VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCIA SEEDBOXOV V SIETI BITTORRENT

DETECTION OF SEEDBOXES IN BITTORRENT NETWORK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN GRNÁČ
AUTHOR

VEDOUCÍ PRÁCE — Ing. LIBOR POLČÁK, Ph.D.
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/20465/2017/xgrnac00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Grnáč Martin**
Obor: Informační technologie
Téma: **Detekce seedboxů v síti BitTorrent**
BitTorrent Seedbox Detection
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se s protokolem BitTorrent.
2. Nastudujte možnosti detekce seedboxů.
3. Po konzultaci s vedoucím práce navrhněte metodu detekce seedboxů.
4. Návrh implementujte.
5. Implementaci otestujte v laboratoři i v prostředí Internetu.
6. Navrhněte možná rozšíření projektu.

Literatura:

- Dario Rossi, Guilhem Pujol, Xiao Wang, Fabien Mathieu, 2014. Peeking through the BitTorrent Seedbox Hosting Ecosystem. In: Traffic Monitoring and Analysis. TMA 2014. Lecture Notes in Computer Science, vol 8406. Springer, Berlin, Heidelberg.
- Stefan Schindler. Analysis of BitTorrent Trackers and Peers. Bakalářská práce, 2015 Friedrich-Alexander-Universität, Erlangen-Nürnberg.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Polčák Libor, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

L.S.

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Bakalárska práca sa venuje problematike sledovania a detekcie seedboxov v sieti BitTorrent za pomoci technológie netflow. V teoretickej časti je predstavená a popísaná architektúra P2P, základy a kľúčové pojmy architektúry BitTorrent a teoretická definícia seedboxu. Taktiež sú tu rozobrané metódy pomocou ktorých sa dá detekovať sieťová komunikácia a ďalej je uvedená analýza seedboxov v sieti a hľadanie ich charakteristík. Na základe týchto znalostí a sledovaní je navrhnutá sada nástrojov, ktoré napomáhajú ich detekcií. V praktickej časti je predstavená implementácia týchto nástrojov a výsledky ich testovania.

Abstract

Bachelor's thesis is focused on issues with monitoring and detection of seedboxes in BitTorrent network with help of netflow technology. In the theoretical part of this thesis is introduced and described P2P architecture, basics and key terms of BitTorrent architecture and theoretical definition of seedbox. There are also described specific methods which can be used for detection of network communication and next there is described process of seedbox analysis in network and process of finding its characteristics. On base of this knowledge and observations is designed a set of tools,which help with detection of seedboxes. In the practical part of this work is presented implementation of these tools and results of testing these tools.

Kľúčové slová

BitTorrent, seedbox, detekcia, netflow, analýza, sieťová prevádzka, P2P, netflow

Keywords

BitTorrent, seedbox, detection, netflow, analysis, network traffic, P2P, netflow

Citácia

GRNÁČ, Martin. *DETEKCIA SEEDBOXOV V SIETI BITTORRENT*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Libor Polčák, Ph.D.

DETEKCIA SEEDBOXOV V SIETI BITTORRENT

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Libora Polčáka,Ph.D. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Martin Grnáč
16. mája 2018

Poďakovanie

Týmto by som chcel poďakovať pánovi Ing. Liborovi Polčákovi, Ph.D. za jeho cenné rady a odbornú pomoc pri tvorbe tejto bakalárskej práce.

Obsah

Kapitola 1

Úvod

Peer-to-peer (P2P) siete majú v dnešnom prostredí internetu a sieťových aplikácií neustále narastajúcu popularitu a využitie. Najväčšie zastúpenie v používaní tohto princípu majú aplikácie na zdieľanie súborov, ale uplatnenie môžeme nájsť aj v hlasových službách - Peer-to-peer SIP (P2P-SIP) alebo pri prenose multimediálneho obsahu - Peer-to-peer TV (P2PTV) a za posledné obdobie narastá aj počet decentralizovaných kryptomien fungujúcich na tomto princípe. Charakteristické chovanie P2P sietí je vytváranie veľkého množstva krátkych spojení a keďže s narastajúcim počtom nových užívateľov v sieti sa týmto rýchlo zaberá aj prenosové pásmo, je potreba tento typ sieťovej premávky vedieť detekovať a následne regulovať.

BitTorrent je jedným z najpoužívanejších protokolov na zdieľanie súborov. V rámci sietí, v ktorých sa nachádzajú uzly komunikujúce týmto protokolom, sa môžu vyskytovať zariadenia ktoré sa podieľajú na zdieľaní viac súborov naraz - seedbox. Takéto zariadenia môžu vyťažovať sieť niekoľko násobne viac ako klasický bittorrentový uzol. Keďže tieto zariadenia pri komunikácií produkujú veľkú časť internetovej premávky, môže v danej sieti dôjsť k jej zahlteniu. V rámci práce sieťového administrátora by bolo vhodné, aby vedel tieto zariadenia v sieti nájsť, následne zasiahnuť a upraviť nastavenia a štruktúru siete tak, aby bolo využitie prenosového pásma optimálne.

Predmetom tejto práce, je analyzovanie komunikácie seedbox zariadení v sieti, pri ktorej by sme mali zistiť, ako sa tieto zariadenia v sieti správajú a nájsť charakteristické znaky, ktorými sa pri komunikácií vyznačujú. Pomocou zistených poznatkov a nájdených znakov je navrhnutá sada nástrojov, ktoré pomáhajú k detekcií seedboxov v danej sieti.

V druhej kapitole bude popísaná problematika P2P sietí a protokolu BitTorrent. Taktiež je v tejto kapitole popísané, čo je to seedbox a aké typy seedboxov existujú. V tretej kapitole sú uvedené metódy, ktoré sa využívajú pre detekciu sieťovej komunikácie a môžu byť využité na rozpoznanie a detekovanie komunikácie P2P. Taktiež táto kapitola uvádza nástroje, ktoré boli použité vrámci tejto práce pre analýzu komunikácie BitTorrent, ktorou sa seedboxy vyznačujú. V štvrtej kapitole sú analyzované charakteristiky seedboxov. Keďže seedboxy komunikujú pomocou protokolu BitTorrent, ktorá má určité charakteristiky, budú sa tieto charakteristiky vyskytovať aj na sledovaných seedboxoch. V piatej kapitole je popísaná implementácia nástroja, ktorý bol počas tejto práce vytvorený a ktorý napomáha k detekcii seedboxov. V šiestej kapitole je uvedené testovanie tohoto nástroja. Siedma kapitola zhŕňa celkový proces tejto práce, jej výsledky a ďalšie možné smerovanie tejto práce.

Kapitola 2

Peer-to-peer siete a protokol BitTorrent

Pre pochopenie akým spôsobom seedbox komunikuje a ako jeho komunikácia prebieha, je dôležité porozumieť architektúre a protokolom na základe ktorých je táto komunikácia postavená. Preto v tejto kapitole bude popísaná architektúra P2P sietí, následne bude rozobraný protokol BitTorrent a súčasti jeho architektúry a ku koncu kapitoly bude popísané zariadenie seedbox.

2.1 P2P siete a ich architektúra

Najznámejšia sieťová architektúra, ktorú poznáme je architektúra klient-server. Komunikácia v tomto type architektúry je založená na princípe dotaz a odpoveď, kedy klient posiela dotaz serveru zaobalený vo popred špecifikovanom protokole, ktorému obe strany rozumejú. Po prijatí dotazu server spracuje odoslanú požiadavku a na základe tohto vykoná príslušné akcie a následne odošle klientovi odpoveď. Model klient-server využívajú protokoly ako sú HTTP alebo Telnet. Nespornou nevýhodou tejto architektúry je preťažovanie siete. Obzvlášť pri zvyšujúcom sa počte klientov, ktorí sa súbežne na server dotazujú, narastá taktiež vyťaženie tohto servera. Následkom tohto môže dôjsť k vyčerpaniu jeho zdrojov a ďalší klienti sa potom už na túto službu nedostanú [17].

Alternatívnou architektúrou k modelu klient-server je takzvaná P2P architektúra. V tejto architektúre prebieha komunikácia medzi rovnocennými uzlami. Takýto uzol nazývame peer. Peer sa počas komunikácie správa ako klient a zároveň aj ako server a teda pri komunikácií dvoch peerov si každý peer zabezpečuje súčasne klientskú aj serverovú časť.

Čistá P2P architektúra nepozná prvok akým je server. V praxi sa ale objavujú špecializované servery, ktoré zabezpečujú počiatočné naviazanie komunikácie medzi dvoma peermi (napríklad v prípade BitTorrentu). Základný rozdiel v štruktúrach oboch týchto sieťových architektúr môžeme vidieť na obrázku 2.1.

Komunikáciu medzi dvoma peermi môžeme rozdeliť do týchto fáz:

- Signalizácia - Táto fáza zahŕňa vyhľadávanie a vytváranie spojenia medzi peermi. Počas tejto fázy peer zisťuje, ktorý peeri v sieti majú požadované dáta.
- Prenos dát - V tejto fáze dochádza ku kontaktovaniu a zahájeniu prenosu dát od konkrétne nájdeného peera [13].

Protokoly rôznych P2P aplikácií majú niekoľko spoločných rysov. Prvým rysom je to, že sú vytvorené nad aplikačnou vrstvou. Druhým rysom je to, že peeri majú v rámci siete unikátny identifikátor ako napr. peer ID alebo unikátna adresa peera. Tretím spoločným znakom je podobnosť typov zasielaných správ v rôznych P2P protokoloch. Posledným znakom je to, že P2P protokol podporuje určitý typ smerovania a teda správa môže byť zaslaná k cieľovému peerovi cez niekoľko medzi-peerov, ktorí ležia na ceste medzi týmito dvoma komunikujúcimi peermi [2].

Záujem verejnosti o P2P siete výrazne vzrástol po objavení systémov pre zdieľanie dát. Jednou z takýchto prvých P2P sietí bola Napster. Táto sieť slúžila na zdieľanie digitálnych audio súborov. Hlasová služba Skype bola taktiež pôvodne založená na princípe P2P, neskôr však prešla na centralizovaný princíp. V súčasnosti niektoré organizácie a spoločnosti poskytujúce služby po sieti využívajúce P2P pre aktualizáciu software svojím klientom (napriklad hra World of Tanks alebo Microsoft Windows) [20, 2]. Tento spôsob aktualizácie je výhodný pre obidve strany, pretože poskytovatelia služieb ušetria na prenosovom pásme a klienti zase budú mať zabezpečenú rýchlu dostupnosť dát.

So vzrastajúcou popularitou P2P aplikácií rastie aj ich dopad na výkon siete. Odhadované zastúpenie P2P premávky v sieťach veľkých poskytovateľov internetu je až 50% [2]. Je pravdepodobné, že do budúcna bude toto číslo naďalej narastať vzhľadom na popularizáciu a vytváraní nových decentralizovaných kryptomien a nasadzovaniu tohto princípu pri aktualizácií software. S týmto narastaním sa však taktiež objavuje problém u poskytovateľov internetu so zahlcovaním siete veľkým množstvom dát, ktoré P2P aplikácie vyprodukujú. Druhý problém je taktiež fakt, že siete poskytovateľov internetu nie sú optimálne navrhnuté, aby zvládali takéto P2P premávky. Toto je spôsobené predovšetkým tým, že sa peer správa súčasne ako klient aj ako server a komunikácia je symetrická, ale štruktúra sietí providerov je nastavená tak, že sťahovacia kapacita (downstream) v nej je minimálne päť krát tak väčšia ako kapacita nahrávacia (upstream) [2]. Sieť providera je z tohto dôvodu viac zahltená nahrávacou premávkou, ktorá je produkovaná P2P aplikáciami.

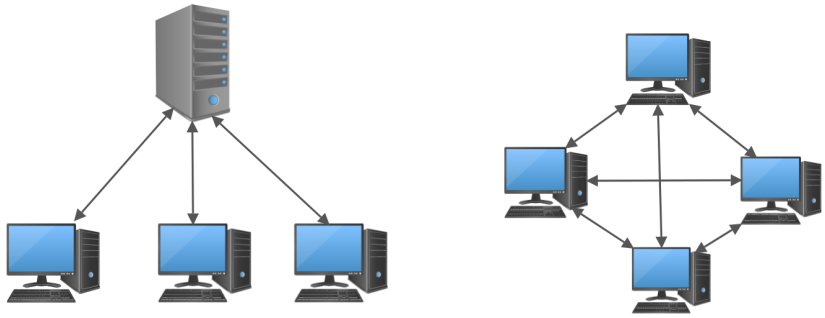
Často spomínaná je aj otázka legálnosti P2P sietí. Samotná myšlienka zdieľania súborov týmto spôsobom nelegálna nie je. Problém však nastáva v prípade, ak si užívatelia začnú navzájom zdielať materiál, ktorý podlieha autorským zákonom ako sú napr. filmy, hudba alebo komerčný software, následkom čoho môže byť začatie trestného stíhania užívateľov, ktorí sa na zdieľaní takéhoto materiálu podieľali [9].

Výhodou P2P sietí je predovšetkým to, že s narastajúcim počtom užívateľov rastie aj celková dostupná prenosová kapacita siete na rozdiel od modelu klient-server, kde pri náraste klientov klesá priemerná prenosová rýchlosť, pretože sa užívajúci klienti musia o server deliť. Nevýhodu tohto typu komunikácie predstavuje bezpečnostné riziko, ktoré je spôsobené v neznalosti identity komunikujúceho uzla a neznalosti toho, či obsah sťahovaného súboru je ten, ktorý sme si vyžiadali (napr. možnosť stiahnutia si škodlivého kódu) [15].

