VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ANONYMIZACE UŽIVATELŮ PŘI SBĚRU DAT O SÍŤOVÉM PROVOZU
ANONYMIZATION OF USERS WHEN COLLECTING NETWORK TRAFFIC

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Lukáš Hamár
AUTHOR

VEDOUCÍ PRÁCE — Ing. Marek Sikora
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Lukáš Hamár ***ID:*** 186076
Ročník: 3 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Anonymizace uživatelů při sběru dat o síťovém provozu

POKYNY PRO VYPRACOVÁNÍ:

Bakalářská práce je zaměřena na možnosti skrytí identity koncových uživatelů a jejich IP adres při analýze síťového provozu a tvorbě statistik. Úkolem bakalářské práce je navrhnout a ověřit postup, jenž zajistí získání anonymních dat pro tvorbu statistik.

DOPORUČENÁ LITERATURA:

[1] GUERRA PEREZ, K.; YANG, X.; SCOTT-HAYWARD, S.; SEZER, S. Feature study on a programmable network traffic classifier, (2017), International System on Chip Conference, art. no. 7905446, pp. 108-113. DOI: 10.1109/SOCC.2016.7905446.

[2] CLARKE, N.; LI, F.; FURNELL, S. A novel privacy preserving user identification approach for network traffic, (2017), Computers and Security, 70, pp. 335-350. DOI: 10.1016/j.cose.2017.06.012.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 29.5.2018

Vedoucí práce: Ing. Marek Sikora
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto práca sa zaoberá anonymizáciou dát, ktoré by mohli viesť k odhaleniu identity koncových užívateľov v sieťovej prevádzke. Práca popisuje algoritmy, pomocou ktorých sú anonymizované jednotlivé častí dát a taktiež popisuje nástroje, ktoré spomenuté techniky využívajú na anonymizáciu sieťovej prevádzky. V ďalšej časti je popísané zostrojenie vlastnej laboratórnej siete, v ktorej je zachytávaná sieťová prevádzka obsahujúca dáta formátu pcap aj NetFlow. S využitím týchto dát sú testované anonymizačné programy a jednotlivé výsledky sú medzi sebou porovnané. V poslednej časti práce je vytvorené grafické prostredie pre jeden z testovaných anonymizačných programov.

KĽÚČOVÉ SLOVÁ

anonymizačné techniky, anonymizačné nástroje, generátor sieťovej prevádzky, grafické prostredie, IP adresa, monitorovanie siete, NetFlow, ochrana osobných údajov, pcap, Scrub-tcpdump, sieťová prevádzka, tcpdump, Tkinter, virtualizácia serveru, virtuálna sieť, Wireshark

ABSTRACT

This thesis deals with anonymization of data, which could lead to disclosure of the identity of end users in network traffic. Work describes algorithms by which individual data parts are anonymized and also tools which use these techniques for network traffic anonymization. The next part of the thesis describes construction of a laboratory network, in which is the network traffic captured, containing pcap and NetFlow data. With using of the captured data, the anonymization tools are tested and the results are compared. In the last part of the thesis is created graphical interface for one of the tested anonymization softwares.

KEYWORDS

anonymization techniques, anonymization tools, graphical interface, IP address, network traffic, network traffic generator, NetFlow, network monitoring, privacy protection, pcap, server virtualization, Scrub-tcpdump, tcpdump, Tkinter, virtual network, Wireshark

HAMÁR, Lukáš. *Anonymizace uživatelů při sběru dat o síťovém provozu*. Brno, 2018, 62 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Marek Sikora

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Anonymizace uživatelů při sběru dat o síťovém provozu“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Marekovi Sikorovi, za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

Spracovanie a analýza dát v sieti má vo svete informačných technológií opodstatnený význam. S neustálym pribúdaním koncových zariadení narastá záťaž sieťovej infraštruktúry, preto je potrebné zväčšovať ju a implementovať nové technológie. Na monitorovanie, zber dát a následné tvorenie štatistík sa používajú rôzne prostriedky a nástroje ako sú napríklad protokol SNMP, NetFlow, IPFIX a iné. Na základe vytvorených štatistík je možné posúdiť zaťaženie siete, možné vonkajšie alebo aj vnútorné hrozby a tým predísť možnému kolapsu sieťovej infraštruktúry.

Pri tvorení štatistík sú používané dáta, ktoré môžu obsahovať citlivé informácie koncových užívateľov. Pomocou IP adresy a ďalších identifikátorov je možné odhaliť identitu užívateľa, čo môže viesť k zneužitiu citlivých informácií a osobných údajov. Preto sa vynára otázka bezpečnosti a anonymizácie užívateľov v súvislosti so zberom dát vo verejnej sieti. Na anonymizáciu dát sa používajú viaceré techniky a nástroje. Pri výbere anoynmizačnej techniky je potrebné zvážiť, ktoré dáta je vhodné anonymizovať aby získaná štatistika siete mohla poskytnúť potrebné informácie, pomocou ktorých je možné sledovať a analyzovať stav sieťovej infraštruktúry.

Prvá kapitola bakalárskej práce je venovaná nástrojom, ktoré sa používajú na monitorovanie siete alebo zachytávanie sieťovej komunikácie a následnú analýzu. Táto časť sa venuje aj ochrane osobných údajov. Opisuje smernicu, ktorá v Českej republike povoľuje plošný zber a uchovávanie dát. Taktiež je v tejto časti vysvetlené nariadenie, ktoré vstúpi do platnosti v máji 2018 a bude sa týkať ochrany osobných údajov.

Druhá kapitola opisuje samotnú anonymizáciu. Popisuje jednotlivé anonymizačné techniky, akým spôsobom a ktorú časť poľa TCP/IP paketu dokážu anonymizovať. Následne sú popísané jednotlivé anonymizačné nástroje, ktoré sú roztriedené podľa formátu dát, s ktorým dokážu pracovať.

Tretia kapitola opisuje zostrojenie vlastnej laboratórnej siete. Následne popisuje krok za krokom postup, akým boli jednotlivé prvky siete upravované, aby boli schopné generovať a zachytávať dáta sieťovej prevádzky vo formáte *pcap* a NetFlow.

Štvrtá kapitola sa zaoberá testovaním anonymizačných programov a jednotlivých techník, s využitím sieťovej prevádzky získanej z vytvorenej laboratórnej siete. Je tu popísaný postup ako pri jednotlivých programoch docieliť proces anonymizácie dát. Taktiež sú v tejto časti uvedené nedostatky programov zistené v priebehu testovania, ktoré môžu spôsobiť únik citlivých informácií koncových bodov.

V piatej kapitole je vytvorené grafické rozhranie pre jeden z testovaných programov. Grafické prostredie programu by malo zjednodušiť prácu s ním a zamedziť tak chybnému procesu anonymizácie.

1 MONITOROVANIE SIETÍ A ZBER DÁT

Monitorovanie siete patrí medzi najdôležitejšie funkcie pri správe sieťovej infraštruktúry. Pomocou monitorovania môžme sledovať parametre, ktoré nám poskytnú informácie o stave siete, výkonnostných charakteristikách alebo možných bezpečnostných hrozbách. Na základe zamerania sa na sledovanie určitých charakteristík, monitorovanie siete možno rozdeliť na aktívne a pasívne.

Aktívne monitorovanie pracuje na základe vkladania testovacích paketov do siete na jednom mieste a následným prijímaním týchto paketov na mieste druhom. Ide o monitorovanie dát v reálnom čase, ktoré nám poskytuje informácie a výkonových charakteristikách siete ako sú oneskorenie paketov, stratovosť, kolísavé oneskorenie paketov (jitter) alebo priepustnosť. Tento druh monitorovania je vhodný na zlepšenie kvality služieb (QoS). Nevýhoda aktívneho monitorovania spočíva vo väčšom zaťažení sieťového hardwaru kvôli vkladaniu testovacích paketov, čo môže pri nadmernom použití viesť k slabému výkonu.

Pasívne monitorovanie spočíva v konštantnom zbere toku dát zo siete počas určitej periódy a následnom vyhodnocovaní výsledkov. Keďže pasívne monitorovanie neanalyzuje informácie zo sieťových prvkov v reálnom čase, je tento typ monitorovania zvyčajne menej náročný na hardwarové zdroje ako aktívne monitorovanie. V dôsledku toho, že dátový tok je zbieraný konštantne v čase, je pasívne monitorovanie vhodné na analýzu veľkých objemov dát a pomocou neho sme schopný zistiť využitie kapacity siete, ktoré aplikácie majú najväčší nárok na kapacitu, alebo či v sieti dochádza bezpečnostným útokom. Techniky pasívneho monitorovania však vyžadujú špecializované zariadenia na meranie prevádzky, čo môže viesť k väčšej finančnej záťaži ako pri aktívnom monitorovaní [1], [2].

Na základe vyššie spomenutých výhod a nevýhod aktívneho či pasívneho monitorovania, je pre získanie čo najväčšieho množstva informácií vhodné tieto techniky kombinovať.

1.1 Nástroje pre monitorovanie

Pre analýzu zozbieraných dátových tokov zo sieťových prvkov bolo vyvinutých viacero protokolov a nástrojov.

1.1.1 SNMP

Simple Network Management Protocol je protokol pracujúci na aplikačnej vrstve, ktorý funguje pomocou komunikácie medzi dvoma stranami na báze modelu klient/server. SNMP je jeden z najviac používaných protokolov pre správu siete.

Podporuje ho množstvo aktívnych sieťových prvkov ako sú smerovače, prepínače, prístupové body, tlačiarne, servery. Vďaka využívaniu transportného datagramového protokolu (UDP) je komunikácia veľmi rýchla ale pri využívaní UDP protokolu môže dôjsť ku strate paketov [3]. Systém SNMP sa skladá z nasledujúcih prvkov:

- **Network management system (NMS)**: Jedná sa o správcu (server), ktorý komunikuje s agentmi tak, že im posiela žiadosti a čaká na odpoveď. Následne zhromažďuje dáta, z ktorých dostáva dôležité informácie o sieťových prvkoch. Sú to informácie ako množstvo prenášaných dát, verzie používaných ovládačov, počet uzlov v sieti a podobne.
- **Agent**: Je softwarová časť zariadenia, od ktorého chceme získavať dáta. Komunikácia so serverom funguje tak, že server posiela žiadosti, agent ich prijíma a posiela odpovede v určitom intervale alebo pri určitej situácii, čo môže byť chyba (Trap).
- **Managed device**: Sú to samotné sieťové prvky, v ktorých je implementovaný SNMP agent. Prvky zbierajú dáta a upravujú ich do použiteľnej podoby pre servery (NMS). Medzi sieťové prvky patria už spomenuté prepínače, rozbočovače, prístupové servery, tlačiarne.

SNMP agenti používajú UDP port 161 a manažéri dynamický UDP port, ktorý si zvolia tak, aby boli schopní komunikovať s rôznymi agentmi. Agent následné posiela odpoveď na port, z ktorého mu prišla žiadosť. Môže však nastať situácia, kedy dôjde k poškodeniu zariadenia a agent nebude vedieť, kde má odoslať správu. Preto je pre takéto situácie zvolený port 162.

Protokol SNMP vznikol v roku 1989 a bol definovaný organizáciou IETF (Internet Engineering Task Force). Čiastočne vychádzal z jednoduchého monitorovacieho protokolu SGMP. Od začiatku 90. rokov sa stal najpoužívanejším protokolom pre správu sietí. Postupom čase sa funkcie protokolu SNMP rozširovali a sú známe tri verzie [4]:

- **SNMPv1**
  - vznik v roku 1989
  - pre získanie väčšieho množstva dát nestačí poslať len jednu žiadosť
  - slabé zabezpečenie (nešifrovaná komunikácia)
- **SNMPv2**
  - vznik v roku 1993
  - zlepšenie výkonu pre získavanie väčšieho množstva dát
  - nekompatibilný so SNMPv1 (zmena formátu zasielaných správ)
  - autentizácia textovým heslom
  - nedostatočná bezpečnosť

- **SNMPv3**
  - vznik v roku 2004
  - overovanie menom a heslom
  - možnosť šifrovať celú komunikáciu

Protokol SNMP sám nedefinuje, ktoré premenné by mal systém používať. Preto sa používa riadiaca informačná báza (MIB), kde sa popisuje štruktúra spravovaných dát. Každá hodnota v SNMP protokole je identifikovaná číselným identifikátorom (OID), ktorý presne určuje umiestnenie v hierarchickej štruktúre. Na obr. 1.1 je zobrazený formát SNMP správy. Tá obsahuje nasledujúce hodnoty:

- **Verzia** protokolu SNMP.
- **Comunity string** slúži na zabezpečenie pomocou kombinácie mena a hesla.
- **PDU typ** určuje, o aký typ správy ide(*Get*, *Set*, ...).
- **ID žiadosti** označuje príslušné dvojice žiadostí a odpovedí.
- **Error status** oznamuje úspešnosť požiadavku alebo prípadne údáva typ chyby.
- **Error ID** podrobnejšie informuje o vyskytnutej chybe a priraďuje jej určitú hodnotu.
- **OID** je identifikátor objektu.
- **Hodnota** označuje konkrétnu hodnotu premennej.

Obr. 1.1: Štruktúra SNMP správy.

SNMPv1 definuje 5 základných operácií, pomocou ktorých je možná komunikácia medzi NMS a agentmi alebo medzi NMS navzájom:

- **GetRequest** – Žiadosť o informáciu, ktorú posiela manažér agentovi o stave alebo hodnote nejakého prvku. Agent vracia odpoveď s aktuálnymi hodnotami.
- **GetNext** – Žiadosť vysielaná od manažéra smerom k agentovi o ďalšiu informáciu v MIB tabuľke, ktorá naväzuje na predchádzajúcu informáciu v MIB.
- **GetResponse** – Príkaz, pomocou ktorého posiela agent manažérovi odpoveď na príkaz GetResponse. V prípade výskytu chyby sú použité polia *error-status* a *error-index*.
- **SetRequest** – Požiadavka od manažéra k agentovi na zmenu nastavenia zariadenia alebo hodnoty v MIB tabuľke. Následne je vrátená nová nastavená hodnota.
- **Trap** – V prípade, že sa v systéme naskytne chyba alebo zvláštna situácia, je potrebné oznámiť o tom manažéra *Trap* správou. Ide o nevyžiadanú správu

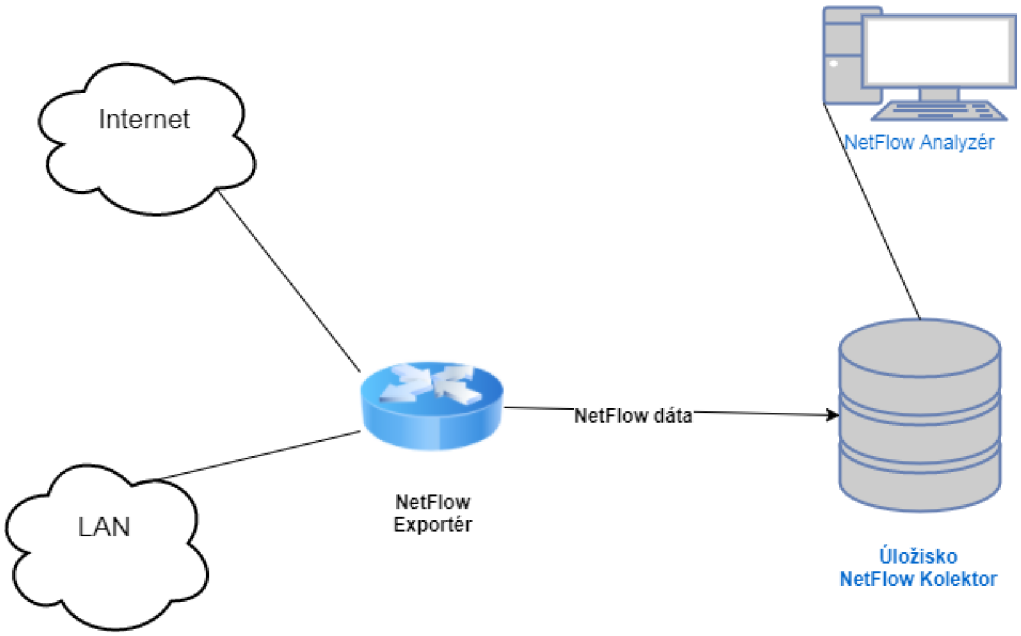
poslanú agentom manažérovi. Chybou alebo zvláštnou situáciu sa rozumie napríklad výpadok spoja, uzlu alebo zahltenie siete.

Verzie SNMPv2 a SNMPv3 sú rozšírené o ďalšie operácie.

- **GetBulk** – Používa sa v prípade, keď manažér potrebuje od agenta získať väčšie množstvo dát, ktoré pravdepodobne prekračujú maximálnu povolenú veľkosť SNMP správy. Bez použitia tejto operácie by bolo potrebné posielať viac žiadostí *GetNext* pre prenos veľkého počtu dát.
- **Inform** – Operácia, ktorá povoľuje výmenu Trap informácií medzi viacerými manažérmi.

1.1.2 NetFlow

NetFlow je technológia vyvinutá firmou Cisco Systems a jej účelom je monitorovanie sieťovej prevádzky na základe IP tokov v reálnom čase. Architektúra sa skladá z niekoľko NetFlow exportérov a jedného NetFlow kolektoru ako popisuje obr. 1.2. NetFlow exportér je pripojený na monitorovanú linku, typicky smerovač, kde analyzuje prechádzajúce pakety a na základe zachytených IP tokov generuje NetFlow štatistiky, ktoré odosiela na NetFlow kolektor. Štatistiky sú odosielané prostredníctvom protokolu UDP alebo pomocou transportného protokolu pre riadenie toku (SCTP). Kolektor zbiera získané štatistiky z viacerých exportérov a pomocou aplikácií je následne možné zobraziť a analyzovať získanú prevádzku pomocou grafov a tabuliek.



Obr. 1.2: Schéma architektúry technológie Netflow

Technológia NetFlow je charakteristická tým, že pracuje s takzvaným IP tokom. Každý paket, ktorý je smerovaný v sieti, obsahuje 7 atribútov. Atribúty používané protokolom NetFlow:

- IP zdrojová adresa,
- IP cieľová adresa,
- zdrojový port,
- cieľový port,
- typ protokolu vrstvy 3,
- trieda služby,
- rozhranie smerovača alebo prepínača.

Pakety so spoločnými vyššie spomenutými atribútmi sú zoskupené do toku. Pre každý tok sú zaznamenávané komunikujúce strany, doba vzniku IP toku, dĺžka trvania, počet prenesených bajtov a niektoré ďalšie informácie [5].

Pri tejto architektúre sa objavujú nevýhody v podobe záťaže smerovacieho výkonu na smerovačoch kvôli výpočtom NetFlow štatistík. Preto väčšina smerovačov používa na vstupe vzorkovanie, čo znamená, že pre výpočet sa používa len každý n-tý paket. To má za následok určité zníženie presnosti merania a bezpečnosti.

