popsáno níže. Je možné kombinovat jednotlivé argumenty. Avšak pokud se použijí argumenty opakovaně, tak se konečná hodnota bude rovnat hodnotě při posledním výskytu zadaného argumentu programu.

Zdrojová komunikace se nachází v souboru *file.pcap*.

```
$ ./flow –f file.pcap
```

IP adresa NetFlow kolektoru je *127.0.0.1*.

```
$ ./flow –c 127.0.0.1
```

Doménové jméno NetFlow kolektoru je *localhost*.

```
$ ./flow –c localhost
```

IP adresa kolektoru je *127.0.0.1* s cílovým číslem portu *4096*.

```
$ ./flow –c 127.0.0.1:4096
```

Doménové jméno NetFlow kolektoru je *localhost* s cílovým číslem portu *4096*.

```
$ ./flow –c localhost:4096
```

Interval v sekundách, po kterém se mají aktivní NetFlow záznamy vyexportovat na kolektor, je *120*.

```
$ ./flow –a 120
```

Interval v sekundách, po kterém se mají neaktivní NetFlow záznamy vyexportovat na kolektor, je *5*.

```
$ ./flow –i 5
```

Velikost vyrovnávací paměti pro NetFlow záznamy (flow-cache) je *256* záznamů.

```
$ ./flow –m 256
```

Kapitola 5

Závěr

Testování výsledné implementace probíhalo pomocí programu `nfcapd` a `nfdump` z řad nástrojů NFDUMP a pomocí programu softflowd. Pro kontrolu informací ze zachycené komunikace byl použit program WireShark. Program `nfcapd`, který reprezentoval kolektor, sloužil k zachytání odesílaných NetFlow paketů. Kontrolní výsledky byly generovány programem softflowd. Vygenerovala se tedy komunikace ze stejného pcap souboru jak přes `softflowd` tak přes vlastní implementaci. Porovnávání jednotlivých výsledků probíhalo pomocí programu `nfdump`. Byla zjištěna občasná odlišnost v jednotlivých časech v řádech tisícin sekund, které mohou být způsobeny zaokrouhlováním při převodu jednotek. Dále byly zjištěny občasné rozdíly v pořadí toků, ale celkové statistky bylo shodné. Důvodem je rozdílná implementace exportní strategie u obou programů.

Literatura

[1] Bártů, T. *IPK-project2* [online]. 1.0.0. Listopad 2022 [cit. 2022-11-10]. Jedná se o implementaci projektu z IPK — Počítačové komunikace a sítě. Dostupné z: https://github.com/paetricc/IPK-project2.

[2] Carstens, T. Programming with pcap. *TCPDump&LibPcap* [online]. 2002 [cit. 2022-11-10]. Dostupné z: https://www.tcpdump.org/pcap.html.

[3] NetFlow Export Datagram Format. *Cisco* [online]. 14.9.2007 [cit. 2022-11-09]. Dostupné z: https://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/3-6/user/guide/format.html.

[4] Claiss, B. Cisco Systems NetFlow Services Export Version 9. *IETF RFC 3954* [online]. Říjen 2004 [cit. 2022-11-09]. Dostupné z: https://www.ietf.org/rfc/rfc3954.txt.

[5] Daniel. NetFlow. *Wayback Machine* [online]. Říjen 2004. 17.7.2017 14:20 [cit. 2022-11-09]. Dostupné z: https://web.archive.org/web/20170222053806/https://pliki.ip-sa.pl/wiki/Wiki.jsp?page=NetFlow.

[6] Laštovička, M. *Dolování IP charakteristik z NetFlow*. Brno, CZ, 2014. Bakalářská práce. Masarykova univerzita v Brně, Fakulta informatiky. Dostupné z: https://is.muni.cz/th/g2z4z/prace.pdf.

[7] Matoušek, P. *Síťové aplikace a jejich architektura*. Brno: VUTIUM, 2014. ISBN 978-80-214-3766-1.

[8] Matoušek, P. *Síťové aplikace a správa sítí: Monitorování toků NetFlow* [Univerzitní přednáška]. 2021 [cit. 2022-11-09].

[9] Matoušek, P. *Síťové aplikace a správa sítí: Pokročilé programování sítí TCP/IP* [Univerzitní přednáška]. 2022 [cit. 2022-11-09].