2.2 Protokol BitTorrent

Protokol BitTorrent bol vytvorený za účelom zdieľania dát, kedy je počas procesu zdieľania možné získať požadovaný súbor za relatívne krátky čas. Zdieľanie dát prebieha medzi veľkým počtom uzlov, pričom zaťaženie uzla, ktorý má k dispozícií kompletný zdroj dát je minimálne. Toto je dosiahnuté tým, že súbor je rozdelený na niekoľko menších častí a ďalej distribuovaný medzi jednotlivé uzly v sieti, ktoré sa podieľajú na sťahovaní daného súboru.

Predpokladá sa že komunikácia protokolom BitTorrent tvorí až 98% zo všetkej P2P premávky [14].



Obr. 2.1: Porovnanie architektúr klient-server (vpravo) a P2P (vľavo).

Zakladateľom tohto protokolu je Bram Cohen a jeho referenčná implementácia bola napísaná v jazyku Python. Pre začatie procesu sťahovania v sieti BitTorrent sú potrebné tieto tri časti:

- **Torrent súbor**, ktorý má v sebe potrebné metadáta o súbore alebo skupine súborov. Tieto súbory sú rozdistribuované na webových serveroch - indexoch.
- **Tracker server** od ktorého peeri získavajú IP adresy a čísla naslúchajúcich portov ostatných peerov.
- **Peer Wire Protokol** pomocou ktorého prebieha komunikácia medzi dvoma peermi.

Z hľadiska stavu zdieľania súboru delíme peerov v sieti BitTorrent na dva typy:

- **Leecher**, je označenie peera v torrentovej sieti, ktorý zatiaľ nemá kompletnú kópiu dát a stále ich sťahuje. Ak získa nejakú časť sťahovaného súboru tak ju môže začať zdieľať do siete ostatným peerom.
- **Seeder**, je označenie peera, ktorý má dostupnú celú kópiu sťahovaného súboru a už ho len zdieľa do siete ostatným peerom. Pokiaľ leecher ostane v sieti ďalej dostupný aj po dokončení sťahovania daného súboru od ostatných peerov, stáva sa z neho tiež seeder. V rámci siete BitTorrent môžu byť dva typy seederov. Prvý typ seedera sa nazýva *počiatočný seeder*, ktorý ako prvý začal zdieľanie daného torrentu do swarmu. Druhým typom je seeder, ktorý obsah získal stiahnutím buď od počiatočného seedera alebo ostatných seederov a leecherov v danom swarme.

Pre každý vytvorený torrent existuje zoskupenie peerov, ktorý sa v určitý časový interval na zdieľaní daného torrentu podieľajú. Takáto skupina sa nazýva **swarm**. Aby určitý swarm mohol existovať je potrebné, aby sa v danom swarme vyskytoval aspoň jeden seeder, ktorý môže odosielať dáta ostatným peerom, ktorý sa do swarmu zapoja.

Každý peer v swarme má počas procesu zdieľania určitý pomer v dátach, ktoré v rámci swarmu odoslal a dátach, ktoré prijal. Tento pomer je v BitTorrent terminológií označovaný ako **ratio**. Užívateľ by mal dodržiavať minimálne ratio = 1.0 a teda odoslať do siete rovnaký počet dát ako prijal.

2.2.1 Torrent súbor

Obsah zdieľaný v sieti BitTorrent môže byť samostatný súbor alebo adresár, ktorý obsahuje podadresáre s viacerými súbormi. Informácie o tomto súbore alebo množine súborov, ktorá sa bude sťahovať je zapísana v špeciálnom type súboru. Tento súbor má príponu *.torrent* a nesie v sebe metadáta o danom torrente.

Celý obsah súboru je reprezentovaný v špeciálnom kódovaní - *bencoding*. Bencoding navrhol samotný Bram Cohen a jeho účelom je efektívnejší prenos dát po sieti. Hodnoty v tomto type kódovania sú umiestnené do slovníkových štruktúr. Klasický torrent súbor má v sebe slovník, v ktorom sú dva hlavné kľúče - announce a info.

Kľúč *announce* v sebe obsahuje URL trackeru, ktorý treba zodpovedajúco nakontaktovať pre zahájenie sťahovania. Vo veľkej časti prípadov sa v torrent súboroch vyskytuje aj rozšírenie multitracker, kedy je v slovníku zakomponovaný aj zoznam iných trackerov, ktorý daný torrent spravujú. Tento zoznam je v torrentovom súbore uchovávaný pod názvom *announce-list*.

Kľúč *info* v sebe obsahuje slovník informácií o zdieľanom súbore prípadne zdieľaných súboroch. Hodnoty v slovníku, ktorý je uložený pod kľúčom *info* sú - *name, piece length, pieces, length, files*. Hodnota *name* popisuje meno sťahovaného súboru, prípadne sťahovaného adresára. *Piece length* popisuje veľkosť jedného bloku sťahovaného súboru. *Pieces* obsahuje konkatenáciu jednotlivých SHA-1 hash hodnôt každého bloku súboru. Počet všetkých blokov na ktorý bol daný súbor rozdelený, môžeme zistiť predelením dĺžky tohto reťazca hodnotou 20, pretože hashe sú vytvárané hashovaciou funkciou SHA-1, ktorá vytvára hashe o dĺžke 20 Bytov. *Length* obsahuje celkovú veľkosť súboru. *Files* je zoznam slovníkov, ktorý obsahuje informácie o jednotlivých súboroch v adresári. Tento slovník je prítomný, ak je torrent zložený z viacerých súborov. V zozname *files* sú hodnoty *lenght* a *path*. *Path* v sebe obsahuje cestu k súboru v sťahovanom adresári [4].

2.2.2 Tracker

Jedným z hlavných problémov siete BitTorrent je zistiť kontaktné informácie ostatných uzlov v sieti. Klasickým riešením býva použitie servera, ktorému peeri oznamujú pripojenie sa do daného torrentového swarmu. Takýto server sa nazýva tracker.

Peer, ktorý chce sťahovať daný torrent nakontaktuje tracker pomocou protokolu HTTP, HTTPS alebo UDP. Tracker mu ako odpoveď odošle zoznam peerov pre daný torrent. Zaznamenanie toho, ktorým protokolom tracker komunikuje, je zapísané v jeho URL, ktoré je zakódované v torrent súbore. Trackre pre komunikáciu využívajú najčastejšie port 6969. Zo štúdie, ktorú som vykonal na zložke 1230 torrentov a ktorej výsledok je zobrazený v tabuľke 2.1 si môžeme všimnúť, že veľké zastúpenie majú aj porty 2710, 80 a 1337.

Ďalej som v svojej štúdií zistil, že pri sledovaní všetkých protokolov, ktoré trackery v analyzovanej zložke používajú pre naviazanie spojenia má až 82% zastúpenie protokol HTTP. Tieto výsledky sú zobrazené v tabulke 2.2.

Peer odosiela na tracker správu typu announce request. Ukážka tohto typu správy je na obrázku 2.2. Táto správa obsahuje následujúce položky:

- `info_hash` - Obsahom tohto parametru je SHA-1 hash, ktorý je získaný zahashovaním slovníkovej štruktúry *info*, ktorá sa nachádza v torrent súbore.
- `peer_id` - 20 bajtový reťazec, ktorý si volia peeri samostatne a používajú ho ako svoj identifikátor počas komunikácie.

Číslo portu	Počet výskytov
2710	973
6969	747
80	201
1337	127
443	40
451	36
8082	23

Tabuľka 2.1: Zobrazenie počtu výskytov jednotlivých portov ktoré trackery používajú v analyzovanej zložke s torrentovými súbormi.

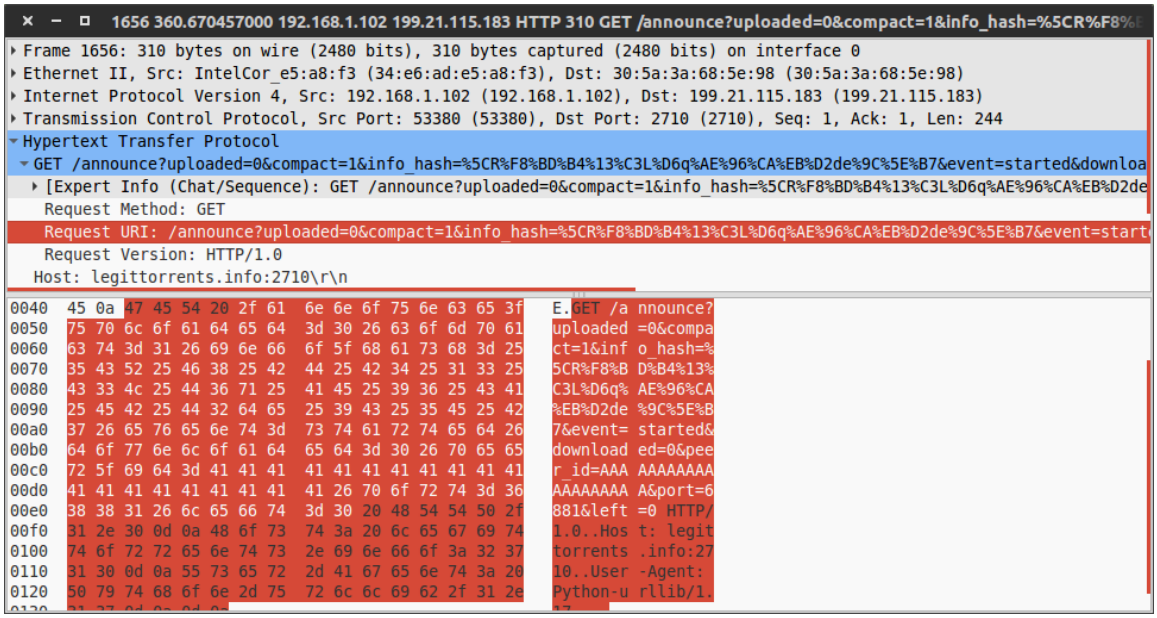
Typ protokolu	Počet výskytov
HTTP	1765
UDP	342
HTTPS	40

Tabuľka 2.2: Zobrazenie počtu výskytov jednotlivých typov protokolov v zložke s analyzovanými torrentovými súbormi.

- `ip` - Voliteľný parameter, ktorý určuje na akej IP adrese sa daný peer vyskytuje. Pri neprítomnosti tohto parametru si vie tracker zistiť IP adresu peera z IP protokolu.
- `port` - Číslo portu na ktorom peer naslúcha. Chovanie popísané v štandardizovanej BitTorrent dokumentácií je, že peer sa pokúsi ustanoviť pasívne otvorenie komunikácie na porte 6881 a v prípade, že je tento port zabraný tak vyskúša otvoriť port 6883 a takto inkrementálne postupuje až k portu 6889. V praxi ale peer nie je na tieto porty fixovaný a dnešní torrentoví klienti umožňujú zvoliť si vlastný naslúchací port.
- `left` - Počet častí súboru, ktoré daný peer ešte musí prijať aby mal kompletnú kópiu dát.
- `uploaded` - Počet častí súboru, ktoré daný peer doposiaľ odoslal.
- `downloaded` - Počet častí súboru, ktoré daný peer doposiaľ prijal.
- `event` - Tento parameter je voliteľný a popisuje stav sťahovania.
- `compact` - Príznak podľa ktorého tracker rozozná či má odpovedať normálnou správou alebo správou kde sú hodnoty reprezentované vo forme binárneho reťazca, čím sa šetrí prenosové pásmo. Tento príznak môže nadobúdať hodnoty 1 alebo 0 [4, 19].

Po tom ako peer odošle správu trackeru s danými požiadavkami, tracker odošle naspäť klientovi odpoveď v kódovaní bencode. Názorná ukážka takejto správy je na obrázku 2.3. Táto správa nesie následujúce informácie:

- `failure reason` - Správa o chybe pri zlyhaní requestu.
- `interval` - Doba po ktorej peer môže znovu zaslať request.
- `peers` - Zoznam, ktorý obsahuje tri atribúty pre každého peera v danom swarme.



Obr. 2.2: Obsah paketu so správou announce request.

- `peers/peer id` - Identifikátor peera v swarme. Tento identifikátor si peer zvolí sám.
- `peers/port` - Port peera na ktorom naslúcha.
- `peers/ip` - IP adresa peera [4, 19].

2.2.3 UDP Tracker Protocol

UDP Tracker Protocol je rozšírenie pre komunikáciu s trackerom. Ako som uviedol vyššie v predchádzajúcej podkapitole 2.2.2, komunikácia s trackerom prebieha pomocou protokolu HTTP, ktorý sa v počiatočných fázach siete BitTorrent používal. Protokol HTTP predstavoval pre trackre problém z hľadiska réžie pri prenose dát, pretože rámec po zostavení obsahuje v sebe zapuzdrené protokoly IP, TCP a HTTP. Taktiež pre komunikáciu pomocou TCP je použitých viac paketov pre udržanie komunikácie. UDP je na rozdiel od TCP protokol, ktorý neudržiava spojenie medzi dvoma uzlami a teda odpadá réžia paketov, ktoré sú potrebné pre udržanie spojenia. Toto je výhodné hlavne v prospech trackera, ktorý si týmto prístupom šetrí kapacitu prenosového pásmo. UDP Tracker Protocol obsahuje tri rozdielne správy typu connect, announce a scrape, ktoré klient odosiela na tracker.