Pre odstránenie nevýhod v NetFlow architektúre sa začali používať pasívne NetFlow sondy. Sú určené na monitorovanie a export NetFlow štatistík. Vďaka svojej jednoduchosti sú veľmi lacné a je možné ich pripojiť do ľubovoľného bodu v sieťovej infraštruktúre, čo znamená, že nie je zaťažovaný smerovací výkon na smerovačoch. Naviac, získané štatistiky sú exportované na kolektor samostatnou linkou. Preto použitie pasívnej sondy zvyšuje bezpečnosť siete proti prípadným útočníkom.

Firma Cisco Systems počas rokov predstavila viacero verzií NetFlow, ktoré sa od seba významne líšia [6]:

- **verzia 1**
  Prvá implementácia obmedzená len na IP adresy verzie 4 (IPv4) (bez IP masky a čísiel autonómnych systémov).
- **verzie 2–4**
  Interné verzie Cisca, ktoré neboli nikdy uvoľnené.
- **verzia 5**
  Najrozšírenejšia verzia podporovaná aktívnymi sieťovými prvkami. Nepodporuje IP adresy verzie 6 (IPv6), fyzické adresy adresy (MAC), čísla virtuálnych lokálnych sietí (VLAN).
- **verzie 6–8**
  Tieto verzie NetFlow sú takmer nepoužívané.
- **verzia 9**
  Podporuje zložky, ktoré verzia 5 nepodporovala. Umožňuje flexibilne nastaviť, aké informácie budú v sieťovej prevádzke sledované.

- **verzia 10**
  Známa tiež ako IPFIX. Umožňuje rozšíriť dátové toky o ďalšie informácie o sieťovej prevádzke.

IPFIX

IP Flow Information Export je protokol definovaný organizáciou IETF. Bol vytvorený na exportovanie IP tokov zo sieťových prvkov do kolektoru aby boli následne spracované na následnú analýzu. Pretože bol odvodený od technológie NetFlow verzie 9 funguje na rovnakom princípe. Hlavný rozdiel medzi NetFlow a IPFIX spočíva v tom, že IPFIX povoľuje polia premenlivej dĺžky. V premenlivých poliach je možné uložiť informácie ako sú internetové adresy (URL), správy alebo hostiteľské stanice HTTP a ďalšie. Táto funkcionalita umožňuje dodávateľom hardwaru vložiť akékoľvek informácie do toku a vyexportovať ho z kolektora, respektíve analyzátora na analýzu [7].

1.1.3 Wireshark

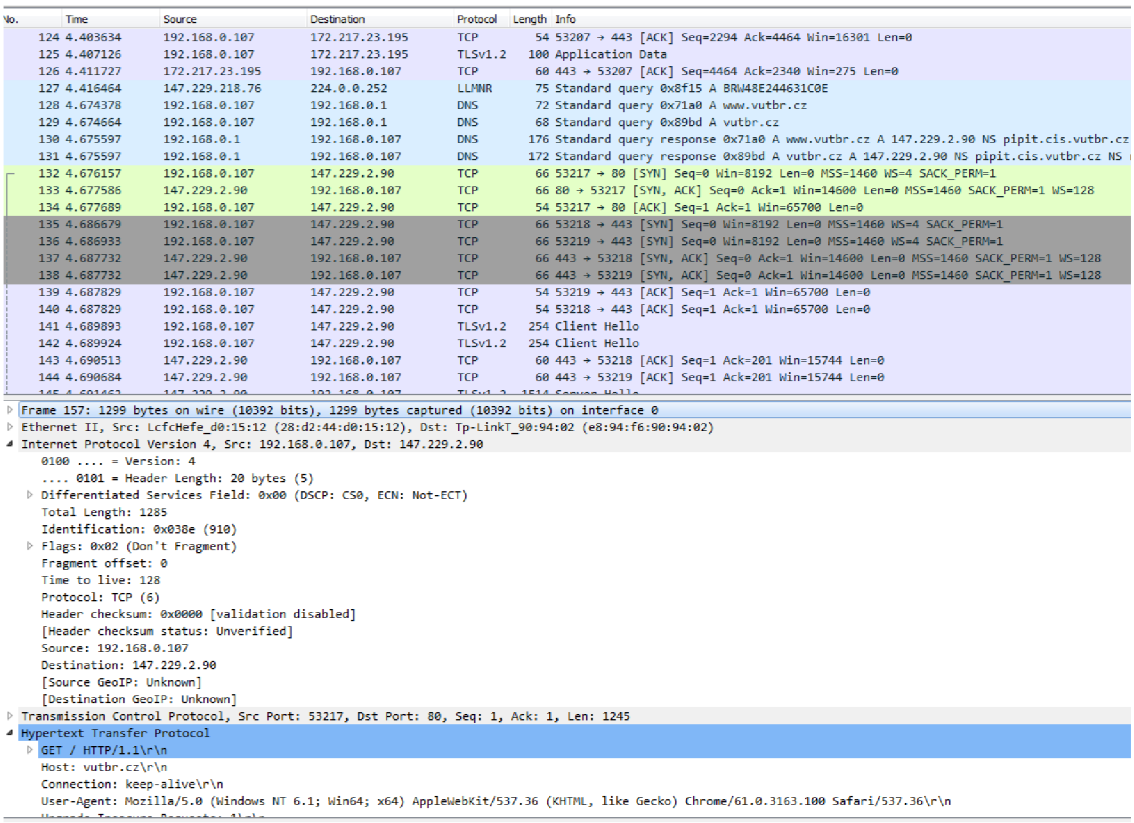
Wireshark je voľne dostupný software s prístupným zdrojovým kódom (open-source), slúžiaci na analýzu sieťovej komunikácie, prípadne odhalenia vzniknutých chýb. Dokáže zachytiť a analyzovať veľký počet sieťových protokolov a následne zachytenú komunikáciu zobraziť v grafickom rozhraní. Analýza sieťovej prevádzky programom Wireshark je zobrazená na obr. 1.3. Nástroj dokáže pracovať taktiež v príkazovom riadku pomocou utility TShark. Vznikol v roku 1998 pod názvom Ethereal a v roku 2006 bol premenovaný na dnešný názov. Je použiteľný pre Unix operačné systémy ako sú Linux, Mac OS alebo Microsoft Windows.

Vlastnosti

Vďaka tomu, že program Wireshark umožňuje nastaviť sieťové rozhrania do promiskuitného módu, je možné zobraziť celú prevádzku na týchto rozhraniach. Pre zachytávanie paketov využíva táto aplikácia knižnicu *pcap*. Výhody softwaru Wireshark sú:

- sieťová prevádzka môže byť zachytávaná z aktuálne bežiacej prevádzky alebo môže byť načítaná zo súboru, na ktorom je už komunikácia predtým zaznamenaná (offline analýza),
- software je schopný analyzovať dáta z rozdielnych typov sietí ako je Ethernet, IEEE 802.11 (Wi-fi), komunikáciu medzi dvoma uzlami (PPP) a virtuálne rozhrania (loopback),

- zachytená prevádzka môže byť analyzovaná pomocou grafického rozhrania alebo príkazového terminálu,
- pre zobrazenie požadovaných dát je možné upraviť zachytenú komunikáciu pomocou filtrov,
- Wireshark používa v grafickom rozhraní farby, aby zjednodušil užívateľom identifikovať rôzne typy prevádzky [8].



Obr. 1.3: Zachytená sieťová komunikácia programom Wireshark

1.1.4 tcpdump

Tcpdump je populárny paketový analyzátor pracujúci pomocou príkazového riadku. Analyzuje komunikáciu v sieti hlavne protokoly zo sady TCP/IP. Vznikol v roku 1988 a bol zostrojený ľuďmi pracujúcimi v tom čase v Lawrence Berkeley Laboratory. Od vtedy bolo vydaných viacero verzií. Najnovšia verzia, *tcpdump 4.9.2*, bola vydaná v septembri roku 2017.

Program pracuje na väčšine UNIX systémoch (Linux, Solaris, Mac OS,...). Pre verziu operačného systému Microsoft Windows bol vytvorený program WinDump. Na zachytávanie paketov používajú UNIX systémy knižnicu *libpcap* a operačný systém Windows používa knižnicu *WinPcap*.

Tcpdump má podobne ako program Wireshark viacero užitočných vlastností. Sú dostupné filtre založené na BPF (Barkeley Packet Filter), slúžiace na zredukovanie počtu viditeľných paketov, čo je výhodné pre analýzu konkrétnych paketov v sieťovej prevádzke. Program ďalej umožňuje zachytené pakety uložiť, aby boli použiteľné pre neskoršiu analýzu. Z toho vyplýva, že programom je možné taktiež analyzovať v minulosti zachytenú komunikáciu. V niektorých prípadoch však musí mať užívateľ špeciálne privilégia od správcu pre zachytávanie paketov zo sieťového rozhrania. Pri analýze už uloženej sieťovej prevádzky nie sú potrebné špeciálne privilégia [9].

1.2 Ochrana osobných údajov

Bezpečnosť koncových užívateľov v sieti prešla v posledných rokoch veľa zmenami. Išlo hlavne o osobné údaje užívateľov a taktiež ich aktivitu v sieti. V roku 2006 bola v Európskej Únii schválená smernica 2006/24/ES, takzvané „data retention“. Tento pojem označuje ukladanie prevádzkových a lokalizačných dát u poskytovateľov telekomunikačných služieb a taktiež poskytovateľov internetových služieb.

V Českej republike bol schválený zákon „č. 127/2005 Sb., o elektronických komunikacích v § 97, odst. 3.“, ktorý nariaďoval uchovávanie získaných dát. Hlavný dôvod pre zavedenie tohto zákonu bolo odhalenie možných teroristických útokov a celkovo zníženie počtu závažných trestných činov. Prijatím vyššie spomenutého zákona vznikla povinnosť uchovávania údajov ako sú:

- typ pripojenia,
- telefónne číslo alebo označenie užívateľa,
- identifikátor užívateľského účtu,
- MAC adresa,
- dátum a čas zahájenia a ukončenia pripojenia,
- označenie prístupového bodu u bezdrátového pripojenia,
- IP adresa a číslo portu,
- meno, priezvisko a adresa zákazníka.

Získané informácie museli byť uchovávané po dobu od šiestich mesiacov až po dva roky. Na rozdiel od telefónneho odpočúvania, nebol uložený obsah komunikácie.

Od počiatku prijatia zákonov a smerníc týkajúcich sa plošného uchovávania dát, boli zákony napádané mnohými občanmi a občianskymi organizáciami. Zákony sa týkali všetkých osôb využívajúcich elektronické služby bez ohľadu na to, či boli podozrivé z páchania trestnej činnosti. Taktiež boli uchovávané údaje, ktoré sa mohli týkať profesného tajomstva (poskytovatelia zdravotných, sociálnych služieb atď.). Zákon taktiež neupravoval, aký okruh osôb má prístup k získaným údajom. Preto bol v roku 2012 zákon o data retention upravený. Boli vymedzené orgány, ktoré sú

oprávnené o dáta žiadať. Doba, počas ktorej mali byť dáta uchovávané sa pevne stanovila na 6 mesiacov. Taktiež vznikla povinnosť, až na výnimky, informovať subjekt údajov o tom, že boli jeho údaje využité [10].

Na základe dôkladnej analýzy prijatej smernice z roku 2006, dospel Súdny dvor EU k záveru že plošný zber dát zasahuje do ľudských práv ako je napríklad právo na súkromie a túto smernicu označil za neplatnú. V roku 2016 rozhodol Európsky súdny dvor o tom, že plošný zber prevádzkových a lokalizačných dát je ilegálny. V Českej republike však zákon prijatý v roku 2005 a upravený v roku 2012, ostáva v platnosti [11].

1.2.1 GDPR

Obecné nariadenie na ochranu osobných údajov (General Data Protection Regulation) je nová legislatíva prijatá Európskou úniou, ktorá zvýši ochranu osobných údajov občanov. Schválená bola 27. 04. 2016 a do platnosti vstúpi 25. 05. 2018. GDPR sa bude dotýkať všetkých, ktorí zhromažďujú alebo spracúvajú osobné údaje užívateľov. Týka sa teda aj poskytovateľov internetových služieb (ISP). Dôjde taktiež k rozšíreniu osobných údajov o technické parametre ako sú e-mail, IP adresa alebo uložené informácie (cookies) v zariadení užívateľa [12].

Správcovia a spracovatelia údajov budú musieť na základe prijatej legislatívy vykonať viaceré opatrenia týkajúce sa najmä týchto oblastí [13]:

- implementácia ochrany dát,
- menovanie poverenca pre ochranu osobných údajov tzv. DPO (Data Protection Officer),
- zavedenie pseudonymizácie osobných údajov (spracovanie osobných údajov tak, aby nemohlo dôjsť k spätnému priradeniu konkrétnej osobe),
- vedenie záznamov o činnostiach spracovania,
- konzultácie s dozorným orgánom pred samotným spracovaním osobných údajov.

2 ANONYMIZÁCIA

Ako bolo spomenuté v predchádzajúcej kapitole, pod osobným údajom sa rozumie taká informácia, ktorá vedie k identifikácii konkrétneho človeka. Anonymizácia je proces, pri ktorom dochádza k upravovaniu sieťových prevádzkových dát v snahe ochrániť identitu koncových bodov pri vyhodnocovaní štatistík získaných zo sieťovej komunikácie. Čelí sa preto konfliktu medzi súkromím koncových užívateľov a výpovednou hodnotu zozbieraných dát. Cieľom anonymizácie je zabrániť identifikácii komunikujúcich koncových bodov a zároveň zachovať čo najväčšiu použiteľnosť získaných dát [14].

Dáta sieťovej prevádzky obsahujú sekvenciu paketov prúdiacu od zdroja do cieľa koncových bodov. Túto sekvenciu nazývame tok dát. Obsahuje viacero polí. Definovaný je piatimi hlavnými poliami: zdrojová IP adresa, cieľová IP adresa, zdrojové číslo portu, koncové číslo portu, typ použitého protokolu. V závislosti na type toku môže obsahovať ďalšie polia akými sú: dĺžka paketu, MAC adresa, číslo toku, číslo autonómneho systému (AS), veľkosť okna a maximálna veľkosť segmentu. Niektoré z týchto polí však dokážu odhaliť identitu koncových bodov. Preto je dostupných viacero anonymizačných algoritmov a anonymizačných nástrojov. Dokážu anonymizovať IP adresy, MAC adresy, čísla portov, dĺžky paketov a iné [15].

2.1 Anonymizačné techniky

Anonymizačné techniky poskytujú rôzne úrovne anonymizácie zachytených dát. Medzi anonymizačné techniky sa radia:

- **Black marker** algoritmus je metóda, pri ktorej sú zmazané alebo nahradené všetky informácie v poli sieťového toku. Hoci je dosiahnutá anonymizácia koncových bodov, následná využiteľnosť dát je veľmi nízka.
- **Enumeračný** algoritmus pracuje na princípe zoraďovania dát. Je vhodný napríklad pre anonymizovanie dĺžky paketov. Po roztriedení paketov podľa hodnoty je hodnota dĺžky najmenšieho paketu priradená paketu s najväčšou dĺžkou a naopak. Pri tejto anonymizácií je zničená informácia časového razítka.
- **Hašovací** algoritmus nahrádza dáta pevným bitovým reťazcom. Každá zmena dát pozmení „hash“ hodnotu. Výsledok po použití tejto techniky je niekedy kratší ako býva hodnota poľa a preto je algoritmus ľahko prelomiteľný.
- **Degradácia presnosti** je algoritmus, ktorý odstraňuje najviac presné časti poľa. Preto sa používa pri anonymizácii poľa časového razítka. Tento proces môže zlúčiť viacero časových razítok do jednej. Údaje vzniknuté po anonymizácii nemusia byť užitočné pre aplikácie, ktoré využívajú presné sekvenčné toky.

- **Permutačný** algoritmus je zvyčajne používaný pre IP adresy a MAC adresy. Aplikuje náhodnú obmenu adresy pomocou množiny možných adries. Používa dve hašovacie tabuľky. Jedna obsahuje informácie o mapovaní neanonymizovaných IP adries do anonymizovaných IP adries a druhá obsahuje všetky uložené anonymizované adresy.Permutačná funkcia je náhodná, preto je mapovanie vždy rozdielne.
- **Anonymizácia zachovávajúca hodnotu prefixu** je algoritmus podobný permutačnému algoritmu. Funguje však na presnom substitučnom systéme. Ak dve IP adresy zdieľajú rovnakú hodnotu prvých n bitov tak ich anonymizované IP adresy budú tiež zdieľať rovnakú hodnotu prvých n bitov. Tento algoritmus udržuje predchádzajúcu štruktúru IP adries na základe udržovania hodnôt prefixu. Kryptografické kľúče, ktoré sú používané, udržujú mapovanie konzistentné.
- **Náhodný časový posun** pridáva náhodný posun každej hodnoty poľa v súbore dát. Vytvorí možnú množinu hodnôt a následne vyberie z vytvorenej množiny hodnotu k posunu.
- **Skracovanie** sa používa k anonymizovaniu IP a MAC adries. Odstraňuje n posledných významných bitov z poľa a nahradí ich nulami. To zaručuje zabráneniu identifikácie koncových bodov.
- **Obrátené skracovanie** je používané na anonymizáciu IP a MAC adries, ale na rozdiel od klasického skracovania, odstraňuje n najviac významných bitov v poli. Vďaka tomu sú neidentifikovateľné adresy siete alebo organizácie.
- **Zničenie časovej jednotky** je algoritmus, ktorý sa používa na anonymizáciu časového razítka, kedy je časová jednotka zničená a nahradená nulami.

Pre jednotlivé polia je možné využiť viacero typov anonymizačných techník ako je zobrazené v tab. 2.1 [15].

2.2 Anonymizačné nástroje

IP adresa sa stane v roku 2018 osobným údajom pre všetky krajiny Európskej únie, kedy vstúpi do platnosti Obecné nariadenie na ochranu osobných údajov (GDPR). Pomocou nej je možné identifikovať koncového užívateľa v sieti a následne zistiť jeho aktivitu. Preto je anonymizácia IP adresy jedným z najdôležitejších krokov k anonymizácií užívateľa v sieti.

Preto v tejto časti budú vyberané nástroje, ktoré používajú techniky respektíve algoritmy na anonymizáciu poľa s IP adresami. Taktiež budú roztriedené podľa toho, s akým formátom dát sú schopné pracovať.