- **connect** – Prvá správa pomocou ktorej sa zo servera získa identifikátor spojenia. Tento identifikátor sa použije v ďalších typoch správ. Týmto typom správy sa vymieňajú informácie potrebné pre ustanovenie spojenia.
- **announce** – Správa typu announce obsahuje tie isté parametre aké obsahuje správa announce pri použití HTTP protokolu a sú popísané v predchádzajúcej podkapitole 2.2.2. Tracker odpovedá na túto správu odpoveďou, ktorá je v bencode kódovaní a obsahuje okrem čísla portov a ip adries peerov aj počet aktívnych seederov a leecherov v swarme.

```
1658 360.821194000 199.21.115.183 192.168.1.102 HTTP 183 HTTP/1.0 200 OK
▸ Transmission Control Protocol, Src Port: 2710 (2710), Dst Port: 53380 (53380), Seq: 1, Ack:
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.0 200 OK\r\n
    ▸ [Expert Info (Chat/Sequence): HTTP/1.0 200 OK\r\n]
      Request Version: HTTP/1.0

0000  34 e6 ad e5 a8 f3 30 5a  3a 68 5e 98 08 00 45 00   4.....0Z :h^...E.
0010  00 a9 c0 07 40 00 2b 06  92 6c c7 15 73 b7 c0 a8   ....@.+. .l..s...
0020  01 66 0a 96 d0 84 6a b3  85 9e f1 29 fd c3 80 18   .f....j. ...)....
0030  00 7a 62 f1 00 00 01 01  08 0a 5c b5 45 98 0f 0e   .zb..... ..\.E...
0040  9e 87 48 54 54 50 2f 31  2e 30 20 32 30 30 20 4f   ..HTTP/1 .0 200 O
0050  4b 0d 0a 0d 0a 64 38 3a  63 6f 6d 70 6c 65 74 65   K....d8: complete
0060  69 31 33 38 65 31 30 3a  69 6e 63 6f 6d 70 6c 65   i138e10: incomple
0070  74 65 69 33 65 38 3a 69  6e 74 65 72 76 61 6c 69   tei3e8:i ntervali
0080  31 38 30 30 65 31 32 3a  6d 69 6e 20 69 6e 74 65   1800e12: min inte
0090  72 76 61 6c 69 31 38 30  30 65 35 3a 70 65 65 72   rvali180 0e5:peer
00a0  73 31 38 3a b1 09 44 64  c0 08 02 56 84 25 a7 a6   s18:..Dd ...V.%..
00b0  5f 5e 45 2b 4f b0 65                               ^E+0.e
```

Obr. 2.3: Obsah paketu so správou announce response.

- **scrape** – Správa typu scrape je principiálne podobná ako request typu announce. Tracker vracia peerovi počet seederov, leecherov a počet stiahnutí torrentu, ktorý je daný info hashom. Nie je tu ale zaistená validita týchto hodnôt, pretože s nimi môže byť následne manipulované [19].

2.2.4 BitTorrentový klient

BitTorrentový klient je software, ktorý automatizuje jednotlivé dielčie úkony, ktoré vedú ku kompletnému stiahnutiu súboru od ostatných peerov v swarme pre daný torrent. Užívateľovi stačí pre stiahnutie požadovaného súboru nahrať do daného bittorrentového klienta torrentový súbor. Mnohé dnešné implementácie bittorrentových klientov poskytujú možnosti ako napríklad možnosť použitia rozšírenia pre získavanie kontaktných informácií na ostatných peerov v swarme, možnosť použitia špecifického protokolu pre komunikáciu s peermi alebo možnosť šifrovania komunikácie. Taktiež sa títo klienti vo väčšine prípadov nedržia striktného využívania portov špecifikovaných v dokumentácií o protokole BitTorrent a využívajú buď vlastné prednastavené porty alebo používajú algoritmy pre náhodné volenie portov pre komunikáciu.

2.2.5 uTP Protocol

Tento typ protokolu bol vyvinutý ako rozšírenie a náhrada prenosu dát nad TCP. Prevádzka v sieti BitTorrent typicky beží na pozadí a tak by mala mať nižšiu prioritu ako ostatné služby, ako napr. čítanie emailu alebo prehliadanie webu. Keď však využívame TCP spojenie, BitTorrent dokáže rýchlo zaplniť odosielaciu frontu. Toto zaplnenie je spôsobené tým že pri začatí sťahovania pomocou protokolu BitTorrent implementovaným nad protokolom TCP sa daný bittorrentový uzol snaží nakontaktovať viacero peerov a tým sa odosielacia fronta zaplní týmito TCP paketmi. Čím viac takýchto spojení vzniká, tým viac sa zaplňa odosielacia fronta a alokuje sa viac prenosového pásma pre bittorrentovú pre-

mávku. Toto má za dôsledok menšie prenosové pásmo pre ostatnú TCP premávku a tým sa táto premávka spomaluje.

Niektoré riešenia tohto problému spočívali v manuálnom prerozdelení šírky pásma pre bittorrentovú premávku a ostatnú TCP premávku. Tieto riešenia ale nemali moc optimálne využitie kvôli tomu, že tieto nastavenia by bolo nutné vykonávať na každom zariadení zvlášť a taktiež pri prerozdelení týchto premávok by mohlo dôjsť k tomu, že pri nevyužívaní klasickej premávky by časť šírky prenosového pásma určená pre túto premávku ostala nevyužitá.

Alternatívnym riešením tohto problému bola implementácia protokolu uTP (Micro Transport Protocol). uTP je navrhnuté tak, že dokáže regulovať svoju rýchlosť od závislosti stavu odosielacej fronty. V prípade že je odosielacia fronta plná obmedzí protokol uTP svoju odosielaciu rýchlosť a tak prenechá možnosť odoslaniu ostatnej premávky. Protokol uTP je implementovaný nad protokolom UDP a toto umožnuje vytvoriť pre daný protokol mechanizmus okna pre reguláciu zahltenia siete a taktiež svoj vlastný spôsob nadviazania spojenia - handshake. Toto pri implementácií protokolu BitTorrent nad protokolom TCP nebolo možné, keďže TCP protokol už tieto mechanizmy má implementované [14].

2.2.6 Peer Wire Protocol

Pre komunikáciu medzi dvoma peermi v sieti BitTorrent je využívaný Peer Wire Protocol. Tento protokol pracuje nad protokolmi TCP a uTP. Protokol v sebe obsahuje indexovanie jednotlivých blokov, ktoré sú umiestnené v torrent súbore pod kľúčom *pieces*. Po tom, ako peer dokončí sťahovanie daného bloku overí, či sa zhoduje jeho hash hodnota s hodnotou požadovanou. Pokiaľ sa hodnoty zhodujú, peer následne oznámi ostatným peerom v swarme, že daným blokom už disponuje a tento blok môže byť od neho stiahnutý ďalšími peermi.

Nadviazanie spojenia za účelom stiahnutia bloku začína špeciálnym typom správy - *handshake* 2.4. Handshake správa začína vždy decimálnou hodnotou 19 (v protokole hexadecimálne značené 0x13), za ktorou nasleduje reťazec "BitTorrent protocol". Uvedená hodnota handshake správy 19 udáva dĺžku tohto reťazca. Za týmto reťazcom nasleduje 8 bajtov rezervovaných pre ďalšie rozšírenia, 20 bajtová hodnota *info hash*, ktorá identifikuje zdieľaný torrent a 20 bajtový identifikátor peera pre daný swarm *peer id*.

Pri handshake správe si obaja peeri medzi sebou vymieňajú hodnoty *info hash* a *peer id*. Info hash je tá istá hodnota, ktorá bola pri správe announce zaslaná na tracker. Pokiaľ sa však pri správe typu handshake nezhodujú hodnoty info hash na oboch stranách peerov, spojenie sa preruší. Po úspešnom prebehnutí handshake správy peer disponujúci daným blokom odošle index identifikujúci tento blok prijímajúcemu peerovi a v ďalšej správe sa daný blok prenesie na peer, ktorý si tento blok vyžiadal [4].

2.2.7 DHT Protocol

DHT Protocol je rozšírenie pre vyhľadávanie peerov pre požadovaný torrent bez potreby trackera. Kontaktné informácie na jednotlivých peerov v sieti sú uložené do distribuovanej hash tabuľky (DHT). Zúčastnený peer, ktorý využíva toto rozšírenie spravuje vlastný DHT uzol, ktorý komunikuje zasielaním správ cez UDP protokol v kódovaní bencode. DHT v sieti BitTorrent je implementovaná na základe dizajnu Kademlia. Princíp fungovania a komunikácie v DHT je nasledovný:

Uzol si vygeneruje vlastný 20 bajtový identifikátor ID. Každý uzol si udržiava smerovaciu tabuľku, ktorá mapuje ID ostatných uzlov na ich IP adresu a číslo UDP portu. Čím bližšie sú ID ostatných uzlov v sieti k ID daného uzla, tým viac uzlov je uložených v jeho smerovacej tabuľke. Vzdialenosť týchto uzlov sa meria na základe bitovej operácie XOR. Uzol si taktiež

```
1079207 2699.306423 147.229.12.246 71.72.219.214 BitTorrent 122 Handshake
Frame 1079207: 122 bytes on wire (976 bits), 122 bytes captured (976 bits)
Ethernet II, Src: AsrockIn_a2:fc:fa (d0:50:99:a2:fc:fa), Dst: ExtremeN_1d:4e:30 (00:04:96:1
Internet Protocol Version 4, Src: 147.229.12.246 (147.229.12.246), Dst: 71.72.219.214 (71.7
Transmission Control Protocol, Src Port: 63076 (63076), Dst Port: 63780 (63780), Seq: 1, Ac
BitTorrent
  Protocol Name Length: 19
  Protocol Name: BitTorrent protocol
  Reserved Extension Bytes: 0000000000100005
  SHA1 Hash of info dictionary: 99091495caf45b236f79b944b7d4f89347854464
  Peer ID: 2d4445313343302d313129393976735241485133

0000  00 04 96 1d 4e 30 d0 50  99 a2 fc fa 08 00 45 00   ....N0.P ......E.
0010  00 6c a6 c4 40 00 3e 06  d1 cd 93 e5 0c f6 47 48   .l..@.>. ......GH
0020  db d6 f6 64 f9 24 83 c2  b8 67 12 a5 39 95 50 18   ...d.$.. .g..9.P.
0030  00 e5 ee 6c 00 00 13 42  69 74 54 6f 72 72 65 6e   ...l...B itTorren
0040  74 20 70 72 6f 74 6f 63  6f 6c 00 00 00 00 00 10   t protoc ol......
0050  00 05 99 09 14 95 ca f4  5b 23 6f 79 b9 44 b7 d4   ........ [#oy.D..
0060  f8 93 47 85 44 64 2d 44  45 31 33 43 30 2d 31 31   ..G.Dd-D E13C0-11
0070  29 39 39 76 73 52 41 48  51 33                     )99vsRAH Q3
```

Obr. 2.4: Obsah paketu so správou handshake.

udržuje tabuľku, ktorá mapuje info hash hodnoty torrentov, ktoré sú v blízkosti ID tohto uzlu na zoznamy IP adries a TCP portov peerov, o ktorých je známe, že daný torrent sťahujú.

Pri vyhľadávaní peerov v DHT dotazujúci uzol zasiela najbližším známym uzlom iteratívne dotazy, ktoré obsahujú info hash sťahovaného torrentu. Dotaz prebieha zasielaním správy *get_peers*. Pokiaľ dotazovaný uzol momentálne obsahuje peerov, ktorý zdieľajú daný torrent, vráti dotazovanému uzlu zoznam kontaktných informácií na týchto peerov. Ak dotazovaný peer momentálne nemá žiadnych peerov pre daný torrent vo svojej tabuľke, vráti dotazovanému uzlu ID najbližšieho uzla pre daný torrent zo svojej tabuľky.

Ak peer začne sťahovať torrent mal by oznámiť tento fakt ostatným blízkym uzlom pre daný torrent. Túto akciu realizuje peer za pomoci správy *announce_peer*.

Fakt, že peer využíva rozšírenie DHT je oznámené v správe handshake v protokole Peer Wire Protocol tým, že je v časti ôsmich rezervovaných bajtov nastavený posledný bit [19].

2.3 Seedbox

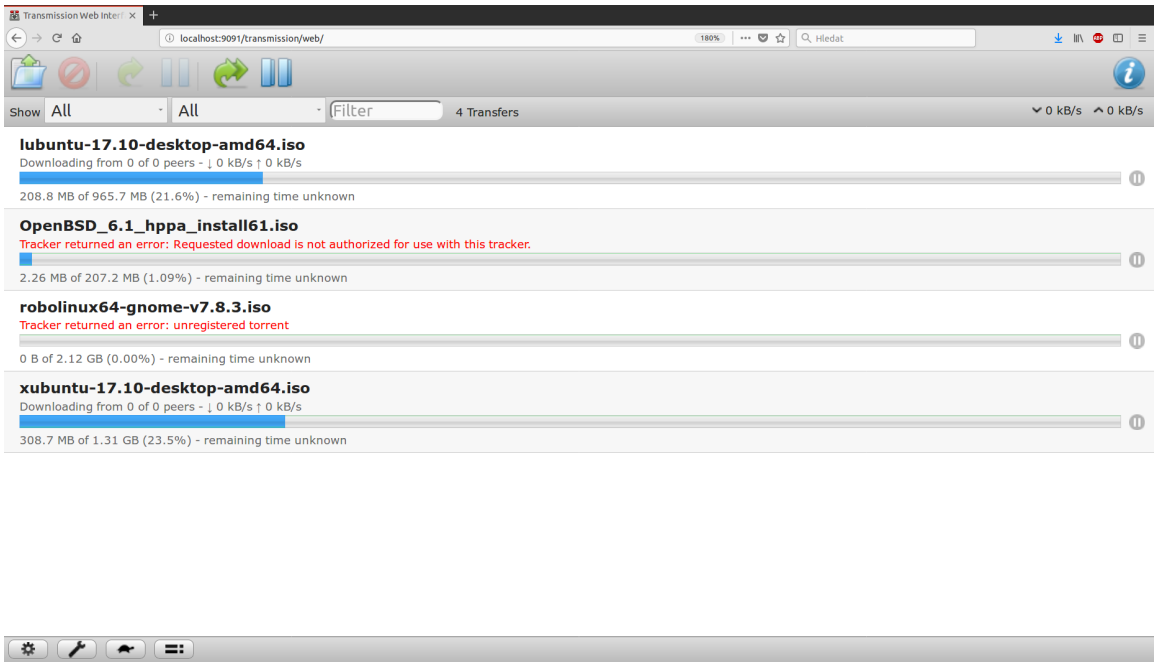
Seedbox je server, ktorý je využívaný pre účely nahrávania a sťahovania digitálnych súborov. V tejto práci budem rozoberať problematiku seedboxov, ktoré zdieľajú dáta v rámci siete BitTorrent. To však ale neznamená, že sa seedboxy využívajú len v tomto type P2P siete (napríklad eDonkey2000).

Hlavnou úlohou seedboxu je udržovať stále zdieľanie torrentov. V seedboxoch je často pravidlom udržovanie ratio nad hodnotou 1.0 a teda odoslať do siete rovnaký alebo väčší počet dát ako bolo stiahnutých. Z hľadiska umiestnenia týchto seedboxov v sieti rozoznávame súkromný seedbox a zdieľaný seedbox.

Zdieľaný seedbox je umiestnený do veľkých dátových centier a vysokorýchlostných sietí, kde sa prenosové rýchlosti pohybujú v rozmedzí od 100 Mbit/s (12,5 MB/s) do 10 Gbit/s (1250 MB/s). Takýto seedbox poskytuje vzdialené úložisko a vysokú rýchlosť prenosu dát viacerým užívateľom, ktorí navzájom tento server zdieľajú. Poskytovatelia takejto služby umožňujú svojim klientom pripojiť sa na seedbox pomocou webového užívateľského rozhra-

nia. Klienti nevlastnia hardware, ale prenajímajú si ho od svojho poskytovateľa. Ak klient chce začať so zdieľaním súboru, pripojí sa na seedbox pomocou webového rozhrania a nahrá torrent súbor alebo URL tohto súboru na seedbox. Po nahratí tohoto súboru sa daný torrent začne ihneď sťahovať a po ukončení sťahovania je následne seedovaný. Ak je súbor kompletne stiahnutý, môže si ho následne užívateľ preniesť na svoj lokálny počítač pomocou protokolou FTP alebo SFTP. Niektorí poskytovatelia zdieľaných seedboxov ponúkajú aj možnosť stiahnutia cez webový prehliadač pomocou protokolu HTTP. Pretože takýto typ seedboxu poskytuje službu viacerým klientom, je ich IP adresa verejná. Zdieľané seedboxy poskytujú klientom aj istú mieru súkromia, keďže klient počas sťahovania súborov nie je súčasťou BitTorrent siete [8]. Taktiež tieto typy seedboxov využívajú užívatelia súkromných trackerov. Súkromné trackre striktne kontrolujú ratio každého peera a pri nedodržiavaní ratio 1.0 a viac nastávajú pre daného peera sankcie vo forme obmedzenia používania tohoto trackera [7].

Súkromný seedbox nie je umiestnený v dátových centrách. Takéto typy seedboxov nemajú verejnú IP adresu a na rozdiel od zdieľaných seedboxov nedosahujú vysokých prenosových rýchlostí. Môže si ho vytvoriť bežný užívateľ internetu a ponechať ho spustený na svojom lokálnom počítači alebo na zariadení vo svojej lokálnej sieti.



Obr. 2.5: Webové rozhranie klienta Transmission.

K vytvoreniu tohoto typu seedboxu postačí, ak si užívateľ zaobstará démona z nejakého zo známych bittorrentových klientov 2.5 (napr. v prípade klienta Transmission démon `transmission-daemon`) alebo ponechá klienta trvalo spusteného [5]. Problém s týmto typom seedboxov nastáva ak sú v lokálnej sieti pemanentne spustené a tým sa zahlcuje prenosové pásmo danej siete, čo môže viesť k jej spomaleniu. Predmetom tejto práce bude hľadanie charakteristických znakov práve tohto typu seedboxov.

Kapitola 3

Analýza a detekcia P2P komunikácie

V tejto kapitole sa budeme zaoberať metódami, ktoré môžme použiť pri detekcií P2P komunikácie a taktiež si uvedieme nástroje, ktoré sú vhodné pre analýzu a sledovanie komunikácie celkovo.

3.1 Metódy detekcie P2P komunikácie

Aby bolo možné rozoznať zariadenia, ktoré komunikujú pomocou P2P protokolov je dôležité vedieť na nejakej vhodnej úrovni detekovať tento typ komunikácie. V mojej práci uvádzam tri najbežnejšie metódy používané pri detekcií P2P komunikácie. Sú to tieto metódy: 1. detekcia pomocou čísel portov, 2. metóda detekcie obsahu paketu a 3. metóda detekcie pomocou sieťových tokov. Pri každej tejto metóde tejto práci uvádzam výhody a nevýhody jej používania.

3.1.1 Detekcia pomocou čísel portov

Jedná sa o najstaršiu a najjednoduchšiu metódou detekcie P2P komunikácie. Je založená na predpoklade, že každá P2P aplikácia má predvolený port, cez ktorý v rámci siete komunikuje s ostatnými peermi. Pri detekcii sa sleduje sieťová prevádzka a nahliada sa do L4 vrstvy TCP a UDP paketov. Predvolené porty niektorých P2P aplikácií sú vyobrazené v tabuľke 3.1. V L4 vrstve sa porovnáva číslo zdrojového alebo cieľového portu z odchyteného paketu s používaným číslom portu detekovanej P2P aplikácie. Pri zhode týchto portov je komunikácia označená za P2P komunikáciu. Sieťovému administrátorovi stačí k vykonaniu takejto detekcie len to, aby prehľadal záznamy spojení, ktoré využívali tieto porty [6].

Výhodou tejto metódy je rýchle porovnanie a detekovanie už na L4 vrstve. Nevýhoda tejto detekcie spočíva v tom, že dnešní klienti P2P aplikácií umožňujú užívateľom zmeniť predvolené číslo portu pre komunikáciu. Niektoré modernejšie P2P aplikácie dokonca využívajú rôzne algoritmy pre náhodné generovanie čísel portov a tak sa táto metóda stáva pri detekovaní danej P2P komunikácie neúčinná a treba zapojiť iné metódy pre zdetekovanie komunikácie.

Účinnosť tejto metódy je najväčšia pri detekcii portov z rozsahu 0 - 1023, keďže sa v ňom vyskytujú aplikácie a služby s nemennými číslami portov (napr. HTTP, FTP a SMTP).

Aplikácia	Číslo portu
BitTorrent	6881-6889
WinMx	6699(TCP), 6257(UDP)
EMule	4662(TCP), 4672(UDP)
Morpheus	6346-6347
Edonkey	4662(TCP)

Tabuľka 3.1: Predvolené porty P2P aplikácií. Informácie sú zo zdroja [6].

3.1.2 Detekcia obsahu paketov

Princíp metódy je založený na tom, že sa pri detekcii prenikne do L7 aplikačnej vrstvy paketu a hľadajú sa charakteristické reťazce daného P2P protokolu. P2P protokoly odosielajú v niektorých častiach komunikácie správy, ktoré majú v aplikačnej vrstve charakteristické znaky alebo reťazce, ktorých nájdenie môže viesť k odhaleniu tejto P2P komunikácie. Takéto správy sa v P2P aplikáciach odosielajú väčšinou pri nadväzovaní spojenia medzi dvoma uzlami. Detekčné nástroje pracujúce na tejto metóde vyhľadávajú tieto charakteristické znaky aplikovaním regulárnych výrazov pri detekcii daného paketu [6, 11].

Nespornou výhodou tejto metódy je vysoká presnosť detekcie. Napriek vysokej presnosti detekcie má ale táto metóda aj zopár nevýhod. Keďže sa P2P aplikácie neustále vyvíjajú a rozširujú, môžu sa charakteristické znaky týchto aplikácií časom zmeniť a tak sa musí zoznam týchto charakteristík v detekčných nástrojoch rozširovať. Taktiež sa táto metóda stáva neúčinnou, ak komunikujúce si uzly ustanovia medzi sebou šifrovanú komunikáciu a dáta sa tak stávajú nečitateľné. Ďalšou nevýhodou je aj náročnosť metódy na potrebné výpočtové zdroje. Čím je šírka pásma danej siete vyššia, tým viac sa tieto výpočtové zdroje zaťažujú, pretože sa musí preveriť každý odoslaný a každý prijatý paket. Sporné je taktiež samotné nahliadanie do obsahu paketov, čím sa narušuje súkromie užívateľov.

3.1.3 Detekcia na základe analýzy sieťových tokov

Táto metóda spočíva v detekcii dátovej časti paketu bez analýzy jej obsahu na aplikačnej vrstve. Metóda vykonáva analýzu nad záznamami tokov. *Tok* je sekvencia za sebou idúcich paketov zo spoločnou vlastnosťou, ktoré prechádzajú bodom pozorovania v určitom časovom intervale [3]. Obvyklými spoločnými vlastnosťami týchto paketov sú adresné informácie - zdrojová adresa, zdrojový port, cieľová adresa, cieľový port a typ protokolu. Tieto informácie sú získané z L3 a L4 vrstvy daného paketu. Obsah paketu sa neukladá. Štatistické údaje o toku sú uložené v zázname o toku. Záznam toku obsahuje pre daný tok adresné informácie, informáciu o veľkosti toku a dobu trvania toku. Táto metóda je rýchlejšia ako metóda detekcie obsahu paketov, keďže predmetom analýzy sú len štatistické informácie o paketoch. Taktiež sa pri tejto metóde nezasahuje do súkromia užívateľov a pri detekcii nezáleží na tom, či je sledovaná komunikácia šifrovaná alebo nie je. Detekcia vyhľadáva určité charakteristické správanie P2P aplikácií v záznamoch o tokoch komunikácie [11]. Dôležité parametre, ktoré sa pri tejto analýze sledujú sú tieto:

- Čísla portov
- Zdrojové a cieľové IP adresy
- Typ protokolu TCP/UDP

- Veľkosť paketov
- Počet prijatých paketov za sekundu
- Rýchlosť prenosu dát
- Pomer úspešných a neúspešných spojení

Výhoda tejto metódy spočíva v nižších nárokoch na výpočtové zdroje na rozdiel od metódy detekcie obsahu paketov, v ktorej sú nároky na výpočtové zdroje vysoké. Nevýhoda tejto metódy spočíva v tom, že charakter chovania niektorých P2P aplikácií sa odhaduje len ťažko, prípadne sa ich charakter chovania môže časom zmeniť. Užívateľ môže taktiež zmeniť čísla portov v aplikácií, zapnúť šifrovanie alebo obmedziť počet spojení. Ďalšou nevýhodou je fakt, že niektoré aplikácie, ktoré nezdieľajú dáta, fungujú tiež na princípe P2P a pri detekcii tak môže dôjsť ku zdetekovaniu aj tohto typu komunikácie - false positives.

Spoločné vlastnosti P2P protokolov sú:

- Obojsmerné spojenia – Princíp P2P sietí pre zdieľanie súborov je sťahovanie a odosielanie dát ostatným uzlom v sieti. Ak užívateľ sťahuje nejaký súbor a už obdržal časť dát z tohto súboru, tak ich odosiela ostatným uzlom, ktorí sa tiež podieľajú na sťahovaní tohto súboru. Spoločným znakom P2P aplikácií je to, že sa odošle rovnaký počet dát ako sa aj prijme. Komunikácia je teda symetrická.

- Veľký počet nadviazaných spojení – Pri zdieľaní súborov je daný súbor rozdelený na niekoľko blokov, ktoré si uzly medzi sebou navzájom vymieňajú, pokým nebudú mať kompletný súbor. Ak uzol chce získať tento blok, musí sa napojiť na viac uzlov. Takýmto spôsobom si zaobstarávajú získanie daného súboru všetky uzly v sieti. Následkom toho je, že na každý uzol v sieti je vytváraný veľký počet spojení spôsobený zasielaním žiadostí na daný blok súboru od ostatných uzlov, ktorý sa na zdieľaní súboru podieľajú.

- Použitie UDP aj TCP protokolu – Väčšina P2P aplikácií využíva pre komunikáciu protokoly TCP a UDP. Napríklad pri BitTorrente sa v používal TCP protokol na zasielanie dát a UDP pre nakontaktovanie ostatných uzlov v sieti. Dnešná implementácia BitTorrentu obsahuje už rozšírenia ktoré umožňujú prenášať dáta aj pomocou protokolu UDP.

- Veľký počet neúspešných spojení – Táto vlastnosť vyplýva z faktu, že užívatelia si môžu svoje klientské aplikácie vypnúť a tým sa stávajú niektoré uzly v sieti nedostupné. Informácie o uzle však bývajú v sieti ešte nejakú dobu aktívne. Pokus o nakontaktovanie takéhoto uzlu následne vedie k neúspešnému spojeniu. Peer, ktorý si vyžiadal tento blok ešte niekoľkokrát zopakuje pokus o nakontaktovanie takéhoto uzla.

- Maximálna veľkosť paketov – Pri prenose dát sa P2P aplikácie snažia posielať čo najväčšie pakety. Táto veľkosť sa pohybuje okolo hodnoty 1400 bajtov. P2P aplikácie, ktoré prenášajú hlas však majú značne nižšie veľkosti paketov a to hlavne z dôvodu zabezpečenia vysokej odozvy.