Tab. 2.1: Anonymizačné techniky v závislosti na type poľa

Pole	**Anonymizačné techniky**
IP adresa	Skracovanie, Obrátené skracovanie, Permutácia, Zachovanie prefixu, Black marker
MAC adresa	Skracovanie, Obrátené skracovanie, Permutácia, Zachovanie prefixu, Black marker
Časové razítko	Degradácia presnosti, Enumerácia, Náhodný posun, Black Marker
Počítadlo	Degradácia presnosti, Black Marker
Číslo portu	Permutácia, Black marker

2.2.1 Nástroje pracujúce s formátom pcap

Anonym tool

Anonym tool je anonymizačný nástroj založený na princípe programu MATLAB, ktorý dokáže pracovať na operačnom systéme Windows aj Linux. Je navrhnutý s grafickým užívateľským rozhraním (GUI) a taktiež umožňuje zobrazenie rôznych druhov analýz ako sú napríklad: veľkosť paketov, dĺžka paketov, priepustnosť. Možnosti nástroja Anonym sú uvedené v tab. 2.2. Medzi veľké výhody nástroja Ano-

Tab. 2.2: Možnosti nástroja Anonym

Typ vstupných dát	pcap, mrt
Typ výstupných dát	pcap, mrt
Anonymizované polia	IP adresa (v4, v6), MAC adresa, Čísla portov, Dĺžka paketov, Časové razítka, Počítadlá
Anonymizačné techniky	Black marker, Anonymizácia so zachovaním prefixu, Skracovanie, Obrátené skracovanie, Časová degradácia presnosti, Časové posunutie

nym patrí schopnosť anonymizovať IPv6 adresy, čo nástroje vyvinuté pred ním neboli schopné. Ďalšia výhoda spočíva v tom, že Anonym, ponúka možnosť použiť

Kolmogorov - Smirnov test. Test slúži na porovnanie množín anonymizovaných údajov s viacerými referenčnými rozdeleniami na odvodenie základnej štruktúry sieťovej prevádzky. To umožňuje extrahovať premenné, odhaľovať problémy alebo anomálie, testovať základné predpoklady a vyvíjať teoretické modely sieťovej prevádzky [15].

TraceWrangler

TraceWrangler je nástroj na zachytávanie sieťovej komunikácie ale jeho hlavné využitie slúži na anonymizáciu dát., Tento nástroj podporuje operačný systém Windows. Pracuje s formátom PCAP ale taktiež dnes novšie známym PCAPng súborovým formátom, ktorý je štandardným formátom používaným programom Wireshark. Podobne, ako vyššie spomenutý nástroj Anonym, podporuje GUI, čo uľahčuje prácu pri odstraňovaní alebo nahradzovaní citlivých dát [16].

Je schopný anonymizovať polia ako sú čísla Ethernetových rozhraní, IPv4 a IPv6 adresy, MAC adresy. Taktiež dokáže pracovať s ďalšími protokolmi, v ktorých dokáže upraviť rôzne informácie. Sú to protokoly na zisťovanie fyzických adries (ARP), TCP protokol, UDP protokol a protokol slúžiaci na odosielanie chybových správ (ICPMv4) [17].

SCRUB-tcpdump

SCRUB-tcpdump je nástroj rozširujúci nástroj tcpdump, ktorý sa využíva na jednoduchú správu dát a súčasne na ochranu citlivých informácií, aby neboli použité v prípadnej analýze. Používateľ môže anonymizovať polia na viacero požadovaných úrovní výberom techník, ktoré môžu odstrániť všetky informácie, pridaním šumu alebo zmenou dát. Tieto viacúrovňové anonymizačné techniky môžu byť uplatnené v rozdielnych poliach simultánne, s rôznymi účinkami na štatistické vlastnosti celej stopy paketu. Keďže organizácie majú bezpečnostné pravidlá s rôznymi požiadavkami na ochranu údajov, viacúrovňové anonymizačné možnosti poskytujú flexibilitu pri výbere anonymizačných schém. Nie je jednotne daná anonymizačná schéma, ktorá by sa používala, preto je pre anonymizačný nástroj dôležité mať viacúrovňové možnosti anonymizácie. Možnosti nástroja SCRUB-tcpdump sú uvedené v tab. 2.3 [18].

Pcap obfuscator

Pcap obfuscator je ďalší z programov slúžiacich na anonymizáciu PCAP súborov naprogamovaný v jazyku Python Je navrhnutý s grafickým rozhraním. Je schopný anonymizovať IPv4 adresy, MAC adresy a čísla VLAN [19].

Tab. 2.3: Možnosti nástroja SCRUB-tcpdump

Typ vstupných dát	tcpdump/pcap, sieťové rozhranie
Typ výstupných dát	tcpdump/pcap formát
Anonymizované polia	IPv4 adresa, TCP/UDP porty, Dĺžka paketov, Časové razítka, Payload, Fragmentation flag
Anonymizačné techniky	Black Marker, Náhodná permutácia so zachovaním prefixu, Náhodná permutácia, Časové posunutie

Capsan

Capsan je nástroj bežiaci v príkazovom riadku, ktorý dokáže modifikovať viaceré polia v paketoch a na to využíva anonymizačné techniky, ktoré sú uvedené v tab.2.4.

Pokiaľ sa v sieťovej prevádzke nachádzajú protokoly, ako napríklad ARP, ktoré neobsahú IP paket, Capsan ich nespracuje a vo výslednom anonymizovanom súbore sa nenachádzajú. Capsan taktiež nespracúva IP fragmenty alebo pakety, ktoré neobsahujú UDP alebo TCP protokoly [20].

Tab. 2.4: Možnosti anonymizácie nástrojom Capsan

Typ vstupných dát	formát pcap
Typ výstupných dát	formát pcap
Anonymizované polia	IPv4 adresy, MAC adresy, čísla portov(TCP, UDP)
Anonymizačné techniky	Náhodná permutácia, Black Marker, Anonymizácia IP adries so zachovaním prefixu

2.2.2 Nástroje pracujúce s formátom NetFLow

NFDUMP

NfDump je nástroj, ktorý zbiera a spracúva NetFlow dáta pomocou príkazového riadku. Podporuje NetFlow (v5, v7, v9). Cieľom tohto nástroja je analyzovať získané dáta z minulosti, ktoré sú obmedzené len kapacitou disku, na ktoré sa ukladajú.

NfDump poskytuje sadu nástrojov, ktoré sa využívajú na správu dát:

- **nfcapd** – Číta NetFlow dáta zo siete a ukladá ich do súborov.
- **nfdump** – Číta dáta zo súborov uložené nástrojom nfcapd.
- **nfprofile** – Číta dáta zo súborov uložené nástrojom nfcapd. Na základe nastavených filtrov (profilov) filtruje dáta a ukladá ich do súborov.
- **nfreplay** – Číta dáta zo súborov uložené nástrojom nfcapd a posiela ich cez sieť ďalšiemu klientovi.

Uvedené nástroje sú optimalizované pre rýchle a efektívne filtrovanie. Používaná syntax vyzerá veľmi podobne ako syntax nástroja tcpdump.

NfDump je schopný anonymizovať IP adresy použitím anonymizačnej techniky so zachovaním prefixu. IP adresy sú anonymizované predtým ako sú uložené do súboru [21].

2.2.3 Nástroje pracujúce s formátom pcap aj NetFlow

Anontool

Anontool je nástroj, ktorý dokáže anonymizovať prevádzku zachytenú zo siete a taktiež prevádzku uloženú. Pracuje na báze anonymizácie pomocou rozhrania pre programovanie aplikácií (AAPI). To dovoľuje užívateľom definovať vlastné anonymizačné aplikácie. AAPI poskytuje širokú škálu anonymizačných techník, ktoré môže užívateľ aplikovať na viacero polí ako je uvedené v tab. 2.5. Na zachytávanie a zapisovanie na disk používa knižnicu *libpcap*. Podporuje taktiež formáty NetFlow (v5, v9) [22].

Tab. 2.5: Možnosti nástroja Anontool

Typ vstupných dát	NetFlow (v5, v9) vo formáte tcpdump, sieťové rozhrania
Typ výstupných dát	tcpdump/pcap formát
Anonymizované polia	IP adresy, väčšina NetFlow polí, NetFlow kontrólne súčty
Anonymizačné tecniky	Black Marker, Náhodná permutácia, Anonymizácia so zachovaním prefixu, Hašovanie

Flaim

Flaim je ďalší z nástrojov používaných na anonymizáciu dát. Používajú ho UNIX operačné systémy (Linux, Mac OS X, ...). Flaim od seba striktne oddeľuje analýzu dát a anonymizáciu. To zabezpečuje jeho architektúra, ktorá sa skladá z dvoch hlavných komponentov. Flaim jadro a Flaim moduly.

Jadro obsahuje anonymizačné algoritmy a politiku anonymizovania jednotlivých polí. Napríklad nebude povolené použiť anonymizáciu IP adresy so zachovaním prefixu na pole s časovým razítkom.

Flaim moduly pozostávajú z knižníc metód, slúžiacich na analýzu rôznych typov záznamov. Samostatný modul sa používa na analýzu len jedného typu záznamu. Modul cez rozhranie komunikuje s jadrom a stará sa o to, aké formáty dát budú na vstupe a výstupe [23].

Anonymizér Flaim podporuje viaceré formáty záznamov a tiež anonymizačných techník. Možnosti, ktoré podporuje Flaim sú uvedené v tab. 2.6.

Tab. 2.6: Možnosti anonymizácie nástrojom Flaim

Typ vstupných dát	NetFlow (v5, v9), tcpdump/pcap, IP tabuľky, NFDUMP
Typ výstupných dát	záleží na type použitého modulu (napr. tcpdump/pcap)
Anonymizované polia	IP adresy (v4, v6), MAC adresy, ďalšie polia (TCP, UDP, ICMP, Ethernet)
Anonymizačné tecniky	Skracovanie, Náhodná permutácia, Black Marker, Časové posunutie, Enumeračný algoritmus, Anonymizácia so zachovaním prefixu, Hašovanie

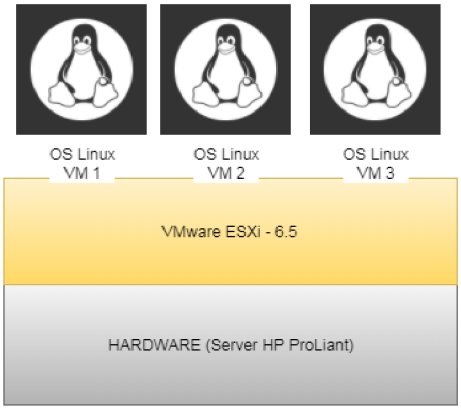
3 PRÍPRAVA PROSTREDIA PRE SIMULÁCIU SIEŤOVEJ PREVÁDZKY

Táto časť sa bude venovať vytvoreniu virtuálnej siete, ktorá by mala simulovať chovanie reálnej siete. V tejto sieti bude prebiehať komunikácia medzi serverom a klientmi a následne sa bude zachytávať pre neskoršiu anonymizáciu jednotlivých častí dát. Pri anonymizácií budú použité nástroje, ktoré pracujú s dátami formátu *pcap* a taktiež s NetFlow dátami. Preto naša laboratórna sieť bude upravená tak, aby dokázala generovať NetFlow dáta a posielať ich do takzvaného kolektora.

3.1 Vytvorenie virtuálnej siete

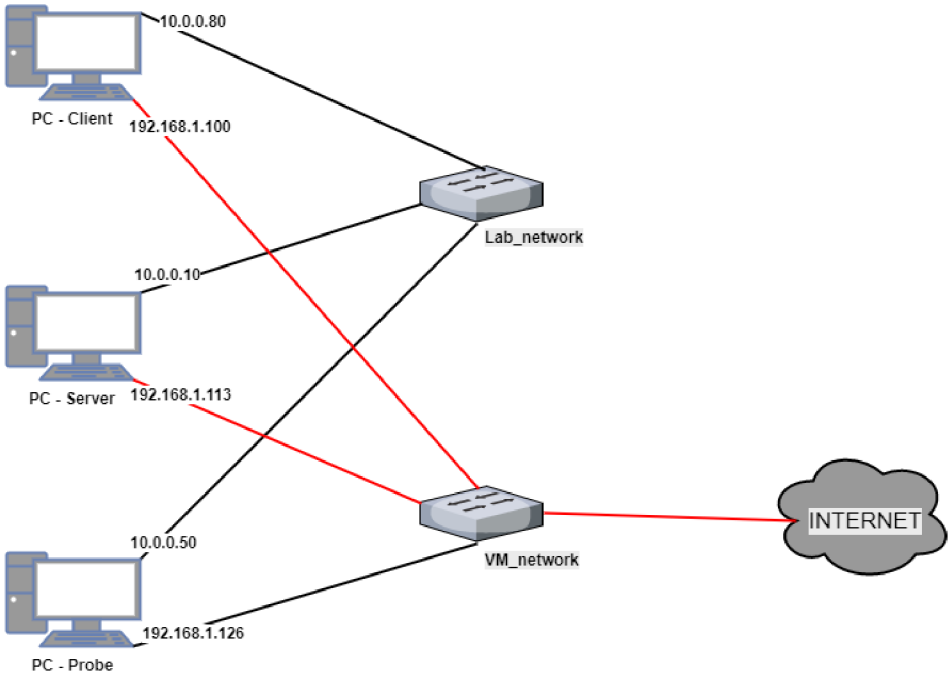
Zostavenie virtuálnej siete je možné uskutočniť pomocou viacerých nástrojov. V našom prípade bol použitý hypervízor VMware HPE - ESXi - 6.5.0, ktorý sa používa na virtualiáciu hardwaru, aby bolo možné na danom zariadení spustiť a pracovať s viacerými operačnými systémami. Princíp architektúry virtualizácie je zobrazený na obr. 3.1.

Ako fyzický hardware bol použitý jeden zo školských serverov HP ProLiant DL80 Gen9. Tento server obsahuje 6 procesorov typu Intel (R) Xeon (R) CPU E5-2603 v4 @ 1,70GHZ a fyzickú pamäť (RAM) s veľkosťou 16GB. Do servera bol priamo nainštalovaný VMware ESXi. Následne bolo možné pripojiť sa na vytvorený ESXi server cez webový prehliadač počítača umiestneného v lokálnej sieti, pomocou IP adresy, ktorá bola nastavená pri inštalácii VMware ESXi na náš server. Po pripojení sa zobrazí grafické rozhranie, kde je možné inštalovať virtuálne stroje, spájať ich do virtuálnych sietí a taktiež kontrolovať stav vytvorenej siete a použitých prvkov pomocou viacerých dostupných ukazovateľov.



Obr. 3.1: Architektúra virtualizácie hardwaru

Pre vytvorenie virtuálnej siete boli vytvorené tri virtuálne stroje s operačným systémom Linux Ubuntu(64 - bit) - 16.04.3 bez grafického rozhrania. Na každom virtuálnom stroji boli nakonfigurované dve sieťové rozhrania. V rozhraní s názvom „ens192“ bola strojom priradená IP adresa 192.168.1.X. Pomocou tohto rozhrania bolo možné vzdialene sa pripojiť (SSH) k jednotlivým PC a taktiež komunikovať v reálnej sieti. Ďalšie rozhranie, „ens160“ ,slúžilo na vzájomné prepojenie všetkých troch PC pomocou virtuálneho prepínača taktiež na naše laboratórne účely. V tomto rozhraní používali PC IP adresy 10.0.0.X. Schéma vytvorenej virtuálnej siete je zobrazená na obr. 3.2.



Obr. 3.2: Schéma virtuálnej siete

3.2 Generovanie a zachytávanie sieťovej komunikácie

Po zostavení laboratórnej siete bolo potrebné nastaviť jednotlivé počítače tak, aby spolu simulovali chovanie reálnej siete. Na počítači PC-Server bol nainštalovaný webový server Apache 2, na ktorý boli neskôr posielané žiadosti o spojenie. Ďalej bol na PC-Server nainštalovaný nástroj na zachytávanie a analýzu sieťovej komunikácie.

Na počítači s názvom PC-Client boli potrebné nástroje, ktoré by dokázali posielať žiadosti a komunikovať s webovým serverom nainštalovaným na PC-Server.

Prvým nástrojom bol ***nping***, ktorý slúži ako paketový generátor. Týmto nástrojom je možné generovať pakety rôznych protokolov a upravovať polia hlavičiek. Na generovanie paketov bol použitý následujúci príkaz:

- `sudo nping --tcp-connect -p 80 10.0.0.10 -c 5`.

Týmto príkazom sa vygenerovalo päť TCP spojení medzi PC-Client a PC-Server, kde je nainštalovaný webový server s číslom portu 80.

Avšak nástroj *nping* dokáže generovať len pakety s rovnakou zdrojovou IP adresou, čo bola v našom prípade IP adresa 10.0.0.80. Pretože neskoršia anonymizácia bude cielená hlavne na anonymizáciu IP adries, bol na generovanie paketov použitý nástroj ***hping3***, ktorý dokáže vo vybraných TCP/IP paketoch náhodne meniť IP adresy, čo bolo v našom prípade potrebné. Nástroj *hping3* dokáže poslať veľké množstvo žiadostí v krátkom čase na požadovanú adresu, čo môže byť využité k odopreniu služby (DoS) danou službou. Na generovanie paketov boli použité nasledujúce príkazy:

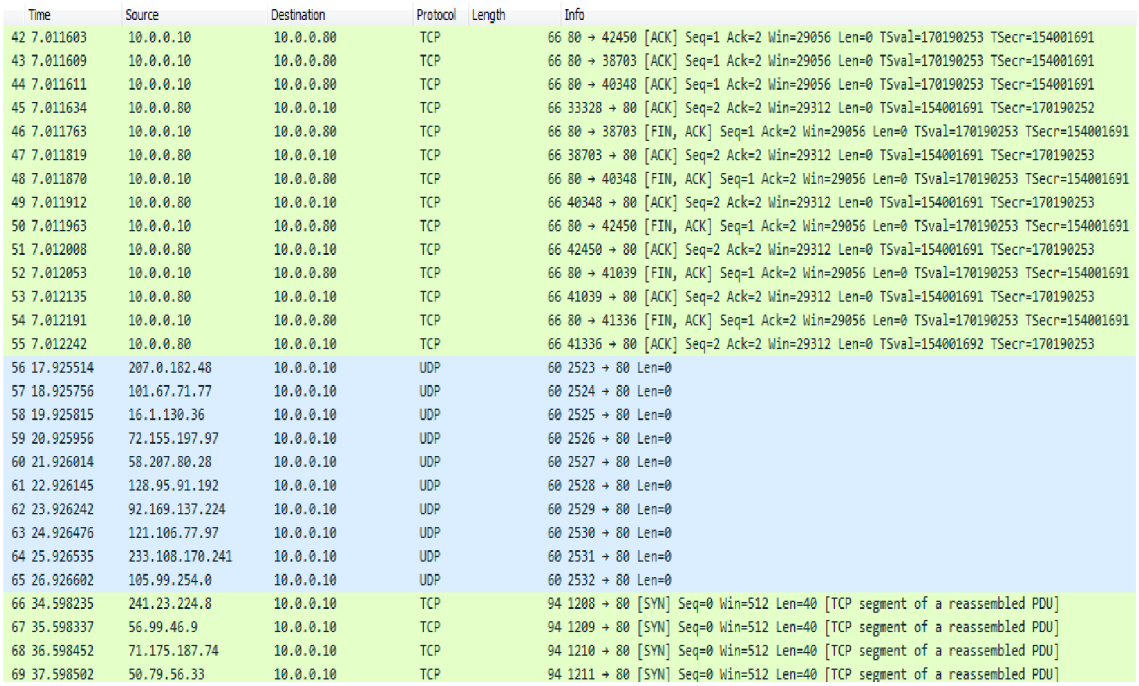
- `sudo hping3 -c 20 -i 1 -I ens160 --rand-source -p 80 10.0.0.10 --data 40 -z -V --syn`,
- `sudo hping3 --udp -c 20 -i 1 -I ens160 -V --rand-source -p 80 10.0.0.10`.

Prvým príkazom bolo vygenerovaných 20 paketov s hlavičkou protokolu TCP, náhodnými zdrojovými IP adresami a typom žiadosti synchronizácia (SYN) uvedenej v TCP hlavičke. Veľkosť dát bola nastavená na 40 bajtov. Druhým príkazom bolo vygenerovaných 20 paketov s hlavičkou protokolu UDP. Pri použití nástroja *hping3* cieľová služba, v našom prípade web server, nenadväzuje s klientmi spojenie ale len prijíma žiadosti.

Aby bolo možné sieťovú komunikáciu medzi klientmi a serverom zachytiť a následne zobraziť, bolo potrebné na PC-Server nainštalovať nástroj ***tcpdump***. Tento nástroj pracuje len v príkazovom riadku a pre prehľadnejšie zobrazenie sme *tcpdump* použili len na zachytenie paketov a uloženie zachytenej prevádzky do súboru formátu *pcap*. Na to bol použitý nasledujúci príkaz:

- `-sudo tcpdump -w network_traffic.pcap`.

Následne mohol byť súbor so zachytenou komunikáciou zobrazený v grafickom režime programom Wireshark. Zachytená sieťová prevádzka je zobrazená na obr. 3.3.



Time	Source	Destination	Protocol	Length	Info
42 7.011603	10.0.0.10	10.0.0.80	TCP		66 80 → 42450 [ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
43 7.011609	10.0.0.10	10.0.0.80	TCP		66 80 → 38703 [ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
44 7.011611	10.0.0.10	10.0.0.80	TCP		66 80 → 40348 [ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
45 7.011634	10.0.0.80	10.0.0.10	TCP		66 33328 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001691 TSecr=170190252
46 7.011763	10.0.0.10	10.0.0.80	TCP		66 80 → 38703 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
47 7.011819	10.0.0.80	10.0.0.10	TCP		66 38703 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001691 TSecr=170190253
48 7.011870	10.0.0.10	10.0.0.80	TCP		66 80 → 40348 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
49 7.011912	10.0.0.80	10.0.0.10	TCP		66 40348 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001691 TSecr=170190253
50 7.011963	10.0.0.10	10.0.0.80	TCP		66 80 → 42450 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
51 7.012008	10.0.0.80	10.0.0.10	TCP		66 42450 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001691 TSecr=170190253
52 7.012053	10.0.0.10	10.0.0.80	TCP		66 80 → 41039 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
53 7.012135	10.0.0.80	10.0.0.10	TCP		66 41039 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001691 TSecr=170190253
54 7.012191	10.0.0.10	10.0.0.80	TCP		66 80 → 41336 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691
55 7.012242	10.0.0.80	10.0.0.10	TCP		66 41336 → 80 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001692 TSecr=170190253
56 17.925514	207.0.182.48	10.0.0.10	UDP		60 2523 → 80 Len=0
57 18.925756	101.67.71.77	10.0.0.10	UDP		60 2524 → 80 Len=0
58 19.925815	16.1.130.36	10.0.0.10	UDP		60 2525 → 80 Len=0
59 20.925956	72.155.197.97	10.0.0.10	UDP		60 2526 → 80 Len=0
60 21.926014	58.207.80.28	10.0.0.10	UDP		60 2527 → 80 Len=0
61 22.926145	128.95.91.192	10.0.0.10	UDP		60 2528 → 80 Len=0
62 23.926242	92.169.137.224	10.0.0.10	UDP		60 2529 → 80 Len=0
63 24.926476	121.106.77.97	10.0.0.10	UDP		60 2530 → 80 Len=0
64 25.926535	233.108.170.241	10.0.0.10	UDP		60 2531 → 80 Len=0
65 26.926602	105.99.254.0	10.0.0.10	UDP		60 2532 → 80 Len=0
66 34.598235	241.23.224.8	10.0.0.10	TCP		94 1208 → 80 [SYN] Seq=0 Win=512 Len=40 [TCP segment of a reassembled PDU]
67 35.598337	56.99.46.9	10.0.0.10	TCP		94 1209 → 80 [SYN] Seq=0 Win=512 Len=40 [TCP segment of a reassembled PDU]
68 36.598452	71.175.187.74	10.0.0.10	TCP		94 1210 → 80 [SYN] Seq=0 Win=512 Len=40 [TCP segment of a reassembled PDU]
69 37.598502	50.79.56.33	10.0.0.10	TCP		94 1211 → 80 [SYN] Seq=0 Win=512 Len=40 [TCP segment of a reassembled PDU]

Obr. 3.3: Zachytená komunikácia v laboratórnej sieti

3.2.1 NetFlow dáta

Keďže sme chceli pracovať aj s dátami formátu NetFlow, bolo potrebné použiť nástroj, ktorý zbiera dáta sieťovej komunikácie a emituje ich do kolektora už ako NetFlow toky. Na to bol použili nástroj ***fprobe***, ktorý sa nainštaloval na PC-Server, kde zachytával komunikáciu medzi PC-Server a PC-Client. Následne *fprobe* exportoval dáta do PC-Probe. Na odchytávanie a export dát boli použité následujúce príkazy:

- `sudo fprobe -i ens160 10.0.0.50:8000`,
- `/etc/init.d/fprobe start`.

V prvom príkaze sa nastavilo na akom rozhraní má *fprobe* zachytávať komunikáciu a na akú IP adresu a číslo portu má dáta posielať. Použitím druhého príkazu začal *fprobe* pracovať.

Keďže exportovanie NetFlow dát bolo nastavené, bolo potrebné použiť nástroj, ktorý pracuje ako kolektor a dokáže dáta prijať a uložiť. Nástroj ***nfdump***, ktorý bol v teoretickej časti uvedený ako nástroj na anonymizáciu dát, dokáže dáta nielen anonymizovať ale aj zachytávať a zobraziť vďaka jeho ďalším nástrojom, ktoré sú jeho súčasťou. Na prijímanie a ukladanie NetFlow dát bol použitý nástroj ***nfcapd***.

- `sudo nfcapd -w -D -p 8000 -z -I ens160 -l /home/probe/netflow`

Pomocou príkazu sa nastavilo aby *nfcapd* načúval na porte 8000, synchronizoval vytváranie súborov s NetFlow dátami na každých 5 minút a ukladal ich do priečinku `/netflow`. Následne bolo možné pomocou *nfdump* vytvorené súbory s dátami zobraziť v konzolovom okne pomocou príkazu:

- `nfdump -r /home/probe/netflow/nfcapd.201712092150`.

Štatistiku, ktorú *nfdump* zobrazil v konzolovom okne, je na obr. 3.4.

```
Date first seen          Duration Proto      Src IP Addr:Port          Dst IP Addr:Port   Packets    Bytes Flows
2017-12-09 21:50:25.065     0.000 TCP      8.213.150.129:1735  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:30.066     0.000 TCP    251.75.123.211:1740  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:27.065     0.000 TCP      76.26.123.91:1737  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:26.065     0.000 TCP    247.130.133.139:1736  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:29.065     0.000 TCP    224.240.164.143:1739  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:28.065     0.000 TCP      86.213.85.91:1738  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:34.066     0.000 TCP      33.241.254.94:1744  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:33.066     0.000 TCP      86.195.13.56:1743  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:32.065     0.000 TCP      209.72.12.31:1742  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:31.065     0.000 TCP      133.91.52.129:1741  ->        10.0.0.10:80           4      140     1
2017-12-09 21:50:52.825     4.007 TCP         10.0.0.80:45368 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:56.832     0.001 TCP         10.0.0.10:80    ->        10.0.0.80:37337        2      112     1
2017-12-09 21:50:52.825     4.007 TCP         10.0.0.10:80    ->        10.0.0.80:45368        2      112     1
2017-12-09 21:50:51.824     5.008 TCP         10.0.0.80:34007 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:50.822     6.010 TCP         10.0.0.10:80    ->        10.0.0.80:33981        2      112     1
2017-12-09 21:50:49.821     7.011 TCP         10.0.0.80:44523 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:50.822     6.010 TCP         10.0.0.80:33981 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:49.821     7.011 TCP         10.0.0.10:80    ->        10.0.0.80:44523        2      112     1
2017-12-09 21:50:55.830     1.002 TCP         10.0.0.80:42976 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:53.827     3.005 TCP         10.0.0.80:36389 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:54.829     2.004 TCP         10.0.0.10:80    ->        10.0.0.80:43165        2      112     1
2017-12-09 21:50:56.832     0.001 TCP         10.0.0.80:37337 ->        10.0.0.10:80           4      216     1
2017-12-09 21:50:51.824     5.008 TCP         10.0.0.10:80    ->        10.0.0.80:34007        2      112     1
2017-12-09 21:50:55.830     1.003 TCP         10.0.0.10:80    ->        10.0.0.80:42976        2      112     1
2017-12-09 21:50:53.827     3.006 TCP         10.0.0.10:80    ->        10.0.0.80:36389        2      112     1
2017-12-09 21:50:54.828     2.004 TCP         10.0.0.80:43165 ->        10.0.0.10:80           4      216     1
2017-12-09 21:51:04.935     0.000 UDP     13.172.180.105:1824  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:03.935     0.000 UDP      185.13.203.39:1823  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:05.935     0.000 UDP        5.67.77.156:1825  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:09.936     0.000 UDP        7.180.14.48:1829  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:06.935     0.000 UDP      151.250.40.94:1826  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:10.936     0.000 UDP     118.180.157.79:1830  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:08.936     0.000 UDP    228.157.147.226:1828  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:07.936     0.000 UDP      187.239.8.157:1827  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:12.936     0.000 UDP    210.210.112.170:1832  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:11.936     0.000 UDP      132.80.13.183:1831  ->        10.0.0.10:80           1       28     1
2017-12-09 21:51:31.000    50.000 UDP         10.0.0.10:37360 ->        10.0.0.50:8000         7     2092     1
2017-12-09 21:53:31.000     0.000 UDP         10.0.0.10:37360 ->        10.0.0.50:8000         1      100     1
Summary: total flows: 38, total bytes: 6496, total packets: 106, avg bps: 279, avg pps: 0, avg bpp: 61
Time window: 2017-12-09 21:50:25 - 2017-12-09 21:53:31
Total flows processed: 38, Blocks skipped: 0, Bytes read: 2248
Sys: 0.000s flows/second: 0.0        Wall: 0.003s flows/second: 12603.6
```

Obr. 3.4: NetFlow dáta zobrazené nástrojom nfdump

4 ANONYMIZÁCIA SIEŤOVEJ PREVÁDZKY

Táto časť sa bude venovať anonymizovaniu sieťovej prevádzky, ktorá bola vygenerovaná a zachytená vo vytvorenej virtuálnej laboratórnej sieti. Budú na nej testované anonymizačné programy, uvedené v teoretickej časti práce. Bude popísaný postup, ktorý je potrebný pre anonymizáciu pomocou jednotlivých programov, ich možnosti anonymizácie a taktiež výhody a nevýhody. Následne budú výsledky anonymizácie jednotlivých programov porovnané.

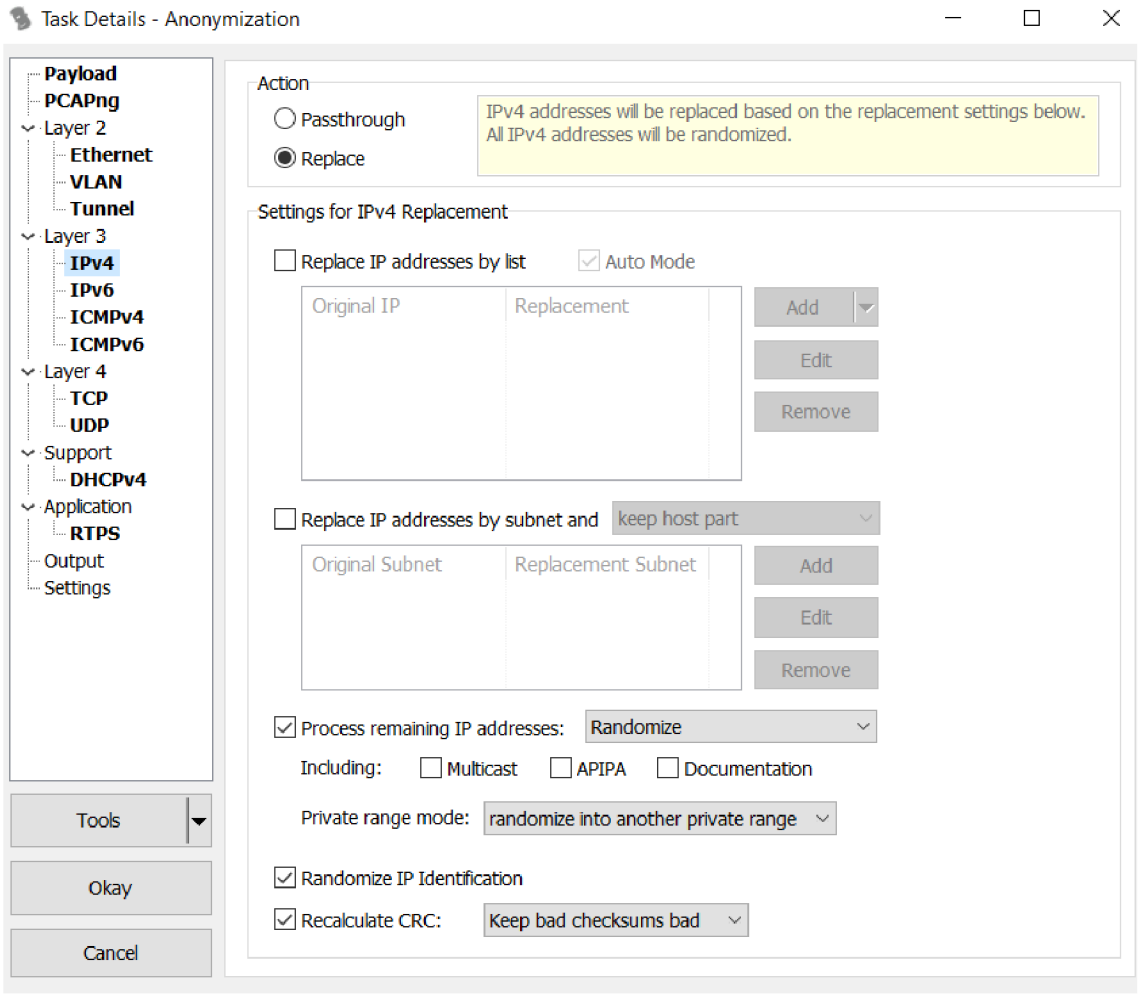
Ako bolo vyššie spomenuté, anonymizovaná bude sieťová prevádzka zachytená vo virtuálnej sieti, ktorej časť je zobrazená na obr. 4.1. Na testovanie budú taktiež použité súbory so sieťovou komunikáciou typu pcap, ktoré sú voľne dostupné na internetovej stránke Austrálskej vojenskej akadémie. Tieto súbory budú na testovanie používané z dôvodu obsahu veľkého množstva dát.

```
    54 7.012191      10.0.0.10           10.0.0.80           TCP      66     80 ГåÆ 41336 [FIN, ACK]
Seq=1 Ack=2 Win=29056 Len=0 TSval=170190253 TSecr=154001691

Frame 54: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: Vmware_0b:3f:44 (00:0c:29:0b:3f:44), Dst: Vmware_19:83:62 (00:0c:29:19:83:62)
Internet Protocol Version 4, Src: 10.0.0.10, Dst: 10.0.0.80
Transmission Control Protocol, Src Port: 80, Dst Port: 41336, Seq: 1, Ack: 2, Len: 0

No.     Time           Source              Destination         Protocol Length Info
    55 7.012242      10.0.0.80           10.0.0.10           TCP      66     41336 ГåÆ 80 [ACK] Seq=2
 Ack=2 Win=29312 Len=0 TSval=154001692 TSecr=170190253

Frame 55: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: Vmware_19:83:62 (00:0c:29:19:83:62), Dst: Vmware_0b:3f:44 (00:0c:29:0b:3f:44)
Internet Protocol Version 4, Src: 10.0.0.80, Dst: 10.0.0.10
Transmission Control Protocol, Src Port: 41336, Dst Port: 80, Seq: 2, Ack: 2, Len: 0

No.     Time           Source              Destination         Protocol Length Info
    56 17.925514     207.0.182.48        10.0.0.10           UDP      60     2523 ГåÆ 80 Len=0

Frame 56: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Ethernet II, Src: Vmware_19:83:62 (00:0c:29:19:83:62), Dst: Vmware_0b:3f:44 (00:0c:29:0b:3f:44)
Internet Protocol Version 4, Src: 207.0.182.48, Dst: 10.0.0.10
User Datagram Protocol, Src Port: 2523, Dst Port: 80

No.     Time           Source              Destination         Protocol Length Info
    57 18.925756     101.67.71.77        10.0.0.10           UDP      60     2524 ГåÆ 80 Len=0

Frame 57: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Ethernet II, Src: Vmware_19:83:62 (00:0c:29:19:83:62), Dst: Vmware_0b:3f:44 (00:0c:29:0b:3f:44)
Internet Protocol Version 4, Src: 101.67.71.77, Dst: 10.0.0.10
User Datagram Protocol, Src Port: 2524, Dst Port: 80

No.     Time           Source              Destination         Protocol Length Info
    58 19.925815     16.1.130.36         10.0.0.10           UDP      60     2525 ГåÆ 80 Len=0

Frame 58: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
```

Obr. 4.1: Sieťová prevádzka virtuálnej siete

4.1 TraceWrangler

Ako bolo spomenuté v teoretickej časti, TraceWrangler pracuje v grafickom rozhraní a po načítaní súboru určeného na anonymizáciu sa zobrazí okno, ktorého ukážka je na obr. 4.2. V ňom je užívateľovi umožnené modifikovať časti a protokoly jednotlivých vrstiev. V prípade, že nie sú upravené žiadne nastavenia, sieťová prevádzka je anonymizovaná defaultnými nastaveniami programu. Taktiež je pri každej záložke možnosť nenahradiť údaje v určitom protokole. V tom prípade nebudú zvolené vrstvy anonymizované.



Obr. 4.2: Nastavenie anonymizácie v programe TraceWrangler

Pretože TraceWrangler nedokáže pracovať so všetkými protokolmi a vrstvami, bola v záložke Payload zvolená možnosť na odstránenie neznámych vrstiev. To zaručilo ochranu pred únikom citlivých častí sieťovej prevádzky, ktoré program nevie spracovať.