- Čísla portov nad 1024 – Dnešné P2P aplikácie podporujú náhodné volenie portov pre komunikáciu. Porty P2P aplikácií teda nemajú pevne zvolené porty, ale spoločnou

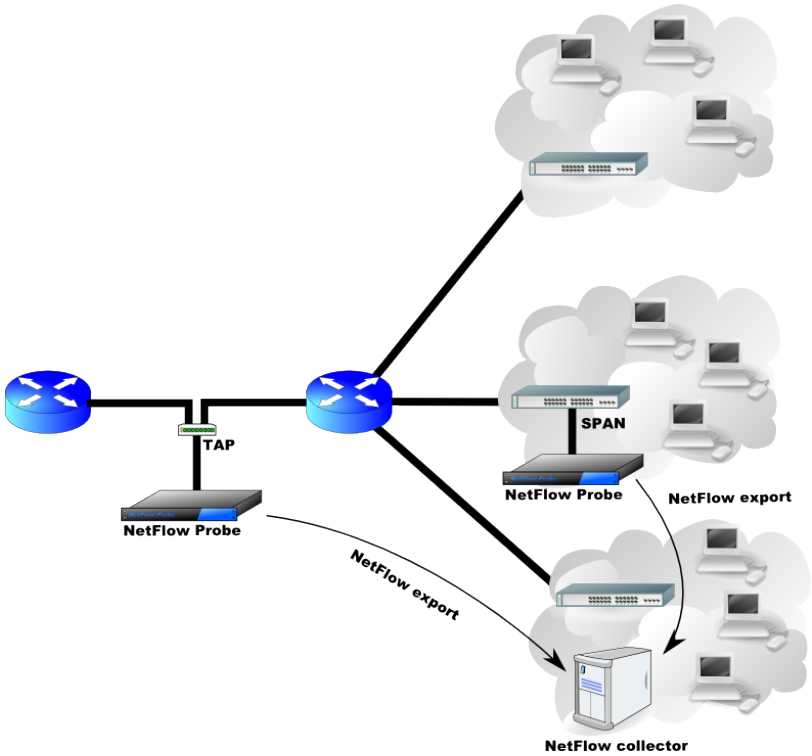
vlastnosťou týchto portov je to, že sa pohybujú nad číslom 1024. Tento fakt je daný tým, že k použitiu čísla portu pod 1024 je potrebné povolenie od systému, na ktorom je daná P2P aplikácia nainštalovaná.

3.2 Nástroje pre analýzu P2P komunikácie

V tejto kapitole budú popísané nástroje, ktorých funkcionalita sa dá využiť pri analýze a detekcii nielen P2P komunikácie, ale aj iných druhov komunikácií.

3.2.1 NetFlow

NetFlow možno definovať ako systém, ktorý vyvinula spoločnosť Cisco Systems za účelom zbierania informácií zo sieťovej prevádzky. Daný systém NetFlow sa stal sa štandardom pri sledovaní dát v oblasti priemyselných sietí. NetFlow poskytuje správcom sietí informácie o dianí v sieti pomocou tokov. Základné prvky architektúry NetFlow sú *exportér*, *kolektor* a *protokol NetFlow*. Zapojenie jednotlivých prvkov systému NetFlow a jeho jednotlivých prvkov v sieti môžme vidieť na danom obrázku 3.1.



Obr. 3.1: Zapojenie systému NetFlow v sieti. Zdroj obrázka [16]

Zber informácií a získavanie štatistík o tokoch prebieha na zariadení exportér, ktorý zbiera tie pakety, ktoré týmto exportérom prechádzajú a informácie o pakete ukladá do zodpovedajúcich tokov. Exportér môže byť v sieti umiestnený na sieťovom zariadení ako je smerovač alebo sonda. Funkciu exportéra môže vykonávať sieťové zariadenie samotné alebo exportérom môže byť software nainštalovaný na danom sieťovom zariadení. V Exportéri sa vytvárajú taktiež záznamy o tokoch, ktoré sú ukladané do pamäte *NetFlow cache*. Ak je tok

považovaný za ukončený, potom je daný záznam o toku odoslaný na zariadenie kolektor. Tok sa považuje za ukončený, ak spĺňa niektorú z následujúcich podmienok:

- Detekovanie konca toku – V prípade protokolu TCP je ukončenie spojenia dané príznakmi FIN a RST, ktoré sú nastavené v hlavičke TCP paketu. Pri ich zdetekovaní sa ukladanie informácií o danom toku preruší a celý záznam o tomto toku je prenesený na kolektor.
- Neaktivita toku – Ak je tok dlho neaktívny, znamená to, že dlhšiu časovú jednotku pakety patriace k danému toku neprešli daným exportérom. V tomto prípade sa nastavuje neaktívny timeout a po jeho vypršaní sa záznam o neaktívnom toku prenáša na kolektor.
- Dlho trvajúci tok – Pri komunikácií dvoch uzlov v sieti môže dôjsť aj k stavu, že niektoré toky majú nadmerne dlhé trvanie. V takomto prípade sa daným tokom nastavuje aktívny timeout a tieto toky sú tak opakovane exportované po vypršaní daného timeoutu.
- Zaplnenie pamäte NetFlow cache – V prípade, že sa v exportéri v pamäti NetFlow cache nazhromaždí počet sledovaných záznamov o tokoch, pričom sa zaplní daná pamäť, nemajú sa prichádzajúce toky kam ďalej ukladať. Exportér teda nájde najstaršie toky ktoré sú v pamäti uložené a prenesie ich na kolektor, čím sa uvoľní pamäť a prichádzajúce toky sa môžu uložiť do pamäti.

Záznamy o tokoch sú cez sieť prenášané v protokole NetFlow. Protokol pracuje nad protokolom UDP alebo SCTP. Pri konfigurácií systému NetFlow musí byť zasielanie tohto protokolu nastavené v exportéri a to na IP adresu kolektoru a na port, na ktorom bude kolektor prijímať pakety. Štandardné číslo portu kolektoru je 2055, ale taktiež sa používajú aj porty 9555 alebo 9995. Po vyexportovnaní záznamu o toku sa tento záznam v pamäti NetFlow cache vymazáva. V prípade používania UDP paketu môže dôjsť k strate tohto záznamu o toku, ak sa po ceste ku kolektoru paket poruší. V súčastnosti bolo vyvinutých desať verzií tohto protokolu z čoho najpoužívanejšia je piata verzia.

Kolektor je aplikácia, ktorá prijíma pakety protokolu NetFlow, ktoré v sebe obsahujú záznamy o tokoch a spracováva a ukladá štatistiky z týchto záznamov na disk alebo do databáze. Dokáže taktiež agregovať dáta podľa zadaných hodnôt (napr. agregácia podľa zdrojovej IP adresy). Kolektor môže prijímať NetFlow pakety z viac ako len jedného exportéru. Umožňuje aj graficky prezentovať získané dáta a tiež vykonávať nad týmito dátami rôzne dotazovacie operácie.

3.2.2 NFDUMP

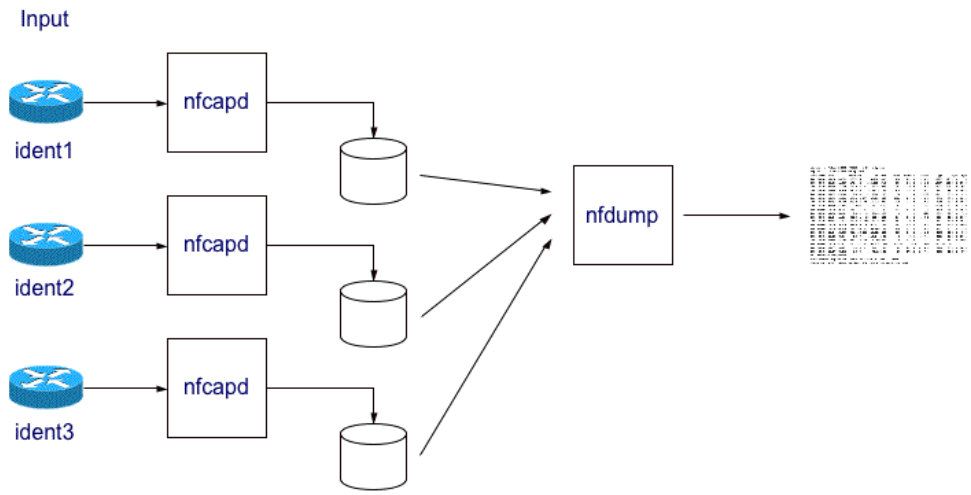
Jedná sa o sadu nástrojov, ktorá dokáže spracovať a vyhodnocovať NetFlow dáta cez rozhranie príkazového riadku. Túto sadu nástrojov vytvoril Peter Haag, je voľne šíriteľná pod licenciou BSD a jej nástroje sú implementované v jazyku C. Nástroje podporujú spracovanie protokolu NetFlow verzie v5,v7 a v9. Jednotlivé nástroje tejto sady sú:

- **nfcapd** (NetFlow capture daemon) – je démon, ktorý zachytáva NetFlow záznamy z exportéra, ukladá ich do súborov a po určitej časovej jednotke tieto súbory rotuje (zvyčajne sa nastavuje interval 5 minút). Súbory sú ukladané pod názvom, ktorý v sebe nesie informáciu o čase, kedy bola vykonaná posledná rotácia súboru.

Formát názvu vyzerá následovne `nfcapd.YYYYMMddhhmm`. V prípade názvu súboru `nfcapd.201804130830` bola teda posledná rotácia daného súboru vykonaná 13-tého apríla 2018 o 8 hodine a 30 minúte a daný súbor v sebe obsahuje záznamy o tokoch od tejto časovej jednotky vyššie. Pre každý exportér je spustený jeden tento démon.

- **nfdump** (NetFlow dump) – program, ktorý číta binárne dáta uložené démonom nfcapd na disk a vypisuje ich v textovej podobe, prípadne v binárnej podobe pre potrebu ich ďalšieho spracovania 3.2. Umožňuje taktiež vytvárať štatistiky o tokoch, agregovať tieto toky a vytvárať filtre pomocou dotazovacieho jazyka. Dokáže vykonať analýzu nad jedným súborom obsahujúcim NetFlow záznamy alebo nad celým adresárom, v ktorom sú tieto súbory uložené.

- **nfreplay** (NetFlow replay) – tento program číta dáta uložené démonom nfcapd podobne ako nfdump s tým rozdielom, že po prečítaní nfreplay tieto dáta odosiela na zvolené koncové zariadenie alebo na multicastovú skupinu. V prípade použitia filtra sú odoslané len dáta ktoré spĺňajú filtrovacie požiadavky. Tento nástroj sa používa aj na sledovanie toho, ako bude vzdialené zariadenie reagovať na NetFlow záznamy prenášané po linke.



Obr. 3.2: Obrázok znázorňuje odchytávanie tokov démonmi nfcapd z troch rôznych sieťových zariadení. Nfdump tieto dáta následne číta a zobrazuje. Zdroj obrázka [1].

3.2.3 Softflowd

Začiatok vývoja projektu softflowd začal v roku 2002. O vývoj projektu sa stará Damien Miller a s drobnými úpravami je udržovaný až dodnes. Projekt je distribuovaný pod licenciou BSD a teda je voľne šíriteľný. Umožňuje sledovanie IPv4 a IPv6 paketov. Podporuje export protokolu NetFlow verzie v1,v5 a v9. Pre odosielanie NetFlow dát je možné použiť UDP na IPv4 protokole, ale taktiež aj UDP na IPv6 protokole alebo odosielanie s použitím multicastu. Program beží ako samostatný démon, ktorý naslúcha na jednom rozhraní v promiskuitnom režime (t.zn v režime, kedy sieťová karta daného zariadenia zachytáva aj dáta, ktoré nie sú pre dané zariadenie určené) [12]. Je možné tiež použiť pcap súbor ako

zdroj, z ktorého bude tento démon čítať pakety a ukladať ich do tokov. Najmä kvôli tejto vlastnosti bol tento nástroj použitý pri tejto práci.

3.2.4 Wireshark

Jedná sa o program, ktorý umožňuje odchytávanie paketov zo zadaného sieťového rozhrania alebo zo súboru typu `.pcap`. Program je pod licenciou GPL a je voľne dostupný a použiteľný na súkromné aj komerčné účely. Wireshark je dostupný pre platformy Windows, MAC OS X a systémoch založených na operačnom systéme Linux a je implementovaný v jazykoch C a C++. Pôvodným autorom tohto nástroja je Gerald Combs. Wireshark obsahuje aj grafické užívateľské rozhranie v ktorom sa zobrazujú odchytené pakety jednotlivých komunikácií.

Obsahy paketov sú vyobrazené v binárnej podobe aj v čitateľnej textovej podobe, v ktorej sú vypísané informácie o danom pakete. Pri analýze obsahu každého paketu je možné nahliadnuť do všetkých vrstiev TCP/IP modelu a analyzovať tak jednotlivé protokoly, z ktorých je paket zložený. Wireshark dokáže rozoznať viac ako 850 protokolov a keďže je tento program vyvíjaný pod licenciou open source, tak sa pri každej aktualizácií pridáva podpora nových protokolov [18]. Program taktiež umožňuje používať či už vlastné alebo vstavané filtre pre vyobrazenie požadovanej komunikácie.

Keďže tento program umožňuje nahliadanie do obsahu paketov na rozdiel od vyššie uvedených nástrojov, ktoré pracovali so záznamami tokov, využíval sa pri tejto práci pri detailnejšom pozorovaní komunikácie seedboxu.