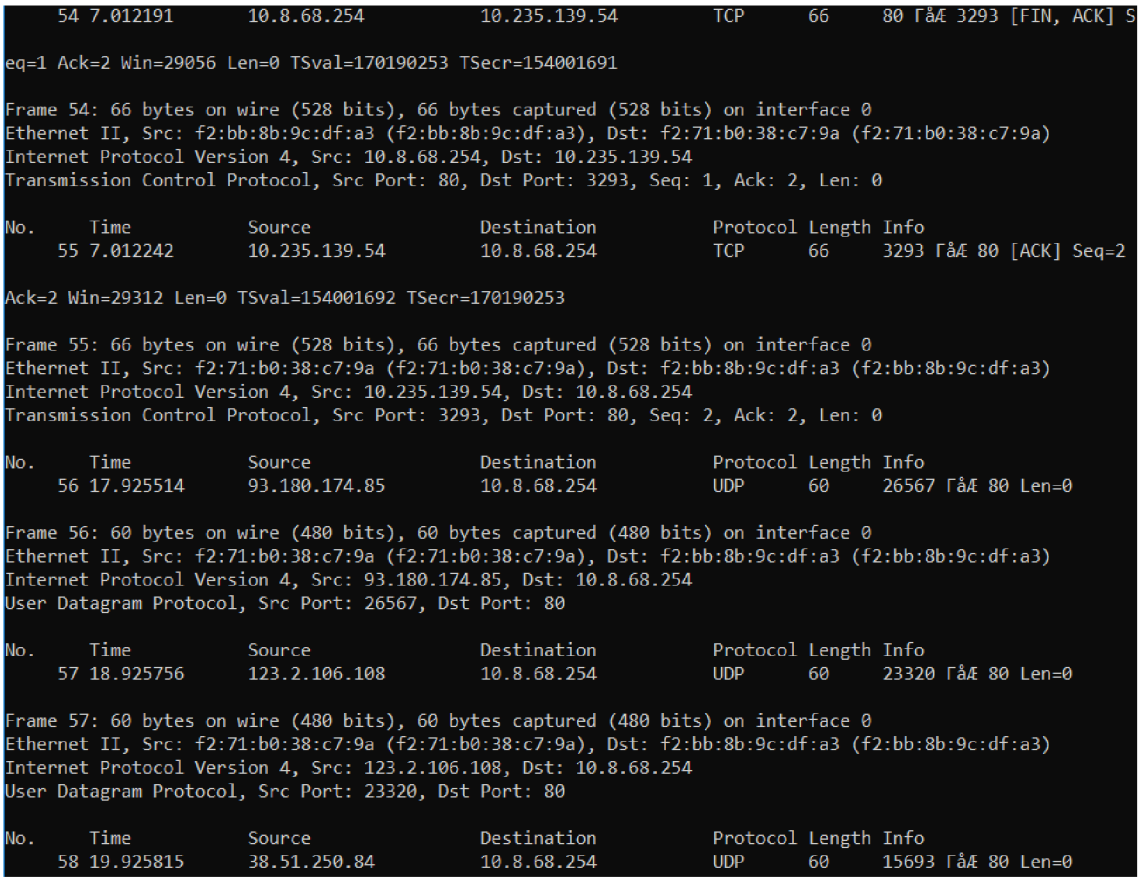
Vo vrstve dva bola nastavená anonymizácia MAC adresy, kde bola zvolená náhodná permutácia adries. Ďalšou možnosťou bolo zvolenie techniky Black Marker, ktorá by MAC adresy v nahratom súbore nahradila nulami. S cieľom zachovať čo najväčšiu výpovednú hodnotu dát pre prípadnú neskoršiu analýzu táto možnosť nebola zvolená.

Vo vrstve tri boli zvolené nastavenia pre anonymizáciu IP adries. Bola zvolená anonymizácia, kedy v prípade výskytu privátnej IP adresy, bola adresa nahradená taktiež privátnou IP adresou. V nastavení IP protokolu bolo zvolené náhodné generovanie identifikátoru. V prípade, že by sa nachádzali v súbore fragmentované pakety, priradil by sa im rovnaký identifikátor. Taktiež bolo zvolené prepočítanie kontrolného súčtu. Tým bola zaručená správna hodnota kontrolného súčtu vo výstupnom súbore, pokiaľ na vstupe paket obsahoval správny kontrolný súčet. V prípade chybného paketu bol kontrolný súčet chybný aj po anonymizácii. Tým bolo umožnené nájsť chybu aj pri analýze modifikovanej sieťovej prevádzky. Pri nastavení anonymizácie IP adries bolo možné manuálne nastaviť konkrétnej IP adrese z načítaného súboru inú konkrétnu adresu.

Vo vrstve štyri boli upravené nastavenia anonymizácie pre protokoly TCP a UDP. Bola zvolená možnosť náhodného priradenia čísla portov s výnimkou čísiel známych portov, 1 – 1024, ktoré zostali nemodifikované. Taktiež bolo zvolené prepočítanie kontrolného súčtu.

Po uložení nastavených parametrov bol PCAP súbor anonymizovaný. Časť anonymizovaných polí je zobrazených na obr. 4.3. Pri porovnaní anonymizovaného súboru a súboru z obr. 4.1, je viditeľná zmena cieľových a zdrojových MAC adries a taktiež zdrojových a cieľových IP adries. Na transportnej vrstve boli anonymizované len privátne čísla portov. Port 80, ktorý patrí do kategórie známych portov zostal nemodifikovaný. Pri porovnaní MAC a IP adries je viditeľné, že ak sa v súbore opakovane nachádzal rámec respektíve paket s rovnakou adresou bola na anonymizáciu zvolená taktiež len jedna adresa, čo zachovalo integritu dát napríklad pri naväzovaní TCP spojenia. Taktiež v prípade výskytu ARP protokolu boli v poliach tohto protokolu anonymizované IP a MAC adresy.

Program Trace Wrangler bol testovaný sieťovou prevádzkou v rozsahu desiatok kilobytov (kB) až po veľkosť jeden gigabyte (GB). Podľa doby trvania procesu anonymizácie je program vhodný skôr pre prácu s menšími súbormi, pretože pri veľkosti 1 GB doba procesu trvala cca 30 minút.



Obr. 4.3: Anonymizované polia programom TraceWrangler

4.2 SCRUB-tcpdump

Anonymizácia sieťovej prevádzky pomocou nástroja SCRUB-tcpdump bola uskutočnená na operačnom systéme Ubuntu (distribúcia Linuxu). Program pracoval v príkazovom riadku, pretože doposiaľ nie je rozšírený o grafické rozhranie. Aby prebehol proces anonymizácie, bolo potrebné nastaviť parameter s cestou ku súboru formátu PCAP a názov výstupného súboru, kde bola uložená anonymizovaná sieťová prevádzka. Ďalší parameter sa nastavoval vo formáte string, kde boli vybraté polia určené k anonymizácii. Ku každému zvolenému poľu bolo potrebné priradiť anonymizačnú techniku. Polia, ktoré bolo možné zvoliť, patrili protokolom sieťovej a transportnej vrstvy. Boli to protokoly IPv4, TCP a UDP. Okrem týchto dvoch vrstiev bolo možné anonymizovať časové razítko, ktoré patri do PCAP poľa. Program automaticky prepočítava kontrolné súčty vyššie uvedených protokolov.

S cieľom odstrániť citlivé informácie zo sieťovej prevádzky, ale zároveň zachovať čo najväčšiu výpovednú hodnotu anonymizovaných dát pre prípadnú analýzu alebo vytváranie štatistík, bol vykonaný nasledujúci príkaz:

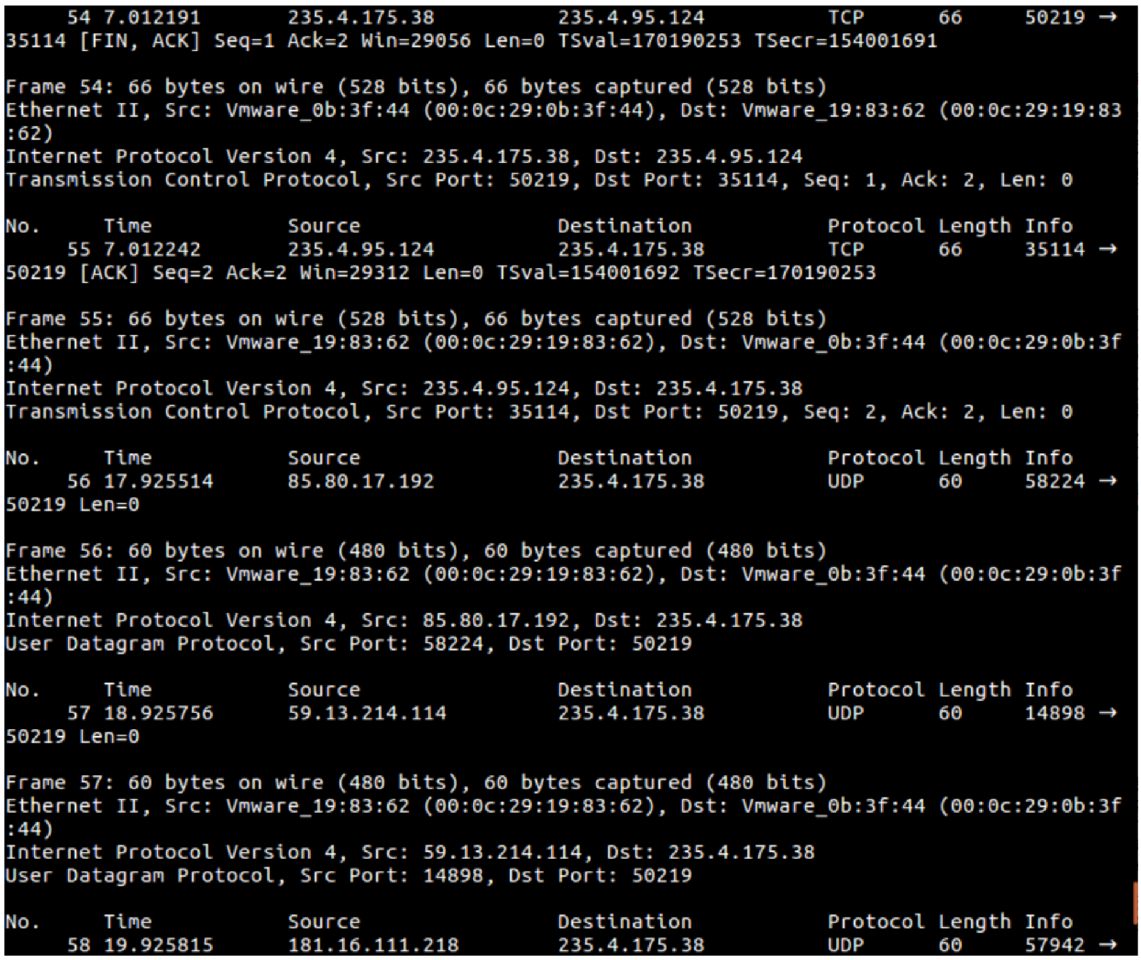
- `./scrub-tcpdump -r NetworkTrafficRepair.pcap -w NetworkTraffic Anon.pcap -k 12345 -o "srcip pp dstip pp tcpsrcport rp tcpdstport rp udpsrcport rp udpdstport rp"`

Parametrom `-r` bol určený vstupný súbor a parametrom `-w` výstupný súbor. Parametrom `-o` sa definoval textový reťazec typu string, v ktorom boli uvedené nasledujúce polia a anonymizačné techniky. Pre zdrojovú a cieľovú IP adresu (`srcip`, `dstip`) bola zvolená anonymizácia IP adresy so zachovaním prefixu. Pre zdrojové a cieľové čísla portov TCP segmentov a UDP datagramov (`tcpsrcport`, `tcpdstport`, `udpsrcport`, `udpdstport`) bola technika náhodného priradenia čísla portu pomocou kľúča, ktorý bol definovaný parametrom `-k`. Časť anonymizovanej sieťovej prevádzky je zobrazená na obr. 4.4. Pri porovnaní zobrazenej sieťovej prevádzky s obr. 4.1, cieľová a zdrojová MAC adresa zostala nezmenená, pretože SCRUB-tcpdump nedokáže pracovať so spojovou vrstvou.

Pri anonymizácii IP adries je viditeľné, že je použitá technika zo zachovaním prefixu, kedy IP adresy s rovnakými prvými dvoma oktetami boli nahradené jednou spoločnou hodnotou. Zvyšné dva oktety patrili hosťom a tie boli anonymizované zvlášť. Ak sa ale zvyšné dva oktety v sieťovej prevádzke opakovali, boli nahradené rovnakou hodnotou .

Na transportnej vrstve boli anonymizované čísla portov. Program nerozlišoval, či ide o známy port, napríklad port 80, alebo privátne číslo portu a nahradil ich hodnotou, vypočítanou podľa zadaného kľúča. Táto anonymizačná technika bola zvolená na základe toho, že pokiaľ sa vo vstupnom súbore čísla portov opakovali vo viacerých segmentoch a datagramoch, boli nahradené rovnakou vygenerovanou hodnotou. Na anonymizáciu portov bolo možné použiť ďalšie dve techniky. V prvej by bola každému portu priradená náhodná hodnota bez ohľadu na to, či sa číslo portu v súbore opakuje, alebo nie. Pomocou druhej techniky by bola známym portom (1 – 1024) priradená 0 a privátnym portom priradená hodnota 65535. Preto bola zvolená technika, kde je v sieťovej prevádzke viditeľné, že komunikácia prebieha medzi viacerými uzlami.

Pokiaľ sa vo vstupnom súbore nachádzali protokoly z aplikačnej vrstvy (DNS, HTTP, DHCP, SSDP atď.), vo výstupnom súbore boli zobrazené len ako UDP datagramy alebo TCP segmenty. Teda najvyššia zobrazená vrstva bola transportná. Avšak pri analýze výstupného súboru po procese anonymizácie programom Wireshark, bolo dostupné ďalšie pole s názvom dáta. Wireshark ponúka možnosť zobraziť dáta paketu. Aj keď po anonymizácii bolo možné zobraziť najvyššiu vrstvu transportnú, v spomenutej možnosti bolo možné zobraziť dáta protokolov vyšších vrstiev. Ako je zobrazené na obr. 4.5, v prípade, že sa vo vstupnom súbore nachádzal pro-



Obr. 4.4: Anonymizované polia programom SCRUB-tcpdump

tokol SSDP (Simple Service Discovery Protokol), vo výstupnom súbore bolo možné zobraziť v dátach informácie ako sú neanonymizované IP adresy. Pri protokole DNS bolo možné zistiť názov webovej adresy.

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=100
LOCATION: http://192.168.0.1:1900/igd.xml
NT: upnp:rootdevice
NTS: ssdp:alive
SERVER: ipos/7.0 UPnP/1.0 TL-WR841N/9.0
USN: uuid:060b7353-fca6-4070-85f4-1fbfb9add62c::upnp:rootdevice
```

Obr. 4.5: Neanonymizované informácie SSDP protokolu

Ak sa v súbore nachádzal ARP protokol, program nedokázal anonymizovať jeho

polia. ARP protokol nachádzajúci sa vo výstupnom anonymizovanom súbore je zobrazený na obr. 4.6. Keďže SCRUB-tcpdump nemodifikuje MAC adresy, podľa nej je možné zistiť jednu z IP adries vstupnej sieťovej prevádzky. Preto je potrebné ARP protokol pred anoymizáciou zo súboru odstrániť.

```
No.     Time           Source                Destination           Protocol Length Info
     18 5.007630       Vmware_0b:3f:44       Vmware_19:83:62       ARP      42     Who has
10.0.0.80? Tell 10.0.0.10

Frame 18: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
Ethernet II, Src: Vmware_0b:3f:44 (00:0c:29:0b:3f:44), Dst: Vmware_19:83:62 (00:0c:29:19:83
:62)
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Vmware_0b:3f:44 (00:0c:29:0b:3f:44)
    Sender IP address: 10.0.0.10
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.0.0.80
```

Obr. 4.6: Polia ARP protokolu

Program bol podobne ako TraceWrangler testovaný pri rôznych veľkostiach vstupných súborov. Pri testovaní súboru o veľkosti 650 MB neindikoval žiadnu chybu, ale pri porovnaní vstupného a výstupného súboru zostali všetky parametre rovnaké. Ďalší testovaný súbor mal veľkosť 310 MB a pri tejto veľkosti program dokázal modifikovať polia pomocou parametrov, ktoré boli zvolené identicky ako vo vyššie uvedenom príkaze.

4.3 Capsan

Program Capsan rovnako ako SCRUB-tcpdump pracuje v príkazovom riadku a anonymizácia bola vykonávaná v operačnom systéme Ubuntu. Užívateľské možnosti tohto programu sú na rozdiel od predchádzajúcich dvoch programov (TraceWrangler, SCRUB-tcpdump) značne obmedzené a preto je nastavenie anonymizácie pomerne jednoduché.

Pre spustenie procesu anonymizácie sieťovej prevádzke bolo potrebné nastaviť cestu ku vstupnému PCAP súboru a názov výstupného súboru. Ostatné možnosti boli voliteľné. Zvolený súbor bol modifikovaný pomocou tohto príkazu:

- `./capsan -r  /Desktop/skuskaRepaired/NetworkTraffic.pcap -w NetworkTrafficAnon.pcap -k 12345 -a AddressMap -p PortMap`

Parametrom `-r` bol určený vstupný súbor a parametrom `-w` názov výstupného súboru. Parametrom `-k` bol určený kľúč pomocou ktorého boli generované nové IP adresy a čísla portov UDP a TCP protokolov. Parameterom `-a` sa nastavilo mapovanie anonymizovaných IP adries k neanonymizovaným. Zároveň sa týmto parametrom vytvoril textový súbor s mapovanými IP adresami. Parameter `-p` funguje rovnako ako parameter `-a`, ale platí pre čísla portov. Výstupný anonymizovaný súbor je zobrazený na obr. 4.7. Pri porovnaní anonymizovaného súbora so sieťovou prevádz-

```
     52 7.012191       152.121.229.20        152.121.229.115       TCP      66     31759 →
 57323 [FIN, ACK] Seq=1 Ack=2 Win=29056 [TCP CHECKSUM INCORRECT] Len=0 TSval=170190253 TSe
cr=154001691

Frame 52: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:
00:00:00)
Internet Protocol Version 4, Src: 152.121.229.20, Dst: 152.121.229.115
Transmission Control Protocol, Src Port: 31759, Dst Port: 57323, Seq: 1, Ack: 2, Len: 0

No.     Time           Source                Destination           Protocol Length Info
     53 7.012242       152.121.229.115       152.121.229.20        TCP      66     57323 →
 31759 [ACK] Seq=2 Ack=2 Win=29312 Len=0 TSval=154001692 TSecr=170190253

Frame 53: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:
00:00:00)
Internet Protocol Version 4, Src: 152.121.229.115, Dst: 152.121.229.20
Transmission Control Protocol, Src Port: 57323, Dst Port: 31759, Seq: 2, Ack: 2, Len: 0

No.     Time           Source                Destination           Protocol Length Info
     54 17.925514      81.45.106.238         152.121.229.20        UDP      60     4268 →
31759 Len=0

Frame 54: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:
00:00:00)
Internet Protocol Version 4, Src: 81.45.106.238, Dst: 152.121.229.20
User Datagram Protocol, Src Port: 4268, Dst Port: 31759

No.     Time           Source                Destination           Protocol Length Info
     55 18.925756      255.88.47.42          152.121.229.20        UDP      60     15508 →
 31759 Len=0

Frame 55: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:
00:00:00)
Internet Protocol Version 4, Src: 255.88.47.42, Dst: 152.121.229.20
User Datagram Protocol, Src Port: 15508, Dst Port: 31759

No.     Time           Source                Destination           Protocol Length Info
     56 19.925815      132.101.57.140        152.121.229.20        UDP      60     8769 →
```

Obr. 4.7: Anonymizované polia programom Capsan

kou vstupného súbora zobrazenej na obr. 4.1 je viditeľné, že na spojovej vrstve boli zdrojová aj cieľová MAC adresa nahradené nulami. To znamená, že program Capsan defaultne používa techniku Black Marker na anonymizovanie MAC adries. Inú anonymizačnú techniku nie je možné použiť.

Na sieťovej vrstve sú anonymizované IP adresy, kde pri nahradzovaní IP adries bola použitá technika anonymizácie IP adries zo zachovaním prefixu. Pokiaľ sa dve

adresy líšili len hodnotou posledného oktetu, tak aj anonymizované adresy boli v poslednom oktete rozdielne. Pokiaľ sa vo vstupnom súbore adresa viac krát opakovala, bola nahradená rovnakou IP adresou. Program generoval IP adresy náhodne a teda nerozlišoval či sa jednalo o privátnu, alebo verejnú IP adresu.

Na transportnej vrstve program anonymizoval polia s číslami portov. Bola aplikovaná technika mapovania portov jedna k jednej. Pokiaľ sa číslo portu vo vstupnom súbore opakovalo, bolo nahradené jednou vygenerovanou hodnotou. Program taktiež nerozlišoval známe a privátne porty.

Capsan automaticky prepočítaval kontrolné súčty protokolov IPv4, TCP a UDP. V prípade, že sa vo vstupnom súbore nachádzal paket, segment alebo datagram s nesprávnym kontrolným súčtom, vo výstupnom súbore bol pri konkrétnom protokole taktiež nesprávny kontrolný súčet.

Capsan spracúva len dáta, ktoré obsahujú IPv4 protokol a na transportnej vrstve TCP a UDP protokoly. V prípade, že sa v sieťovej prevádzke tieto protokoly nenachádzajú, daný paket je z výstupného súboru odstránený. Jedná sa napríklad o ARP protokol alebo IPv6 protokol. Capsan taktiež nespracúva IP fragmenty a orezané pakety. O tom, koľko paketov nebolo spracovaných, informuje po vykonaní procese anonymizácie ako je zobrazené na obr. 4.8.