3.2.5 Libpcap

Libpcap je knižnica, ktorá poskytuje rozhranie pcap, ktoré slúži pre odchytávanie sieťovej komunikácie. Knižnica libpcap v sebe obsahuje implementáciu rozhrania pre Unixové systémy a verzia tejto knižnice, ktorá je prispôsobená pre platformu Windows sa nazýva WinPcap. Táto knižnica je distribuovaná pod licenciou BSD. Knižnica je implementovaná v jazykoch C a C++ [18].

3.2.6 Tcpdump

Tcpdump je program, ktorý podobne ako Wireshark odchytáva pakety z dopredu zvoleného rozhrania, s tým rozdielom, že oproti Wiresharku nedisponuje grafickým užívateľským rozhraním a teda je ovládaný pomocou príkazového riadka. Je implementovaný v jazyku C. Patrí medzi najstaršie nástroje, ktoré odchytávajú pakety. Tcpdump je distribuovaný pod licenciou BSD a teda je voľne dostupný. K odchytávaniu paketov využíva knižnicu libpcap. Podobne ako Wireshark aj tcpdump vypisuje informácie a obsahy o sledovaných paketoch v čitateľnej textovej podobe alebo v binárnej podobe, ktorá je reprezentovaná v hexadecimálnom formáte. Na niektorých Unixových operačných systémoch je pre odchytávanie paketov nutné mať administrátorské práva od daného systému. Tcpdump umožňuje takisto ako Wireshark aplikovať vlastné filtre pre vypisovanie požadovanej komunikácie [18].

Kapitola 4

Analýza seedboxu

V tejto kapitole sa bude popisovať postup vytvorenia seedboxu, ďalej spôsoby, akými som pre túto prácu získal dáta a sledovanie charakteristík jednotlivých vzoriek seedboxov, ktoré sa v týchto dátach nachádzali.

4.1 Vytvorenie seedboxu

Seedbox, z ktorého boli zbierané dáta, bol vytvorený použitím klienta Transmission, ktorý poskytuje spustenie tohto klienta ako démona. Tento klient podporuje multiplatformovosť. Klient bol spustený na operačnom systéme Ubuntu 14.04 LTS. Na tohto klienta je následne možné sa napojiť cez webové rozhranie zadaním IP adresy a portu, na ktorom démon prijíma prichádzajúce spojenia do webového prehliadača.

Tento démon sa spúšťa príkazom `transmission-daemon start` a pozastavuje sa príkazom `transmission-daemon stop`. Pre spustenie tohoto démona je taktiež nutné mať administrátorské práva od daného systému. Pre uvedenie konfiguračných nastavení k danému seedboxu je nutné vykonať tieto nastavenie do konfiguračného súboru `settings.json`. V tomto konfiguračnom súbore je možné vykonať prístupové nastavenia k danému seedboxu.

Medzi najhlavnejšie nastavenia patria povolenie IP adresy zariadení, ktoré sa môžu na seedbox napájať, číslo portu, na ktorom seedbox prijíma prichádzajúce spojenia a nastavenie autentizačných údajov akými sú prihlasovacie meno a heslo. Pre spôsob nastavenia, akým bude možné daný seedbox nakontaktovať v rámci siete bittorrent, je možné zvoliť klasické nastavenie, ktoré ponúka väčšina dnešných bittorrentových klientov. Súčasťou takéhoto nastavenia môže byť napríklad číslo naslúchajúceho portu, rozsah portov, ktoré sa budú pri komunikácii využívať, maximálny počet peerov, ktorým bude súbor zdieľaný, povolenie rozšírení alebo zapnutie šifrovaného spojenia.

4.2 Zbieranie dát

Pre potreby tohoto projektu som si zvolil a použil tri techniky zberu dát zo sieťovej prevádzky. Prvou technikou je sledovanie tokov a ich následné organizovanie do NetFlow záznamov. Dáta, ktoré obsahujú sledované záznamy mi poskytol pre spracovanie vedúci mojej bakalárskej práce. NetFlow záznamy boli vytvorené sledovaním komunikácie v priestoroch internátnej siete VUT. Keďže sa jedná o citlivé údaje bola nad danými dátami vykonaná anonymizácia IP adries jednotlivých komunikujúcich uzlov. Štatistické údaje o jednotlivých tokoch zostali nezmenené.

Druhou technikou bolo odchytávanie paketov pomocou programov Wireshark a tcpdump zo zariadenia, na ktorom bol spustený seedbox alebo klient, ktorý komunikoval pomocou protokolu BitTorrent. Táto technika získania dát bola použitá hlavne preto, lebo pakety poskytujú detailnejší náhľad na charakteristické rysy jednotlivých komunikácií a keďže je protokol BitTorrent umiestnený v aplikačnej vrstve TCP/IP modelu, bolo do aplikačnej vrstvy týchto paketov nahliadané za použitia programov Wireshark a tcpdump. Týmto sa vykonávala kontrola, či sledovaný uzol, ktorý bol podozrivý na BitTorrent komunikáciu v NetFlow záznamoch skutočne pomocou tohoto protokolu aj komunikoval. Pre túto kontrolu som sledoval výskyt retazca 'Bittorrent protocol', ktorý sa v aplikačnej vrstve vyskytuje pri zasielaní správy typu handshake. Tento typ správy je popísaný v kapitole 2. Pre účely mojej práce som si vytvoril vlastný seedbox, z ktorého boli tieto dáta odchytávané a taktiež som od vedúceho práce obdržal .pcap dáta, na ktorom bola zachytená komunikácia seedboxu v sieti VUT.

Tretia technika zberu dát je principiálne podobná prvej už spomínanej technike. V tomto prípade sa taktiež jedná o sledovanie tokov a vytváranie záznamov NetFlow. Pre tento prípad boli však NetFlow záznamy vytvorené pomocou funkcionality nástroja softflowd, ktorá umožnuje čítať .pcap dáta a následne z týchto dát vytvárať záznamy o tokoch. Postup získanie dát sieťovej premávky je následovný:

- **nfcapd** – Spustenie démona nfcapd, ktorému sa nastaví prepínačom -p ľubovoľne zvolený port, na ktorom bude prijímat NetFlow dáta. V prípade absencie tohoto prepínača sa použije predvolené číslo portu 9995. Prepínačom -l sa zvolí adresár, do ktorého sa budú súbory obsahujúce záznamy o tokoch ukladať. Príklad spustenia nfcapd démona je na obrázku 4.1.

- **softflowd** – Pre čítanie .pcap dát a ukladanie týchto dát do záznamov o tokoch a ich následné odosielanie na démona nfcapd sa program softflowd spustí s prepínačmi -n a -r, kde prepínač -n špecifikuje IP adresu a port koncového zariadenia, na ktorom démon nfcapd bude prijímať dáta. V prípade, že sú oba nástroje spustené na rovnakom zariadení použije sa adresa lokálnej slučky – localhost, daná adresou 127.0.0.1, ktorá odkazuje na práve používané zariadenie. Pri získavaní dát pre tento projekt boli oba nástroje spustené na tom istom zariadení. Prepínačom -r sa špecifikuje .pcap súbor s dátami, z ktorých bude nástroj softflowd vytvárať záznamy o tokoch. Príklad spustenia programu softflowd z príkazového riadka je na obrázku 4.2.

Všetky .pcap dáta, ktoré obsahovali pakety odchytené zo zariadení a ktoré komunikovali pomocou BitTorrent protokolu boli pre účely tejto práce taktiež pretransformované na záznamy NetFlow práve pomocou spomínanej tretej techniky zberu dát.

4.3 Hľadanie charakteristík seedboxu

V nasledujúcich podkapitolách budú rozobraté charakteristiky, ktoré sa pri komunikácií v sieti BitTorrent obvykle vyskytujú. Jednotlivé charakteristiky sú rozoberané na piatich vzorkách, ktoré obsahujú záznamy o tokoch. V záznamoch je odchytená činnosť seedboxu. Jednotlivé vzorky záznamov o tokoch sú uvedené v tabuľke 4.1.

Vzorka *seedbox_1* mi bola poskytnutá vedúcim mojej bakalárskej práce. Ostatné zmienené vzorky boli odchytené mnou na vytvorenom seedboxe v rámci riešenia mojej práce. Následne som vzorky analyzoval použitím nástroja nfdump.

```
mato@mato-ThinkPad-E550:~$ nfcapd -p9995 -l ./TEMP/
Add extension: 2 byte input/output interface index
Add extension: 4 byte input/output interface index
Add extension: 2 byte src/dst AS number
Add extension: 4 byte src/dst AS number
Add extension: 4 byte output bytes
Add extension: 8 byte output bytes
Add extension: NSEL Common block
Add extension: NSEL xlate ports
Add extension: NSEL xlate IPv4 addr
Add extension: NSEL xlate IPv6 addr
Add extension: NSEL ACL ingress/egress acl ID
Add extension: NSEL username
Add extension: NSEL max username
Add extension: NEL Common block
Bound to IPv4 host/IP: any, Port: 9995
Startup.
```

Obr. 4.1: Ukážka spustenia démona nfcapd z príkazového riadku.

```
mato@mato-ThinkPad-E550:~$ softflowd -n 127.0.0.1:9995 -r seedbox.pcap
softflowd v0.9.9 starting data collection
Exporting flows to [127.0.0.1]:9995
```

Obr. 4.2: Ukážka spustenia démona softflowd z príkazového riadku.

4.3.1 Pomer komunikácie nad číslo portu 1024

Predpoklad pre sledovanie uvedenej charakteristiky bol vyvodený na základe faktu, že väčšina P2P aplikácií používa pri svojej komunikácií čísla zdrojového a cieľového portu nad hodnotu 1024. Aplikácie, ktoré komunikujú prostredníctvom protokolu BitTorrent túto vlastnosť spĺňajú tiež. Keďže užívatelia využívajú seedboxy hlavne na odosielanie väčšieho počtu torrentov ostatným peerom, bude na danom zariadení, na ktorom sa seedbox nachádza počet tokov ktorých zdrojové a cieľové číslo portu je nad hodnotou 1024 väčší ako počet ostatných tokov tohoto zariadenia. Tento pomer bude taktiež výrazne väčší na zariadeniach, na ktorých sa seedbox nachádza ako na zariadeniach ostatných bežných užívateľov v sledovanej sieti. Táto vlastnosť je daná samotným chovaním protokolu BitTorrent, kedy sa pri poskytovaní súboru vytvárajú nové spojenia na peera, ktorý daný súbor do siete poskytuje a ktoré sú spôsobené požadovaním jednotlivých blokov súboru od ostatných peerov pre daný torrentový swarm. V prípade seedboxu pre každý poskytovaný torrent vznikajú uvedené formy spojení. V tabuľke 4.2 sú pre každú vzorku sledovaného seedboxu vyobrazené počty tokov, ktorých komunikácia prebiehala nad čísla portu s hodnotou 1024 a pod hodnotu čísla portu 1024. Z výsledkov uvedených v tabuľke je možné pozorovať spomínanú charakteristiku seedboxov.

Hodnoty získaných výsledkov môžu do značnej miery ovplyvňovať veľkosti swarmov jednotlivých torrentov, ktoré seedbox do siete zdieľa a teda čím viac peerov sa v danom swarme nachádza tým väčšie hodnoty môže počet tokov nad číslo portu 1024 nadobúdať. Toto číslo môže byť takisto ovplyvnené časom, počas ktorého bola zo seedboxu odchytávaná komunikácia, keďže počet peerov v swarme sa s časom mení.

Vzorka	Začiatok sledovania	Koniec sledovania	Počet tokov
seedbox_1	08:42:49	10:39:29	237957
seedbox_2	00:22:32	01:22:30	18143
seedbox_3	11:12:26	12:12:25	29775
seedbox_4	19:31:02	20:31:00	17370
seedbox_5	21:57:42	22:57:40	15275

Tabuľka 4.1: Popis analyzovaných vzoriek seedboxov.

Vzorka	Číslo portu >= 1024	Číslo portu < 1024
seedbox_1	213075	9205
seedbox_2	16548	1478
seedbox_3	27160	2103
seedbox_4	16401	1495
seedbox_5	13760	1380

Tabuľka 4.2: Tabuľka počtu tokov ktorých komunikácia bola vedená nad číslo portu 1024 a pod číslo portu 1024.

4.3.2 Port príjmajúci UDP a TCP spojenia

Dnešní bittorrentoví klienti ponúkajú užívateľom možnosť nastaviť preferovaný protokol, pomocou ktorého sa bude počas zdieľania torrentu komunikovať. Toto nastavenie rozhoduje pomocou akého protokolu sa bude prenášať Peer Wire Protocol. Peer Wire Protocol je možné prenášať pomocou dvoch protokolov a to uTP, ktorý je vystavaný nad protokolom UDP a pomocou protokolu TCP. Pri nakontaktovaní peera ktorého bittorrentový klient implementuje túto vlastnosť dochádza k situácií, kedy jeden z peerov používa pre komunikáciu iný protokol ako peer ktorého sa snaží nakontaktovať. V takomto prípade sa obaja peeri musia na použití protokolu dohodnúť. Keďže peer prijíma požiadavky od iných peerov pre začatie komunikácie na špecifickom porte, nastáva na zariadení, ktoré komunikuje protokolom BitTorrent situácia, kedy sú na tento port smerované UDP aj TCP toky. Taktiež sa tento port vyznačuje tým, že počet unikátnych ip adries, ktoré sa na tento port pripájali je väčší ako počet unikátnych ip adries napojených na ostatné porty zariadenia. V pripade, že je na zariadení spustený seedbox, ktorý zdieľa viac torrentov, je viac než pravdepodobné, že sa na tomto zariadení bude nachádzať podobne vyťažený port.

Vzorka	Číslo portu	Počet UDP	Počet TCP	Počet IP adries
seedbox_1	55059	40150	24055	3954
seedbox_2	51413	12022	5043	1876
seedbox_3	51616	20775	6193	2301
seedbox_4	39399	17011	0	1640
seedbox_5	39399	15103	0	2023

Tabuľka 4.3: Tabuľka počtu tokov UDP, TCP a počtu IP adries na naslúchajúci port seedboxu.

Z tabuľky 4.3 si môžeme všimnúť, že pri vzorkách seedbox_4 a seedbox_5 nie sú žiadne TCP spojenia vedené na naslúchajúci port zariadenia, na ktorom sa seedbox nachádza. V

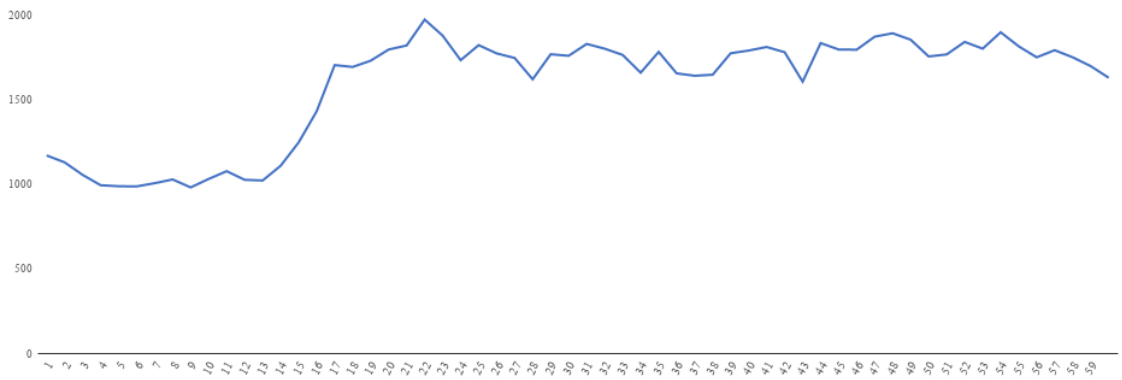
týchto prípadoch všetci peeri v swarmoch pre torrenty, ktoré seedbox zdieľajú mali pravdepodobne v nastaveniach svojich klientov zvolenú komunikáciu pomocou protokolu uTP.

4.3.3 Počet pripojení za časový interval

Pri nadväzovaní komunikácie dochádza v sieti BitTorrent k situácii, kedy sa klient snaží nakontaktovať čo najväčší počet peerov v danom swarme pre získanie jednotlivých blokov sťahovaného súboru. Keďže je problém udržiavať informácie o danom swarme permanentne aktuálne dochádza k situácii, kedy sa v swarme pre daný torrent vyskytujú informácie o peeroch, ktoré sú v swarme nedostupné. Tieto peeri majú buď vypnutého svojho torrentového klienta alebo sa po ceste smerom k danému peerovi mohla vyskytnúť blokácia P2P komunikácie, spôsobená zo strany sieťového providera daného peera. Pri pokuse o nakontaktovanie takýchto peerov je dané spojenie prerušené, ale pokus o nakontaktovanie tohoto peera sa môže ešte niekoľko krát zopakovať. Pri tom, ako seedbox zdieľa do siete viac torrentov, tak dochádza k situácií, kedy je v rámci zariadenia, na ktorom sa seedbox nachádza, veľký počet krátkodobých spojení spôsobený práve týmito neúspešnými pokusmi o nakontaktovanie jednotlivých peerov, o ktorých sú stále v jednotlivých swarmoch vedené informácie.

Vzorka	Priemerný počet tokov
seedbox_1	1634.53
seedbox_2	216.83
seedbox_3	365.91
seedbox_4	209.55
seedbox_5	179.46

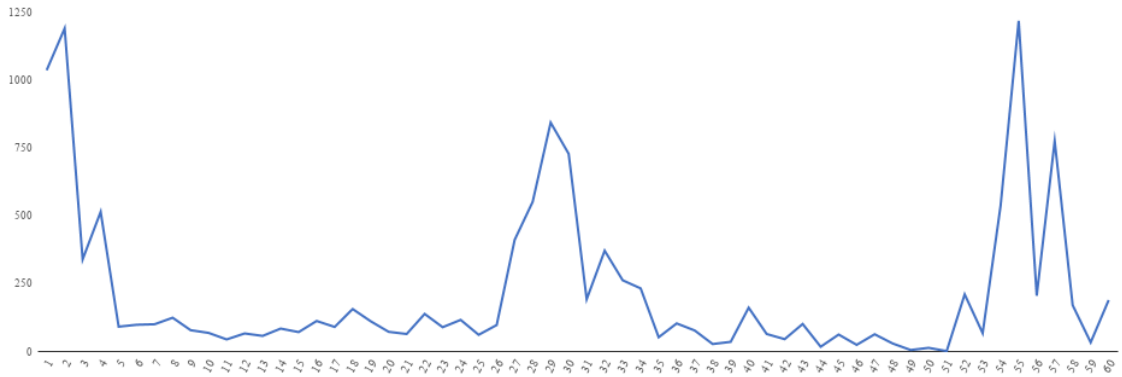
Tabuľka 4.4: Priemerný počet vzniknutých tokov počas minútového intervalu.



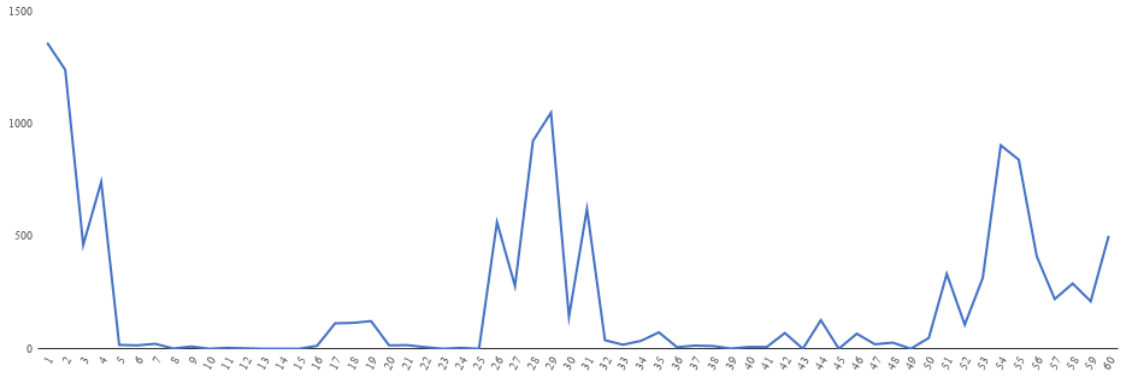
Obr. 4.3: Graf priebehu vzniknutý vrámci hodinového intervalu na vzorke *seedbox_1*.

V tabuľke 4.4 môžeme vidieť priemerný počet vzniknutých a zároveň aj ukončených tokov smerom zo seedboxu a na seedbox počas intervalu jednej minúty. Tieto toky boli smerované nad hodnotu 1024 zdrojového a cieľového portu.

Časový priebeh vzoriek *seedbox_1*, *seedbox_2* a *seedbox_4* sú uvedené na obrázkoch 4.3, 4.4 a 4.5. Os y na daných grafoch reprezentuje počet vzniknutých tokov a os x reprezentuje časovú os jednej hodiny vrámci ktorej boli vzorky sledované. Z obrázkov 4.4 a 4.5 je možné vidieť podobný priebeh grafov pre vzorky *seedbox_2* a *seedbox_4*. Nárasty v počtoch tokov



Obr. 4.4: Graf priebehu vzniknutý vrámci hodinového intervalu na vzorke *seedbox_2*.



Obr. 4.5: Graf priebehu vzniknutý vrámci hodinového intervalu na vzorke *seedbox_4*.

v oboch vzorkách môže byť spôsobené stavom, kedy sa do swarmu pripojilo viacej peerov, ktorí začali mať pre seedovaný torrent záujem a teda sa na daný seedbox začali smerovať požiadavky pre získanie bloku. Pri vzorke *seedbox_1*, ktorý je na obrázku 4.3 nastáva nárast po 14tej minúte, kedy sa počet tokov zvýšil o dvojnásobok predchádzajúceho počtu tokov. V tomto prípade tiež môžeme uvažovať o prípade, kedy sa do swarmu pre zdieľaný torrent pripojili ďalší peeri prípadne bol v túto dobu na seedbox pridaný ďalší torrent ktorý sa začal sťahovať alebo seedovať.

4.3.4 Výskyt komunikácií s trackerom

Pri získavaní informácií o ostatných peeroch v jednotlivých swarmoch, v ktorých sa seedbox podieľa na zdieľaní torrentu používa tento seedbox ako aj každý obvyklý bittorrentový peer signalizačné metódy.

Jednou z týchto metód je použitie trackera, ktorý seedboxu vráti zoznam IP adries a portov peerov pre daný torrentový swarm. V prvých fázach existencie protokolu BitTorrrent bola v špecifikáciách uvedená pre tracker hodnota portu 6969. S postupom času sa však rozsah portov pre trackre zväčšil. V štúdii, ktorá je uvedená v podkapitole 2.2.2 je možné vidieť, že najpoužívanejšie porty trackerov, ktorých hodnota je nad hranicou 1024 majú čísla 6969, 2710 a 1331. Tieto porty samozrejme nie sú jediné a to aké číslo bude tracker používať závisí hlavne na jeho implementácii. Táto štúdia má len poukázať na fakt, že niektoré porty

Vzorka	Port 6969	Port 2710	Port 1331
seedbox_1	0	0	1
seedbox_2	157	79	0
seedbox_3	193	88	0
seedbox_4	181	70	0
seedbox_5	163	57	0

Tabuľka 4.5: Počet tokov, ktoré obsahovali známe porty trackerov.

sú v jednotlivých implementáciách trackerov používané častejšie a teda je pravdepodobné, že sa komunikácie s týmito číslami portov budú vyskytovať aj pri sledovanom seedboxe.

4.3.5 Predvolené porty klientských aplikácií

Peer prijíma požiadavky na komunikáciu od ostatných peerov v danom torrentovom swarme na dopredu špecifikovanom porte. Tento port je možné nastaviť alebo náhodne zvoliť v nastaveniach bittorrentového klienta, ktorý daný peer používa. Ak užívateľ port nenastaví, je danému peerovi pridelený predvolený port, ktorý je nastavený v implementácií torrentového klienta, ktorý peer používa. V prípade bežného užívateľa bittorrentového klienta je dosť možné, že užívateľ používa bittorrentový klient amatérskym spôsobom a teda spustí daný bittorrentový klient bez toho, aby zmenil prednastavený naslúchajúci port. V prípade, keď pozorujeme komunikáciu seedboxu alebo bittorrent komunikáciu všeobecne, môžu nastať situácie, keď viac peerov, ktoré majú rozdielne IP adresy, používajú pre kontakt rovnaké čísla portov. Je pravdepodobné, že prednastavené čísla portov bittorrentových klientov, ktoré majú vysokú popularitu medzi užívateľmi sa budú taktiež vyskytovať medzi číslami portov bittorrentovej komunikácie vedenej na sledovaný seedbox. V štúdii, ktorá je prevzatá zo zdroja [10] a ktorej výsledky sú uvedené v tabuľke 4.6 môžeme pozorovať jednotlivé zastúpenia bittorrentových klientov a ich obľúbenosti medzi užívateľmi.

Klient	Preferovanosť
uTorrent	42.47%
Transmission	21.23%
qBittorrent	18.75%
Deluge	10.7%
Tixati	6.85%

Tabuľka 4.6: Preferovanosť jednotlivých bittorrentových klientov.

Zo štúdie vyplýva, že užívatelia najviac preferujú bittorrentového klienta uTorretnt, ktorého percento preferovanosti má v danej štúdii hodnotu 42%. Druhým najpreferovanejším klientom je v danej štúdií bittorrentový klient Transmission zo zastúpením 21%.

Pri sledovaní počtu výskytov tokov, ktoré obsahovali porty bittorrentových klientov uvedených v štúdií je nutné spomenúť, že klienti uTorrent,Tixati,Deluge majú od inštalácie predvolené porty zvolené náhodne.

V následujúcej štúdií je zistených koľko tokov, ktoré sú smerované na sledované vzorky seedboxov a obsahujú predvolené porty spomínaných bittorrentových klientov.

Keďže väčšinou bittorrentoví klienti uTorrent, Deluge a Tixati majú čísla prednastavených portov náhodne zvolené pri inštalácii, nemôžeme tieto čísla do štúdie zahrnúť. Použité teda budú porty klienta Transmission, ktorého predvolený port je 51413 a klienta qBittor-

Vzorka	Port 8999	Port 51413	Port 6881-6889
seedbox_1	1.59%	4.85%	0.09%
seedbox_2	1.92%	10.87%	3.66%
seedbox_3	1.55%	3.91%	4.98%
seedbox_4	1.50%	6.62%	5.51%
seedbox_5	0.98%	6.60%	4.90%

Tabuľka 4.7: Počet tokov, ktoré obsahovali prednastavené porty bittorrentovych klientov.

rent, ktorého predvolený port je 8999. Taktiež je do štúdie zaradený rozsah portov 6881-6889, ktoré sú uvedené ako naslúchajúce porty v BitTorrent špecifikácií BEP. Výsledky štúdie sú uvedené v tabulke 4.7. Percentuálne hodnoty v tabuľke znamenajú, koľko percent zo všetkých tokov, ktoré obsahovali číslo zdrojového a cieľového portu väčšie ako 1024, v sebe obsahovali daný predvolený port klienta.