```
lukas@lukasUbuntu:~/Desktop/capsan-build$ ./capsan  -r ~/Desktop/skuskaRepaired/
NetworkTrafficRepair.pcap -w NetworkTrafficAnon.pcap -k 12345 -a AddressMap -p P
ortMap
Packets read:                80
Packets written:             74
Skipped truncated packets:   1
Skipped non-IPv4 packets:    4
Skipped non-UDP/TCP packets: 1
Skipped IP fragments:        0
```

Obr. 4.8: Výpis nespracovaných paketov programom Capsan

V prípade, že nie sú zadané parametre pre ukladanie mapovania portov alebo IP adries, ponúka Capsan možnosť deanonymizovať adresu alebo port. V tomto prípade je potrebné pri nastavení parametrov anonymizácie zvoliť aj kľúč, podľa ktorého je možné spätne zistiť pôvodnú hodnotu. Táto funkcia bola využitá zadaním nasledujúceho príkazu:

- `./capsan -n 152.121.229.20 -k 12345 -reverse`

Po zadaní príkazu sa v príkazovom riadku vypísala IP adresa `10.0.0.10`, ktorá odpovedala adrese vo vstupnom súbore a bola nahradená adresou `152.121.229.20`.

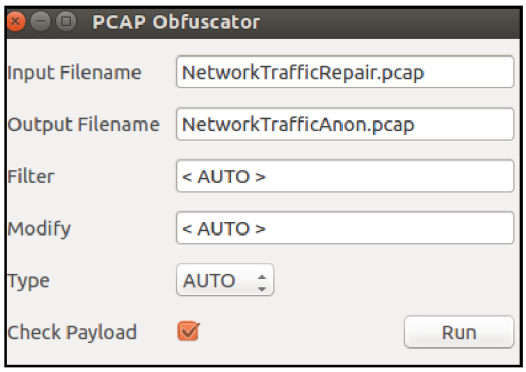
Po anonymizovaní programom Capsan, rovnako ako Scrub-tcpdump, bolo možné pri analýze výstupného súboru zobraziť najvyššie polia protokolov transportnej

vrstvy a taktiež bolo možné pomocou programu Wireshark zistiť neanonymizované informácie z určitých protokolov aplikačnej vrstvy.

Pri testovaní programu vstupnými PCAP súbormi rôznych veľkostí, Capsan dokázal spracovať a anonymizovať súbory od najmenších súborov až po maximálnu testovanú veľkosť súboru 2 GB.

4.4 Pcap obfuscator

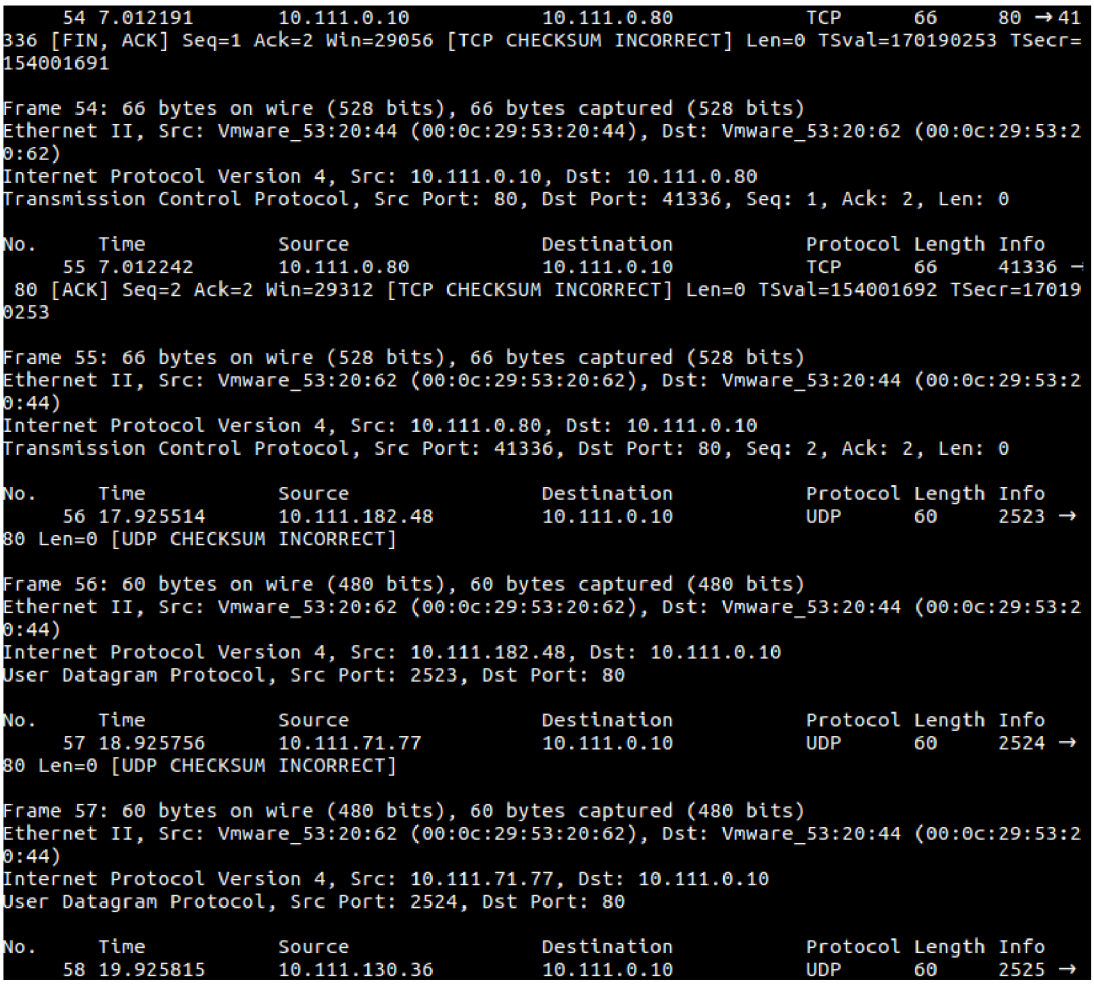
Táto aplikácia umožňovala nastaviť parametre anonymizácie sieťovej prevádzky pomocou jednoduchého grafického rozhrania, ktoré je zobrazené aj s nastavenými parametrami na obr. 4.9. Okrem nahrania vstupného PCAP súboru a zadania názvu výstupného súboru, bolo možné nastaviť anonymizáciu konkrétnej IP a MAC adresy alebo VLAN ID. Bola ponechané nastavenie **< AUTO >**, čo znamenalo anonymizovanie adries prípadne VLAN ID v celom súbore. Zaškrtnutá možnosť **check payload** znamenala, že program modifikoval IP adresy v textovo orientovaných protokoloch ako sú HTTP a SIP (Session Initiation Protocol).



Obr. 4.9: Grafické užívateľské rozhranie programu Pcap obfuscator

Po vykonaní procesu anonymizácie je časť sieťovej prevádzky zobrazená na obr. 4.10. Pri porovnaní modifikovanej sieťovej prevádzky z obr. 4.1, sú viditeľné zmeny vo výstupnom súbore. Pcap obfuscator anonymizoval MAC adresy na spojovej vrstve tak, že prvé tri byty ponechal rovnaké, čo spôsobilo zachovanie identity výrobcu. Modifikované boli až zvyšné tri byty. Pokiaľ sa vo vstupnom súbore konkrétna MAC adresa opakovala, bola nahradená len jednou modifikovanou MAC adresou.

Na sieťovej vrstve program anonymizoval IP adresy tak, že v celom súbore so sieťovou prevádzkou nahradil prvé dva oktety rovnakými hodnotami a pre posledné dva oktety už generoval náhodné hodnoty. Avšak ako aj pri MAC adresách, tak aj pri IP adresách bolo zachované mapovanie jedna k jednej.



Obr. 4.10: Anonymizované polia programom Capsan

Vo výstupnom súbore ostali čísla portov nezmenené. Na rozdiel od vyššie testovaných programov, Pcap obfuscator nedokázal správne prepočítať kontrolný súčet, a všetky pakety IPv4 protokolu, UDP datagramy a TCP segmenty mali nesprávny kontrolný súčet. Program taktiež nedokázal anonymizovať polia v ARP protokole a ponechával tento protokol vo výstupnom súbore. Pokiaľ sa vo vstupnom súbore nachádzali protokoly vyšších vrstiev ako je sieťová vrstva boli zobrazené aj vo výstupnom súbore. To znamená, že Pcap obfuscator len modifikoval polia ako sú zdrojová a cieľová MAC adresa a IPv4 adresa. Ostatné protokoly sa nachádzali vo výstupnom súbore neanonymizované.

Pri testovaní súborov rôznych veľkostí Pcap obfuscator dokázal anonymizovať súbor maximálne o veľkosti 100 MB. Pri testovaní väčšieho objemu dát sieťovej prevádzky, vstupné súbory spracoval ale vytvori len prázdny výstupný PCAP súbor.

4.5 NFDUMP

Program nfdump pracuje v príkazovom riadku v linuxových distribúciach. Na anonymizáciu netflow dát bol použitý jeden zo sady nástrojov tohto programu a to **nfanon**. Časť zachytených netflow dát v sieťovej prevádzke je zobrazených na obr.4.11. Nástroj nfanon neumožňuje nastavovať parametre anonymizácie. Anonymizuje len

Date first seen	Duration	Proto	Src IP Addr:Port		Dst IP Addr:Port	Packets	Bytes	Flows
2017-12-09 18:31	0.000	ICMP	10.0.0.50:0	->	10.0.0.10:3.3	1	556	1
2017-12-09 18:30	0.000	TCP	173.90.159.0:1214	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	UDP	105.99.254.0:2532	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	TCP	241.23.224.8:1208	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	TCP	56.99.46.9:1209	->	10.0.0.10:80	1	60	1
2017-12-09 18:31	15.000	UDP	10.0.0.10:52164	->	10.0.0.50:8000	2	1072	1
2017-12-09 18:30	3.006	TCP	10.0.0.10:80	->	10.0.0.80:38703	3	104	1
2017-12-09 18:30	0.000	TCP	10.0.0.10:80	->	10.0.0.80:33328	1	32	1
2017-12-09 18:30	1.003	TCP	10.0.0.10:80	->	10.0.0.80:41039	3	104	1
2017-12-09 18:30	0.002	TCP	10.0.0.10:80	->	10.0.0.80:41336	3	104	1
2017-12-09 18:30	0.000	TCP	10.0.0.10:80	->	10.0.0.80:40348	2	64	1
2017-12-09 18:30	0.000	TCP	10.0.0.10:80	->	10.0.0.80:38566	1	32	1
2017-12-09 18:30	2.004	TCP	10.0.0.10:80	->	10.0.0.80:42450	3	104	1
2017-12-09 18:30	0.000	TCP	10.0.0.10:80	->	10.0.0.80:35544	1	32	1
2017-12-09 18:30	0.000	UDP	58.207.80.28:2527	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	TCP	50.79.56.33:1211	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	UDP	16.1.130.36:2525	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	UDP	207.0.182.48:2523	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	TCP	71.175.187.74:1210	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	UDP	101.67.71.77:2524	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	3.006	TCP	10.0.0.80:38703	->	10.0.0.10:80	4	136	1
2017-12-09 18:30	0.001	TCP	10.0.0.80:33328	->	10.0.0.10:80	2	64	1
2017-12-09 18:30	1.003	TCP	10.0.0.80:41039	->	10.0.0.10:80	4	136	1
2017-12-09 18:30	0.002	TCP	10.0.0.80:41336	->	10.0.0.10:80	4	136	1
2017-12-09 18:30	0.001	TCP	10.0.0.80:40348	->	10.0.0.10:80	2	64	1
2017-12-09 18:30	0.001	TCP	10.0.0.80:38566	->	10.0.0.10:80	2	64	1
2017-12-09 18:30	2.004	TCP	10.0.0.80:42450	->	10.0.0.10:80	4	136	1
2017-12-09 18:30	0.001	TCP	10.0.0.80:35544	->	10.0.0.10:80	2	64	1
2017-12-09 18:30	0.000	TCP	247.150.175.85:1217	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	UDP	121.106.77.97:2530	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	UDP	72.155.197.97:2526	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	TCP	138.9.34.103:1212	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	TCP	60.124.57.131:1213	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	TCP	59.209.201.182:1215	->	10.0.0.10:80	1	60	1
2017-12-09 18:30	0.000	UDP	128.95.91.192:2528	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	UDP	92.169.137.224:2529	->	10.0.0.10:80	1	8	1
2017-12-09 18:30	0.000	UDP	233.108.170.241:2531	->	10.0.0.10:80	1	8	1

Obr. 4.11: Sieťová prevádzka vo formáte netflow

IP adresy a používa techniku anoymizácie IP adresy so zachovaním prefixu. Bolo možné zvoliť cestu k vstupnému súboru, názov výstupného súboru a reťazec o dĺžke 32 znakov, pomocou ktorých bola modifikácia zdrojových a cieľových adries protokolu IPv4 vykonávaná. Príkaz na spustenie vyzeral nasledovne:

- `nfanon -r  /Desktop/netflow/nfcapd.201712091830 -w /Desktop/netflowAnon/AnonFile -K 01234567890123456789012345678901`

Anonymizovaný výstupný súbor je zobrazený na obr. 4.12. Ako bolo spomenuté, program použil na modifikáciu IP adries anonymizáciu zo zachovaním prefixu. Nerozlišoval, či sa jednalo o privátne alebo verejné adresy. Okrem polí s IP adresami,

iné polia nfanon neanonymizoval.

Nfanon ponúkal ešte možnosť spracovať viacero vstupných súborov, anonymizovať ich a uložiť do jedného výstupného súboru. Testované súbory sa oproti súborom s PCAP dátami líšili hlavne menšou veľkosťou. Ich veľkosť sa pohybovala v jednotkách MB. Tieto veľkosti netflow súborov anonymizoval nástroj nfanon bezchybne.

Date first seen	Duration	Proto	Src IP Addr:Port		Dst IP Addr:Port	Packets	Bytes	Flows
2017-12-09 18:31	0.000	ICMP	117.207.252.111:0	->	117.207.252.72:3.3	1	556	1
2017-12-09 18:30	0.000	TCP	178.162.151.12:1214	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	UDP	53.156.9.179:2532	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	TCP	246.171.193.252:1208	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	TCP	74.28.177.141:1209	->	117.207.252.72:80	1	60	1
2017-12-09 18:31	15.000	UDP	117.207.252.72:52164	->	117.207.252.111:8000	2	1072	1
2017-12-09 18:30	3.006	TCP	117.207.252.72:80	->	117.207.252.44:38703	3	104	1
2017-12-09 18:30	0.000	TCP	117.207.252.72:80	->	117.207.252.44:33328	1	32	1
2017-12-09 18:30	1.003	TCP	117.207.252.72:80	->	117.207.252.44:41039	3	104	1
2017-12-09 18:30	0.002	TCP	117.207.252.72:80	->	117.207.252.44:41336	3	104	1
2017-12-09 18:30	0.000	TCP	117.207.252.72:80	->	117.207.252.44:40348	2	64	1
2017-12-09 18:30	0.000	TCP	117.207.252.72:80	->	117.207.252.44:38566	1	32	1
2017-12-09 18:30	2.004	TCP	117.207.252.72:80	->	117.207.252.44:42450	3	104	1
2017-12-09 18:30	0.000	TCP	117.207.252.72:80	->	117.207.252.44:35544	1	32	1
2017-12-09 18:30	0.000	UDP	73.52.209.163:2527	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	TCP	69.180.212.98:1211	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	UDP	107.207.129.139:2525	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	UDP	207.191.182.51:2523	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	TCP	0.135.191.203:1210	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	UDP	60.195.71.205:2524	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	3.006	TCP	117.207.252.44:38703	->	117.207.252.72:80	4	136	1
2017-12-09 18:30	0.001	TCP	117.207.252.44:33328	->	117.207.252.72:80	2	64	1
2017-12-09 18:30	1.003	TCP	117.207.252.44:41039	->	117.207.252.72:80	4	136	1
2017-12-09 18:30	0.002	TCP	117.207.252.44:41336	->	117.207.252.72:80	4	136	1
2017-12-09 18:30	0.001	TCP	117.207.252.44:40348	->	117.207.252.72:80	2	64	1
2017-12-09 18:30	0.001	TCP	117.207.252.44:38566	->	117.207.252.72:80	2	64	1
2017-12-09 18:30	2.004	TCP	117.207.252.44:42450	->	117.207.252.72:80	4	136	1
2017-12-09 18:30	0.001	TCP	117.207.252.44:35544	->	117.207.252.72:80	2	64	1
2017-12-09 18:30	0.000	TCP	240.149.183.213:1217	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	UDP	41.98.180.114:2530	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	UDP	8.164.5.109:2526	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	TCP	137.201.65.136:1212	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	TCP	76.7.217.124:1213	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	TCP	72.47.245.87:1215	->	117.207.252.72:80	1	60	1
2017-12-09 18:30	0.000	UDP	132.95.107.192:2528	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	UDP	18.159.139.12:2529	->	117.207.252.72:80	1	8	1
2017-12-09 18:30	0.000	UDP	234.25.169.13:2531	->	117.207.252.72:80	1	8	1

Obr. 4.12: Anonymizovaná sieťová prevádzka nástrojom nfanon

4.6 Výsledky testovania

Sieťová prevádzka bola testovaná pomocou piatich anonymizačných programov. Štyri z nich pracovali zo súbormi typu pcap a posledný, Nfdump, pracoval s netflow dátami.

Program TraceWrangler pracoval v grafickom rozhraní. Ponúkal možnosti modifikovať protokoly sieťovej, spojovej a transportnej vrstvy. Napriek tomu, že program nevie pracovať so všetkými vrstvami a protokolmi, bolo možné nastaviť odstránenie neznámych vrstiev, čo zabránilo úniku citlivých informácií. Ďalšou výhodou tohto

programu bolo inteligentné nahradzovanie privátnych, verejných alebo multicastových IP adries rovnakou skupinou adries. Touto funkciou bola zachovaná vysoká výpovedná hodnota výstupných anonymizovaných dát. Nevýhodou TraceWrangleru bola doba trvania anonymizácie. Pri vstupnom súbore o veľkosti 1 GB trval proces anonymizácie približne 30 minút. Program taktiež mohol ponúkať anonymizáciu viacerých polí hlavičiek protokolov IPv4 a protokolov transportnej vrstvy (TCP, UDP).

Anonymizácia v programe Scrub-tcpdump bola vykonávaná v príkazovom riadku. Program dokázal spracovať protokoly IPv4, UDP a TCP. Taktiež umožňoval anonymizovať časové razítko pcap poľa. Bolo možné využiť viacero druhov anonymizačných techník pre konkrétne polia hlavičiek spomenutých protokolov. Užívateľ musel v tomto prípade poznať význam každého poľa, pretože zmenou hodnoty v konkrétnom poli, mohla byť ovplyvnená vyššia vrstva TCP/IP modelu. Každé pole a anonymizačná technika mala špecifické označenie vo forme skratky a priraďovala sa do textového reťazca typu string. Užívateľ musel poznať každú skratku, čo vo vysokej miere zvyšovalo náročnosť zadávania parametrov. V prípade, že bol nejaký údaj chybne zadaný, program chybu nerozpoznal, príkaz spracoval a vzniknutú chybu bolo možné odhaliť až pri kontrole výstupného súboru. Program taktiež nedokázal pracovať so spojovou vrstvou, protokolmi ako sú ARP a protokolmi aplikačnej vrstvy. Pri dôkladnej analýze výstupného súboru bolo možné zistiť nezmenené hodnoty vstupného súboru. Program dokázal spracovať vstupné súbory o veľkosti 650 MB.

Program Capsan taktiež pracoval v príkazovom riadku a dokázal anonymizovať zdrojové a cieľové adresy protokolu IPv4 a čísla portov protokolov TCP a UDP. Program umožňoval zvoliť, ktoré zo spomenutých polí možno modifikovať. Anonymizoval taktiež MAC adresu technikou Black Marker, čo bolo nastavené defaultne a nebolo možné MAC adresy zachovať. V prípade, že sa vo vstupnom súbore nachádzali protokoly, ktoré program nespracúva, boli odstránené. Tým bolo zabránené úniku citlivých informácií. Po ukončení procesu anonymizovania, bolo zobrazené koľko paketov bolo zo súboru odstránených. Capsan dokázal anonymizovať všetky použité sieťové prevádzky až do veľkosti 2GB.

Pcap obfuscator pracoval pomocou grafického rozhrania a dokázal modifikovať IPv4 adresy a MAC adresy. Ako jediný z testovaných programov nedokázal pri procese anonymizácie prepočítať kontrolný súčet a anonymizovať čísla portov protokolov UDP a TCP. Ostatné protokoly nedokázal modifikovať a preto sa na výstupe mohli objaviť napríklad pôvodné adresy v iných protokoloch. Pcap obfuscator dokázal spracovať súbor o veľkosti maximálne 100 MB, čo bolo najmenej spomedzi všetkých testovaných programov.

Nfdump ako jediný pracoval s netflow dátami. Dokázal anonymizovať len IP

adresu technikou so zachovaním prefixu. Použité netflow dáta mali značne menšiu veľkosť oproti dátam formátu pcap a to v jednotkách MB. Program dokázal spracovať všetky použité sieťové prevádzky.

Zvyšné tri programy popísané v teoretickej časti neboli testované z viacerých dôvodov. Program Anonym tool bolo možné spustiť pomocou programu Matlab, ale program dokázal anonymizovať len pcap súbor v textovom súbore, ktorý bol priložený k programu ako testovací. Sieťové prevádzky použité pri testovaní ostatných programov dokázal načítať ale pri spustení procesu anonymizácie program zobrazil z nezisteného dôvodu chybové hlásenia. Pri inštalácii programu Anontool boli zobrazované chyby, ktoré sa nepodarilo odstrániť a program nemohol byť spustený. Internetová stránka, kde sa mal nachádzať program Flaim, obsahovala podrobný popis práce s programom, ale program v tom čase nebol na stránke dostupný.

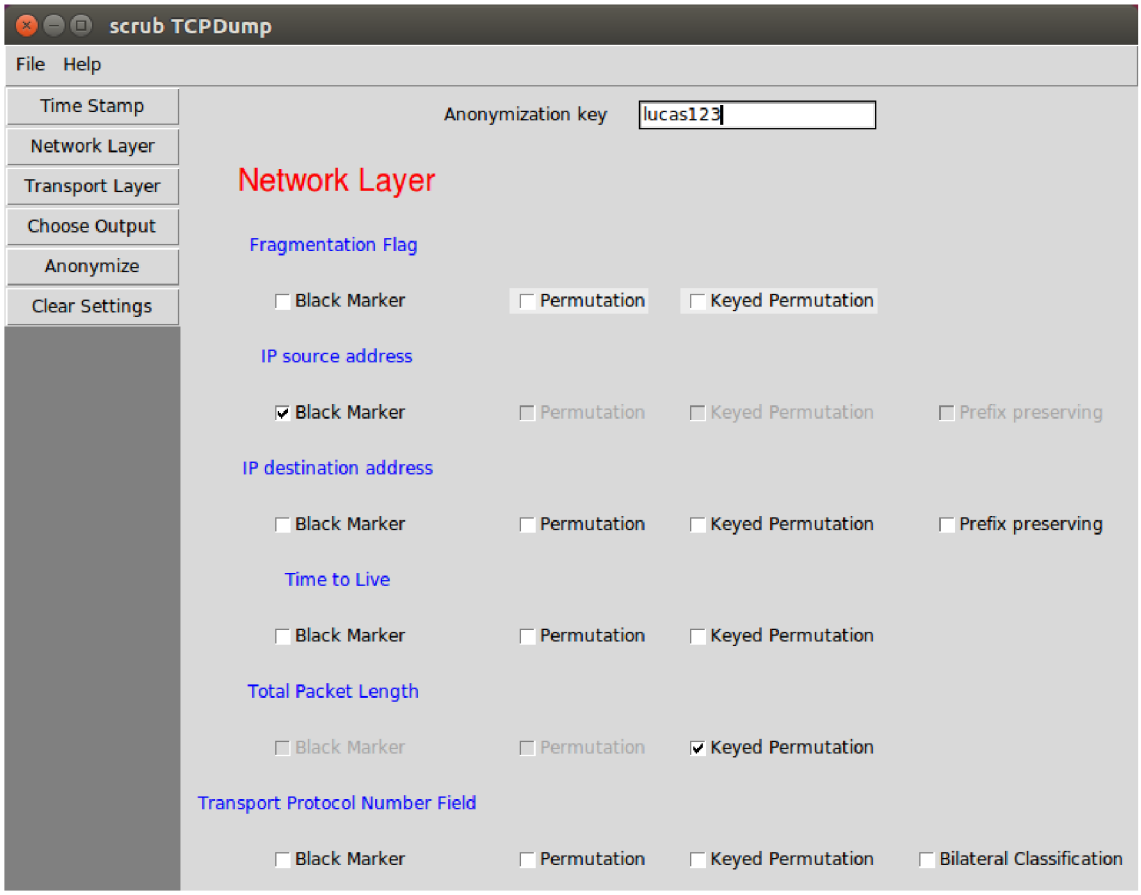
Medzi najlepšie pracujúce anonymizačné programy možno zaradiť TraceWrangler, pretože dokázal spoľahlivo odstrániť citlivé informácie zo všetkých vrstiev sieťovej prevádzky a program Scrub-tcpdump, kvôli širokým možnostiam anonymizácie viacerých polí spomenutých hlavičiek protokolov. Pre zlepšenie a zjednodušenie práce, zamedzenie zadávaniu chybných parametrov v programe Scrub-tcpdump, bolo v ďalšej časti práce vytvorené grafické rozhranie.

5 VYTVORENIE GRAFICKÉHO ROZHRANIA

Ako bolo spomenuté v predchádzajúcej kapitole, program Scrub-tcpdump pracuje pomocou zadávania príkazov v príkazovom riadku linuxových operačných systémoch. Parametre anoynomizácie sa zadávajú vo forme textových skratiek, čo komplikuje prácu s programom a zvyšuje pravdepodobnosť zadania chybného parametru. Program však zadaný chybný textový reťazec nedokáže identifikovať. Preto bolo vytvorené grafické prostredie programu Scrub-tcpdump v programovacom jazyku Python. V ňom bol použitý modul TKinter, ktorý umožňoval prácu s grafickými prvkami.

5.1 Štruktúra programu

Základnú štruktúru vzhľadu grafického rozhrania programu Scrub-tcpdump je možno vidieť na obr. 5.1. Hlavné okno je rozdelené na štyri časti.



Obr. 5.1: Vzhľad programu Scrub-tcpdump

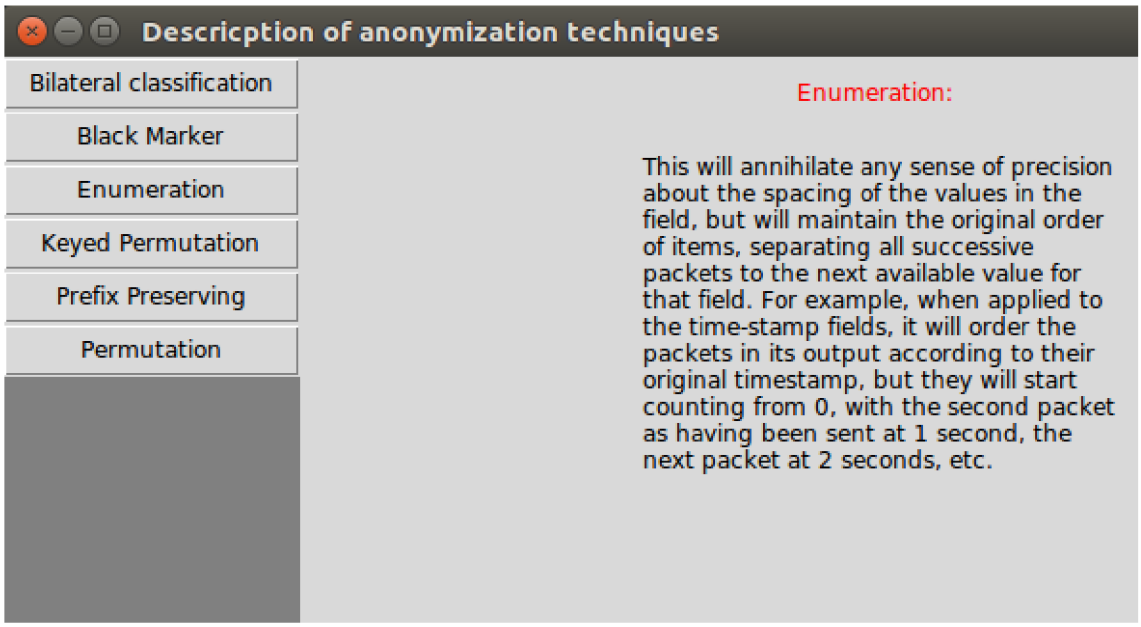
Ľavá časť okna obsahuje tlačítka, pomocou ktorých je možné prepínať medzi vrstvami „Network Layer“, „Transport Layer“ a vrstvou pcap poľa, kde možno nastaviť parametre anonymizácie pomocou tlačítka „Time Stamp“. Ďalším tlačítkom, „Choose Output“, sa zobrazí nastavenie, kde užívateľ zvolí názov výstupného adresára a názov výstupného súboru. Tlačítkom „Anonymize“ sa pri správne nastavených parametroch spustí proces anonymizácie. Posledným tlačítkom je „Clear Settings“, keď pri jeho stlačení sú vymazané všetky nastavené parametre.

V pravej časti okna je pri prepínaní vrstiev stále zobrazený panel pre zadanie anonymizačného kľúča, ktorý je pri technike „Keyed Permutation“ potrebný. Pod týmto panelom sa dynamický mení užívateľom zvolená vrstva. Na obr. 5.1 je zobrazená sieťová vrstva s poliami, ktoré je možné anonymizovať. Pod názvom každého poľa sú zobrazené techniky, ktorými možno dané pole modifikovať. Na každé pole je možné použiť vždy len jednu techniku. Po zaškrtnutí konkrétnej techniky, nie je možné použiť inú techniku pokiaľ je pôvodná technika zaznačená. Táto situácia je ošetrená následujúcim kódom:

```
def clickBMIPs():
        if bmIPsNet.get() == 1:
            global ipSoAdd
            ipSoAdd = 'srcip bm '
            permIPsNetB.configure(state = DISABLED)
            keyIPsNetB.configure(state = DISABLED)
            presIPsNetB.configure(state = DISABLED)
            print(ipSoAdd)

        if bmIPsNet.get()== 0:
            ipSoAdd = ''
            permIPsNetB.configure(state = ACTIVE)
            keyIPsNetB.configure(state = ACTIVE)
            presIPsNetB.configure(state = ACTIVE)
```

Tento kód je použitý pre zaškrtávacie tlačidlo techniky Black Marker poľa zdrojovej IP adresy. V tejto časti kódu je vytvorená premenná obsahujúca skratku poľa a techniky, ktorú Scrub-tcpdump neskôr bude prijímať ako parameter pre anonymizovanie. Podobné časti kódu sú vytvorené pre každé zaškrtávacie tlačidlo, určené k nastaveniu typu poľa a použitej techniky.

V hornej časti okna je umiestnené menu. V záložke „File“ je umiestnené tlačítko pre nahratie vstupného súboru, určeného k anonymizácii a tlačidlo pre ukončenie celého programu. Cez záložku „Help“ je možné otvoriť ďalšie okno, kde sú vysvetlené jednotlivé anonymizačné techniky. Náhľad okna je zobrazený na obr. 5.2.



Obr. 5.2: Okno popisujúce jednotlivé anonymizačné techniky

5.2 Spracovanie zvolených parametrov

Program Scrub-tcpdump potrebuje pre vykonanie procesu anonymizácie tieto parametre:

- cesta a názov vstupného súboru,
- cesta a názov výstupného súboru,
- konkrétne polia s anonymizačnou technikou v textovom reťazci,
- anonymizačný kľúč (technika „Keyed Randomize“).

Po kliknutí na tlačidlo pre vloženie vstupného súboru v záložke „File“, sa spustí funkcia s následujúcim kódom:

```
def openFile():
    global file
    file = str(askopenfilename(initialdir = "/",\
           title = "Select input file",\
           filetypes = (("pcap files","*.pcap"),\
           ("pcapng files","*.pcapng"))))
```

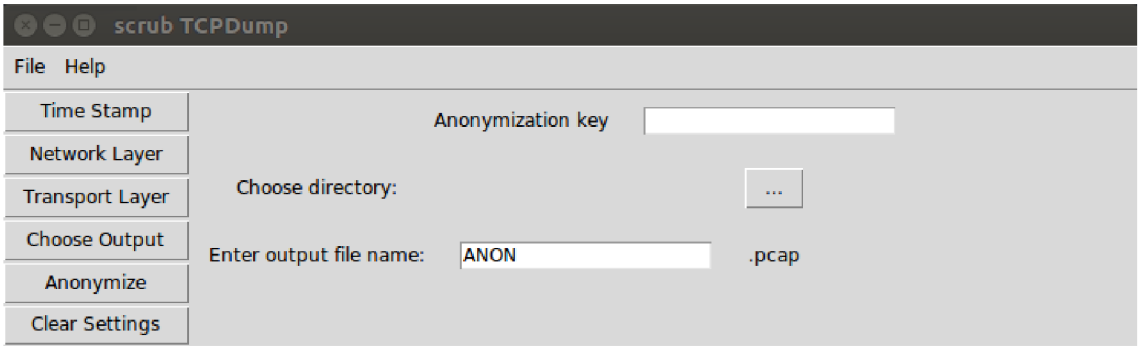
Pomocou tohto kódu sa užívateľ môže pohybovať v adresároch, kde môže zvoliť len súbory typu pcap, prípadne pcapng. Zvolený súbor aj s cestou bude uložený vo forme stringu do premennej pre ďalšie použitie.

Po kliknutí na tlačidlo „Choose Output“ sa zobrazí ponuka pre zvolenie adresára a názvu výstupného súboru ako je zobrazené na obr. 5.3. V prípade, že užívateľ

nezadá názov výstupného súboru, názov je defaultne nastavený na „ANON.pcap“. Po kliknutí na tlačidlo pre zvolenie výstupného adresára je spustená následujúca funkcia:

```
def saveFile():
        global outDirectory
        outDirectory = askdirectory()
```

V nej sa priradí cesta so zvoleným adresárom do premennej taktiež pre budúce použitie.



Obr. 5.3: Okno pre nastavenie uloženia výstupného súboru

V predchádzajúcej sekcii je zobrazený a popísaný kód, pomocou ktorého sa ukladajú parametre pre anonymizáciu konkrétnych polí do zvolenej premennej. Po zvolení všetkých potrebných parametrov, je možné po kliknutí na tlačidlo „Anonymize“ spustiť proces anonymizácie. Vtedy sa vykonáva kód z následujúcej metódy:

```
def runTCPDump(event):
    parameters = ' -o "'+tmStampPar+netFragFlagPar+ipSoAdd \
                 +ipDeAdd+ttlPar+pckLenPar+protNumPar \
                 +payPar+tcpSoPar+tcpDePar+udpSoPar \
                 +udpDePar+seqNumPar+tcpFlgPar+winSizePar+'"'
    inputfile = ' -r '+file
    outputfile = ' -w '+outDirectory+'/'+outputName+'.pcap'
    program = './scrub-tcpdump '
    arguments = program+inputfile+outputfile+parameters
    finish = subprocess.call(arguments, shell = True)
```

Do premennej „parameters“ je priradený znak „-o“, po ktorom Scrub-tcpdump očakáva textový reťazec so zvolenými poliami a technikami. Po znaku „-o“ nasledujú premenné všetkých polí nachádzajúce sa v jednotlivých vrstvách. Do ďalších premenných sú načítané vstupný a výstupný názov súboru a taktiež názov programu,

ktorý sa spustí ako subproces. Všetky spomenuté parametre sa uložia do jednej premennej a tá sa predá ako argument funkcii `subprocess.call()`, ktorá spustí samotný proces anonoymizácie.

5.3 Ošetrenie chybových stavov

Metóda `def runTCPDump(event)` obsahuje okrem kódu popísaného v predošlej sekcii ďalšie časti kódu, ktoré zabránia tomu, aby nastali udalosti, ktoré by môžu spôsobiť pád, poprípade chybu v procese anonymizácie. Metóda je ošetrená následujúcim kódom:

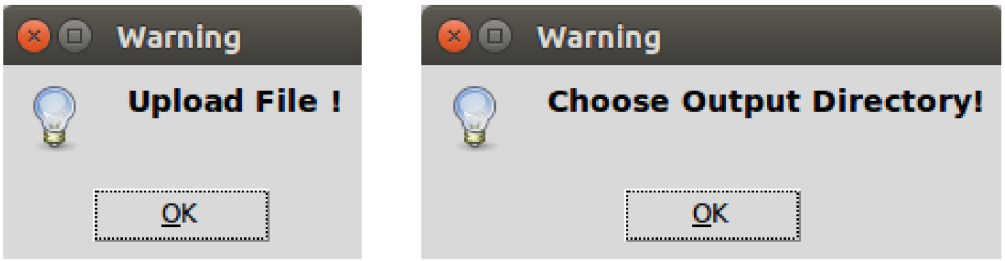
```
def runTCPDump(event):
    if file == '':
        messagebox.showinfo("Warning", "Upload File !")
    elif outDirectory == '':
        messagebox.showinfo("Warning", "Choose Output \
                            Directory!")
    else:
        if anonKey == '':
            ...
            if parameters == ' -o ""':
                messagebox.showinfo("Warning", "Choose "\
                 "Atleast One Anonymization technique!")
            else:
                if parameters.find('rp') == -1 :
                    finish = subprocess.call(arguments,       \
                                             shell = True)
                    if finish == 0:
                        messagebox.showinfo("Congratulation", \
                        "Anonymization is completed! Output"  \
                        " file is saved in " + outDirectory)
                    elif finish != 0:
                          messagebox.showinfo("Error",        \
                            "Too many chosen fields!")
                else:
                    messagebox.showinfo("Warning",            \
                            "Set the Anonymization Key !")
        else:
            ...
```

Bodky umiestnené v kóde sú substitúciou kódu, kde sa načítajú zvolené parametre. Ďalšie bodky na konci kódu obsahujú takmer identický kód uvedený v tomto výpise. Jednotlivé podmienky budú vysvetlené v následujúcej časti.

Hneď po stlačení tlačidla „Anonymize“ a zavolaní metódy `def runTCPDump()` je vykonaný kód, ktorý skontroluje, či je zvolený vstupný súbor a výstupný adresár. Kód vyzerá nasledovne:

```
def runTCPDump(event):
    ################ INPUT FILE IS NOT UPLOADED
    if file == '':
        messagebox.showinfo("Warning", "Upload File !")
    ################ OUTPUT DIRECTORY IS NOT UPLOADED
    elif outDirectory == '':
        messagebox.showinfo("Warning",\
                            "Choose Output Directory!")
    else:
        ...
```

Touto podmienkou sa skontrolujú premenné, ktoré majú obsahovať zvolený súbor alebo adresár. Ak je jedna z premenných prázdna, metóda sa ukončí a nespustí sa proces anonymizácie. Užívateľ je o nezvolení jedného z týchto parametrov upozornený vyskakovacími oknami zobrazenými na obr. 5.4. V kóde za `else:` nasleduje kód pre načítanie zvolených parametrov a spustenie procesu anonymizácie, ktorý je uvedený vyššie.



Obr. 5.4: Vyskakovacie okná pri nezvolení vstupu/výstupu

Ošetrená je taktiež situácia, kedy užívateľ nezvolí ani jednu z anonymizačných techník pre konkrétne pole. V tomto prípade by nastal pád procesu, preto bolo potrebné túto situáciu ošetriť uvedeným kódom:

```
if parameters == ' -o ""':
    messagebox.showinfo("Warning", "Choose "\
     "Atleast One Anonymization technique!")
```

V prípade, kedy sa za „-o“ nachádzajú len prázdne úvodzovky, užívateľ je vyzvaný na zvolenie aspoň jednej anonymizačnej techniky, aby anonymizácia mohla nastať.

Celá metóda `def runTCPDump(event)` je rozdelená podmienkou na dve vetvy. Prvá vetva je zobrazená vo výpise uvedenom vyššie. Tá sa vykonáva vtedy, keď nie zadaný anonymizačný kľúč. Ako bolo už spomenuté, ten je potrebný len pri využití techniky „Keyed Permutation“. Preto je potrebné, aby bol pri zvolení tejto techniky užívateľ upozornený, ak chce spustiť anonymizáciu bez zadania kľúča. To zabezpečuje následujúca časť kódu:

```
if parameters.find('rp') == -1 :
    finish = subprocess.call(arguments, \
             shell = True)
    .
    .
    .
else:
    messagebox.showinfo("Warning",              \
            "Set the Anonymization Key !")
```

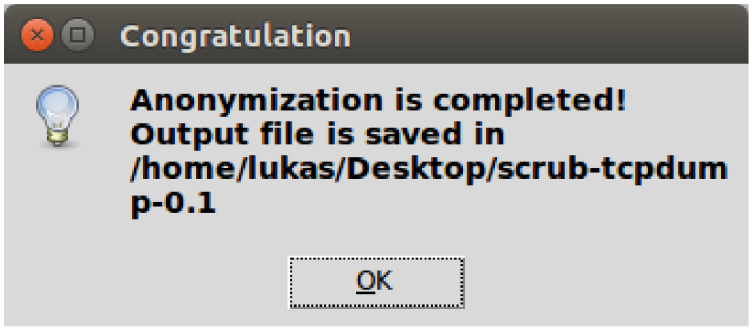
Premenná „parameters“ typu string je prehľadaná funkciou `parameters.find()` a pokiaľ sa v nej nenachádza reťazec „rp“, je proces spustený. Reťazec „rp“ znamená, že medzi zvolenými technikami sa nachádza technika „Keyed Permutation“. V tom prípade je užívateľ vyzvaný na vloženie anonymizačného kľúča.

Program Scrub-tcpdump, dokáže pracovať len s určitým množstvom zvolených parametrov. Pri preťažení nastáva pád procesu. Preto je táto situácia ošetrená nasledujúcou časťou kódu:

```
finish = subprocess.call(arguments,         \
         shell = True)

if finish == 0:
    messagebox.showinfo("Congratulation", \
     "Anonymization is completed! Output"  \
     " file is saved in " + outDirectory)
elif finish != 0:
    messagebox.showinfo("Error",          \
    "Too many chosen fields!")
```

V prípade úspešného ukončenia procesu anonymizácie, je do premennej „finish“ priradená nula a užívateľ je o úspešnosti procesu informovaný oknom, kde je taktiež uvedený názov adresára s výstupným anonymizovaným súborom, ako je zobrazené na

obr. 5.5. V prípade chyby spôsobenej príliš veľa parametrami, je užívateľ upozornený, že zvolil príliš veľa parametrov.



Obr. 5.5: Informácia o úspešnej anonymizácií

Druhá vetva metódy `def runTCPDump(event)` obsahuje takmer identický kód s tým rozdielom, že sa vykonáva v prípade zadania anonymizačného kľúča. Ak je kľúč zadaný, ale technika „Keyed Permutation“ nebola vybraná, proces anonymizácie je vykonaný. Užívateľ je pomocou vyskakovacieho okna len informovaný, že kľúč nemusel zadať.

6 ZÁVER

Táto práca sa zaoberá anonymizáciou užívateľov pri zbere dát v sieťovej prevádzke. Jej cieľom nájsť riešenie, ktoré zamedzí odhaleniu identity užívateľov pri analýze a vytváraní štatistík zo sieťovej prevádzky.

V teoretickej časti boli popísané nástroje, ktoré sa zaoberajú monitorovaním siete a taktiež zachytávaním sieťovej prevádzky. Ďalej sú v teoretickej časti analyzované nariadenia, ktoré sa týkajú ochrany osobných údajov koncových užívateľov v sieti. Ďalej sú v práci popísané anonymizačné techniky, ktoré sú používané viacerými nástrojmi slúžiacimi k anonymizácii sieťovej prevádzky.

Praktická časť predstavuje vytvorenie virtuálnej laboratórnej siete pomocou nástroja VMware hypervisor ESXi. Týmto nástrojom je virtualizovaný fyzický server, na ktorom sú vytvorené tri virtuálne stroje s nainštalovaným operačným systémom Linux Ubuntu. Vo virtuálnej sieti sú na jednom počítači nainštalované nástroje slúžiace ku generovaniu paketov, ktoré sú posielané na druhý počítač reprezentujúci webový server. Na ňom je sieťová prevádzka zachytávaná a ukladaná do súboru. Z webového servera bola sieťová prevádzka pretváraná a emitovaná do tretieho počítača s dátami vo formáte NetFlow. V treťom počítači je nainštalovaný nástroj určený pre ukladanie a zobrazovanie NetFlow dát.

V ďalšej časti práce sú testované anonymizačné programy uvedené v teoretickej časti práce, v snahe nájsť čo najlepšie riešenie pre ochranu identity. Programy sú testované na sieťovej prevádzke o rôznej veľkosti. Sieťová prevádzka je získaná zachytávaním komunikácie vo virtuálnej sieti. Testovaných bolo osem programov, z ktorých tri neboli z viacerých dôvodov funkčné. Zvyšných päť programov dokázalo anonymizovať zdrojovú a cieľovú IP adresu protokolu IPv4. Okrem programu TraceWrangler, nedokážu programy modifikovať IP adresy v protokole ARP a taktiež nedokážu anonymizovať protokoly aplikačnej vrstvy.

V poslednej časti práce je vytvorené grafické prostredie pre program Scrub-tcpdump, kvôli jeho širokej variabilite použitia anonymizačných techník na polia hlavičiek protokolov Ipv4, TCP a UDP. Pretože parametre anonymizácie sú v príkazovom riadku zadávané vo forme skratiek do spoločného textového reťazca, grafické prostredie zjednodušilo prácu s týmto programom. Program pri práci v príkazovom riadku taktiež neupozorňuje na chybne zadané parametre anonymizácie, čo je možné zistiť až po analýze výstupného súboru. Vytvorením grafického prostredia sú možné stavy, kedy by došlo ku chybnému procesu anonymizácie, ošetrené.

Záverom možno konštatovať, že testovaním anonymizačných programov boli zistené vlastnosti jednotlivých nástrojov. V závislosti na požiadavkách anonymizácie a typu sieťovej prevádzky, je možné vybrať konkrétny program a popísaný postup a tým zaistiť získanie anonymných dát pre tvorbu štatistík.

LITERATÚRA

[1] IRIS Network Systems. *Active versus Passive Network Monitoring: An infographic guide, revisited* [online]. 2016 [cit. 2017-11-22]. Dostupné z: <https://goo.gl/ws1E26>.

[2] IRIS Network Systems. *Active vs. Passive network monitoring: An Infographic Guide* [online]. 2016 [cit. 2017-11-22]. Dostupné z: <https://www.irisns.com/wp-content/uploads/2016/04/TIMELINE_3.png>.

[3] SAMURAJ. *SNMP - Simple Network Management Protocol* [online]. 2016 [cit. 2017-11-22]. Dostupné z: <https://www.samuraj-cz.com/clanek/snmp-simple-network-management-protocol/>.

[4] BEZPALEC, Pavel. *Nové trendy v elektronických komunikacích Management ICT systémů: Protokoly používané pro dohled nad datovou sítí* [online]. [cit. 2017-11-22]. Dostupné z: <https://publi.cz/books/242/06.html>.

[5] CISCO. *Introduction to Cisco IOS NetFlow - A Technical Overview* [online]. 2012 [cit. 2017-11-22]. Dostupné z: <https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html>.

[6] Flowmon. *NETFLOW, NOVÁ ÉRA MONITOROVÁNÍ POČÍTAČOVÝCH SÍTÍ* [online]. [cit. 2017-11-22]. Dostupné z: <https://www.flowmon.com/cs/solutions/use-case/netflow-ipfix>.

[7] PC & Network DOWNLOADS. *What is IPFIX – The Protocol that's giving Netflow Analyzing a Run for its Money!* [online]. 2016 [cit. 2017-11-22]. Dostupné z: <https://www.pcwdld.com/what-is-ipfix>.

[8] About Wireshark. WIRESHARK [online]. 2017 [cit. 2017-11-22]. Dostupné z: <https://www.wireshark.org/>.

[9] TCPDUMP & LIPCAP. *TCPDUMP* [online]. 2017 [cit. 2017-11-22]. Dostupné z: <http://www.tcpdump.org/tcpdump_man.html>.

[10] Epravo. *Jaké budou dopady zrušení směrnice o data retention?* [online]. 2014 [cit. 2017-11-23]. Dostupné z: <https://www.epravo.cz/top/clanky/jake-budou-dopady-zruseni-smernice-o-data-retention-94415.html>.

[11] Czech Free Press. *Plošný sběr dat je v Evropské unii ilegální, rozhodl Evropský soudní dvůr. Legislativu má změnit Švédsko, Velká Británie i Česká republika* [online]. 2016 [cit. 2017-11-23]. Dostupné z: <https://goo.gl/3roXTW>.

[12] GDPR. *Co je GDPR a jak bude aplikováno v Česku* [online]. 2016 [cit. 2017-11-23]. Dostupné z: <https://www.gdpr.cz/gdpr/co-je-gdpr/>.

[13] GDPR. *Jaké povinnosti ukládá GDPR institucím a firmám* [online]. 2016 [cit. 2017-11-23]. Dostupné z: <https://www.gdpr.cz/gdpr/povinnosti/>.

[14] Základy IT gramotnosti. *Anonymizace* [online]. 2014 [cit. 2017-11-23]. Dostupné z: <https://is.muni.cz/do/1492/el/sitmu/law/html/ch02s10.html>.

[15] FARAH, Tanjila a Ljiljana TRAJKOVIC. Anonym: A tool for anonymization of the Internet traffic. *2013 IEEE International Conference on Cybernetics (CYBCO)* [online]. IEEE, 2013, , 261-266 [cit. 2017-11-23]. DOI: 10.1109/CYBConf.2013.6617434. ISBN 978-1-4673-6469-0. Dostupné z: <http://ieeexplore.ieee.org/document/6617434/>.

[16] TraceWrangler. *TraceWrangler - Packet Capture Toolkit* [online]. 2017 [cit. 2017-11-28]. Dostupné z: <https://www.tracewrangler.com/>.

[17] BONGERTZ, Jasper. *Trace File Sanitization NG*. In: SHARKFEST 13 [online]. 2013 [cit. 2017-11-28]. Dostupné z: <https://sharkfest.wireshark.org/sharkfest.13/presentations/SEC-04_Trace-File-Sanitization-NG_Jasper-Bongertz.pdf>.

[18] YURCIK, William, Clay WOOLAM, Greg HELLINGS, Latifur KHAN a Bhavani THURAISINGHAM. SCRUB-tcpdump: A multi-level packet anonymizer demonstrating privacy/analysis tradeoffs. *2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops - SecureComm 2007* [online]. IEEE, 2007, , 49-56 [cit. 2017-11-28]. DOI: 10.1109/SECCOM.2007.4550306. ISBN 978-1-4244-0974-7. Dostupné z: <http://ieeexplore.ieee.org/document/4550306/>.

[19] PCAP obfuscator. *GitHub* [online]. 2018 [cit. 2018-04-08]. Dostupné z:<https://github.com/jorgeborreicho/pcap_obfuscator>.

[20] Capsan. *GitHub* [online]. 2014 [cit. 2018-04-08]. Dostupné z: <https://github.com/jsiwek/capsan>.

[21] Sourceforge. *NFDUMP* [online]. 2014 [cit. 2017-11-29]. Dostupné z: <http://nfdump.sourceforge.net./>.

[22] FOUKARAKIS, Michalis, Demetres ANTONIADES, Spiros ANTONATOS a Evangelos P. MARKATOS. Flexible and high-performance anonymization of NetFlow records using anontool. *2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops - SecureComm* 2007 [online]. IEEE, 2007, , 33-38 [cit. 2017-11-29]. DOI: 10.1109/SECCOM.2007.4550304. ISBN 978-1-4244-0974-7. Dostupné z: <http://ieeexplore.ieee.org/document/4550304/>.

[23] SLAGELL, Adam, Kiran LAKKARAJU a Katherine LUO. FLAIM: A Multi-level Anonymization Framework for Computer and Network Logs. *LISA* [online]. 2006, 3-8 [cit. 2017-11-29]. Dostupné z: <https://www.usenix.net/legacy/events/lisa06/tech/full_papers/slagell/slagell.pdf>.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

QoS	Quality of Services
SNMP	Simple Network Management Protocol
UDP	User Datagram Protocol
NMS	Network Management System
IETF	Internet Engineering Task Force
SGMP	Simple Gateway Monitoring Protocol
MIB	Management Information Base
OID	Object Identifier
IP	Internet Protocol
SCTP	Stream Control Transmission Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
MAC	Media Access Control
VLAN	Virtual Local Area Network
IPFIX	IP Flow Information Export
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
PPP	Point-to-Point Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
GDPR	General Data Protection Regulation
ISP	Internet Service Provider
AS	Autonomous System
GUI	Graphical User Interface
ARP	Address Resolution Protocol
ICMPv4	Internet Control Message Protocol version 4
AS	Autonomous System
ASCII	American Standard Code for Information Interchange
AAPI	Anonymization Application Programming Interface
RAM	Random Access Memory

ZOZNAM PRÍLOH

A OBSAH PRILOŽENÉHO CD

Priložené CD obsahuje adresár `scrub-tcpdump-0.1`, ktorý obsahuje program Scrub-tcpdump aj s vytvoreným grafickým prostredím. Adresár taktiež obsahuje testovaciu sieťovú prevádzku `NetworkTrafficRepair.pcap`. Pre spustenie programu `GUI-scrub.py` je najskôr potrebné nainštalovať balíček `python3-tk`. Potom je možno program spustiť v príkazovom riadku príkazom `python3 GUI-scrub.py`. Avšak je potrebné nachádzať sa v adresári `scrub-tcpdump-0.1`.

Na priloženom CD sa taktiež nachádza elektronická verzia tejto práce.

```
/ ..... koreňový adresár priloženého CD
├── scrub-tcpdump-0.1 ..... adresár programu Scrub-tcpdump
│   ├── GUI-scrub.py ..... grafické prostredie programu
│   └── NetworkTrafficRepair.pcap ..... testovací súbor
└── BakalarskaPraca ..... elektronická kópia tejto stránky
```