Z tabuľky môžeme vidieť, že v každej z testovaných vzoriek bolo zastúpenie spomínaných prednastavených portov torrentových klientov Transmission a qBittorrent a taktiež sa v daných vzorkách vyskytovali aj porty uvedené v špecifikácií k protokolu BitTorrent.

4.3.6 Toky prenášajúce dáta

Zdieľanie torretového suboru má dve fázy. V prvej fáze sa nakontaktuje peer, ktorý má k dispozícií požadovaný blok súboru. Po tom, čo tento peer povolí komunikáciu, je daný blok prenášaný. Najbežnejšia veľkosť jednotlivého bloku je v špecifikácii k protokolu BitTorrent daný ako 256 KiB (vo verzii protokolu BitTorrent pred verziou 3.2 sa používa ako predvolená veľkosť jedného bloku 1 MiB) [4]. Pri sledovaní tokov sú toky, ktoré slúžia na prenos dát charakteristické týmito vlastnosťami:

- Veľkosť dát prenášaných v tomto toku sa pohybuje okolo 256000 bajtov a vyššie. Hodnota 256000 bajtov je zvolená práve podľa spomínanej veľkosti jedného bloku súboru.

- Priemerná veľkosť paketu pre daný tok sa pohybuje okolo 1400 bajtov – táto hodnota bola z daných tokov sledovaná a v podstate sa blíži k maximálnej veľkosti rámca protokolu Ethernet, ktorá je 1500 bajtov. Využitie maximálnej veľkosti Ethernetového rámca sa využíva najmä vtedy, ak po sieti prenášame binárne dáta súborov a teda je potreba čo najväčšiu časť objemu súboru preniesť v jednom pakete.

Pri fáze prenášania dát v protokole BitTorrent sa dá teda predpokladať, že zdrojové a cieľové porty týchto tokov budú väčšie než je hodnota 1024, prenášaná veľkosť sa bude pohybovať okolo 256000 bajtov a priemerná veľkosť jedného paketu sa bude pohybovať okolo hodnoty 1400. Keďže seedboxy, ktoré pozorujeme tiež komunikujú pomocou protokolu BitTorrent mali by byť takéto toky prítomné aj na týchto zariadeniach. V tabuľke 4.8 sa nachádzajú počty takýchto tokov prítomné v jednotlivých vzorkách pozorovaných seedboxov, ktoré boli vyhľadané na základe vyfiltrovania spomínaných parametrov. Aby filtrácia nebola príliš striktná, boli nastavené parametre pre priemernú veľkosť paketu 1000 bajtov a priemerná veľkosť prenesených dát 200000 bajtov.

Vzorka	Počet tokov
seedbox_1	26
seedbox_2	186
seedbox_3	96
seedbox_4	20
seedbox_5	15

Tabuľka 4.8: Počet výskytov tokov, ktoré slúžili na prenos dát v rámci vzorky.

4.3.7 Počet unikátnych IP adries

Veľkosť torrentového swarmu záleží od počtu peerov, ktoré sa v tomto swarme podieľajú na zdieľaní daného súboru. Pri procese kontaktovania vznikajú na zariadenie, ktoré komunikuje protokolom BitTorrent toky, ktoré nesú IP adresy peerov, ktoré majú v danom swarme účasť. Čím väčší je swarm, tým viac je unikátnych IP adries, s ktorými dané zariadenie komunikovalo. Seedbox je určený na zdieľanie väčšieho počtu torrentov a pre každý zdieľaný torrent má účasť v swarme. Dá sa teda očakávať, že počet unikátnych IP adries smerovaných na seedbox bude väčší ako na bežného peera. V tabuľke 4.9 sú pre jednotlivé hodnoty vzoriek seedboxov uvedené počty unikátnych IP adries, ktoré komunikovali nad cieľový a zdrojový port väčší ako 1024 a počty unikátnych IP adries, kde hodnota zdrojového alebo cieľového portu bola pod hodnotou 1024 (napr. klasická komunikácia HTTP alebo FTP). Z výsledkov si môžeme všimnúť, že počet unikátnych IP adries pre porty nad číslo 1024 je značne väčší než počet unikátnych IP adries, ktoré mali aspoň jeden port smerovaný pod číslo 1024. Taktiež sa dá predpokladať, že klasický uzol v sieti, ktorý nekomunikuje pomocou protokolu BitTorrent bude mať väčší počet unikátnych IP adries, ktorých čísla zdrojového alebo cieľového portu boli pod hodnotu 1024 ako počet unikátnych IP adries, ktoré mali zdrojový a cieľový port nad číslom 1024.

Vzorka	Port >= 1024	Port < 1024
seedbox_1	2684	49
seedbox_2	4847	91
seedbox_3	6752	170
seedbox_4	4808	188
seedbox_5	4092	114

Tabuľka 4.9: Počet unikátnych IP adries pripojených nad porty väčšie ako 1024 a menšie ako 1024.

4.3.8 Pomer sťahovania a nahrávania dát

Seedbox je používaný tým spôsobom, že užívateľ nahrá na tento seedbox torrenty od súborov, ktoré chce stiahnuť. Po pridaní týchto torrentov na seedbox sa začne sťahovanie požadovaných súborov od ostatných peerov, ktorý sa nachádzajú v jednotlivých torrentových swarmoch. Ak sa daný súbor stiahne, je začaté jeho zdieľanie do siete ostatným peerom – sledovanie.

Po ukončení sťahovania všetkých súborov nastáva situácia, kedy sú na danom seedboxe seedované všetky súbory. Pri sledovaní sieťovej premávky sa toto správanie prejaví tak, že dané zariadenie, na ktorom sa seedbox nachádza, začne mať v rámci komunikácie, ktorá

Vzorka	**Port >= 1024**	**Port < 1024**
seedbox_1	115 MB	50 MB
seedbox_2	173 MB	2500 MB
seedbox_3	381 MB	22 MB
seedbox_4	124 MB	7 MB
seedbox_5	86 MB	6 MB

Tabuľka 4.10: Veľkosť dát nahrávaných a sťahovaných z jednotlivých seedboxov.

je smerovaná nad zdrojový a cieľový port väčší ako 1024, väčší počet odoslaných bajtov ako počet prijatých bajtov. V tabuľke 4.10 sú uvedené počty stiahnutých a odoslaných bajtov z jednotlivých vzoriek seedboxov. Pri sledovaní tejto charakteristiky tiež závisí na tom v akom stave sa sledované zariadenie počas svojej aktivity nachádzalo. Ak užívateľ počas seedovania torrentov pridá na seedbox ďalší torrent a začne sa jeho sťahovanie, tak sa môže výsledný pomer medzi počtom stiahnutých bajtov a prijatých bajtov zmeniť. Podobná situácia pravdepodobne nastala v prípade vzorky *seedbox_2*, kedy sa počas seedovania torrentov pridal na daný seedbox ďalší torrent, ktorý sa začal sťahovať.

Kapitola 5

Implementácia nástroja

V predchádzajúcich kapitolách bol rozobratý protokol BitTorrent a charakteristiky seedboxov, ktoré pomocou protokolu BitTorrent komunikujú. Na základe zistených poznatkov bol vytvorený nástroj, ktorý napomáha k detekcii seedboxov.

Tento nástroj je zložený zo štyroch častí. Prvý nástroj sa stará o sťahovanie torrentových súborov za pomoci vyhľadávača *Google*. Druhý nástroj slúži na štatistický výpis informácií o priečinku, v ktorom sa stiahnuté torrentové súbory nachádzajú a aktualizuje súbor, ktorý obsahuje používané porty trackerov, novými informáciami o portoch. Tretí nástroj slúži pre získanie portov peerov, ktorý majú účasť v danom torrentovom swarme. Štvrtý nástroj filtruje uzly, ktoré pravdepodobne komunikovali protokolom BitTorrent, vyhodnocuje charakteristiky zaujímavé pre detekovanie seedboxov a ukladá tieto dáta do xml súboru. Všetky vyššie spomínané nástroje sú implementované v skriptovacom jazyku *Python*.

V následujúcich podkapitolách budú jednotlivé nástroje popísané.

5.1 Nástroj pre získavanie torrentových súborov

Skript *downloader.py* využíva k svojej činnosti knižnicu *googlesearch*, ktorá poskytuje API pre prácu s vyhľadávačom Google a knižnicu *libtorrent*, ktorá zas poskytuje API pre prácu s torrentovými súbormi.

Úlohou tohto nástroja je prehľadávať webové stránky na ktorých sa vyskytujú torrentové súbory. Pre vyhľadávanie takýchto stránok je použitá knižnica googlesearch, pomocou ktorej skript zadá vyhľadávací dotaz do vyhľadávača Google a sú vyhľadané a vrátené URL odkazy na indexy webových stránok, v ktorých sa vyskytuje reťazec ”torrent” alebo ”torrents”. Následne sa otvárajú zdrojové kódy týchto indexov a vyhľadávajú sa v nich odkazy, ktoré majú príponu ”.torrent” čo indikuje možný výskyt odkazu práve na torrent súbor. Tento odkaz je následne z webovej stránky vyextrahovaný a stiahnutý. Počas stiahnutia sa v danom torrentovom súbore nájde info hash hodnota a zistí sa, či už sa torrent s danou info hash hodnotou nestiahol. Zoznam stiahnutých info hash hodnôt sa nachádza v súbore `infohash.list`, ktorý sa vytvorí v adresári, kam sa budú torrent súbory sťahovať a po stiahnutí torrent súborov sa tento zoznam aktualizuje na nové info hash hodnoty súborov, ktoré ešte predtým neboli stiahnuté.

Vyhľadávač Google blokuje dotazy, ktoré sú vyhodnotené ako dotazy od robota alebo je dotazov na vyhľadávač v rámci jednej IP adresy viac a v prípade takejto detekcie dotazy z takejto IP adresy časovo obmedzí. Pre tento prípad si nástroj zálohuje aj odkazy na indexy, ktoré pri poslednom spustení prehľadal. Tieto odkazy sa ukladajú do súboru `torrent.cache`

a je automaticky vytváraný pri každom spustení skriptu a ak sa v danom adresári, kam sú ukladané torrentové súbory nachádza, tak sú odkazy v ňom aktualizované na novo získané odkazy.

Skript je možné spustiť s týmito parametrami:

- `--torrent-dir=adresár` – Adresár kam sa sťahované torrent súbory budú ukladať. Pri neuvedení tohto parametru sa torrentové súbory sťahujú do aktuálneho pracovného adresára.
- `--cache-file=súbor` – Súbor s odkazmi na indexy ktoré už boli prehľadané. Je možné zadať vlastný súbor. V prípade nezadania tohto parametru je pre ukladanie odkazov na už stiahnuté torrentové indexy použitý súbor torrent.cache.
- `--links-num=číslo` – Počet odkazov na torrentové indexy z ktorých sa budú sťahovať torrentové súbory. Prednastavený počet je 10. Je dobré neprekračovať hranicu počtu 40 kvôli limitácií vyhľadávača Google.

```
mato@mato-ThinkPad-E550:~/netflowcap$ python2 ../IBT_SKRIPT/src/downloader.py --torrent-dir=../final/
downloading: tails-amd64-3.7.torrent downloaded: 1
downloading: 10x10%202018%20720p%20WEB-HD%20650%20MB%20-%20iExTV.torrent downloaded: 2
downloading: 12.Strong.2018.HDRip.XviD.AC3-EVO%5BEtMovies%5D.torrent downloaded: 3
```

Obr. 5.1: Ukážka výpisu programu počas sťahovania.

Všetky uvedené parametre sú nepovinné. Počas toho ako skript sťahuje a ukladá torrent súbory je záznam o tejto činnosti vypisovaný na štandardný výstup. Výstup nesie informácie o tom aký torrentový súbor bol stiahnutý a koľko torrentových súborov sa doposiaľ stiahlo 5.1. Po ukončení sťahovania je na štandardný výstup vypísaný celkový počet stiahnutých torrentových súborov.

5.2 Nástroj pre analýzu torrentových súborov

Úlohou skriptu *analyzer.py* je extrakcia informácií z torrentových súborov uložených v adresári. Tento skript k svojej činnosti využíva knižnicu *libtorrent* a používa z nej metódy pre dekódovanie torrentových súborov, ktoré sú v kódovaní bencode. Po dekódovaní sa vyextrahujú URL trackerov z kľúčov announce a announce-list. Následne sú z každej tracker URL vyextrahované informácie o protokole, porte a adrese trackera pomocou regulárneho výrazu. Zozbierané informácie o všetkých torrentových súboroch v danom adresári sú následne vypísané na štandardý výstup. Informácie obsahujú počet výskytov jednotlivých portov, ktoré trackere používajú ako svoje naslúchajúce porty, v analyzovanom adresári. Za týmito informáciami sú vypísané informácie o počte využití protokolov, ktorými trackere komunikujú a nakoniec sú vypísané informácie o počte výskytov jednotlivých adries trackerov. Skript taktiež umožnuje vyextrahovať porty, ktoré jednotlivé trackere používali do súboru špecifikovaného užívateľom.

Tento skript obsahuje tri parametre. Všetky parametre sú nepovinné:

- `--torrent-dir=adresár` – Adresár, kam sa sťahované torrent súbory budú ukladať. Pri neuvedení tohto parametru sa torrentové súbory sťahujú do aktuálneho pracovného adresára.

- `--save-ports-file=súbor` – Súbor, do ktorého sa uložia všetky porty používané trackermi.
- `--append` – Pri volbe tohoto prepínača sa do súbora s portami priložia nové porty získané z adresára. Pri absencií tohoto prepínača sa celý súbor s portmi prepíše.

5.3 Nástroj pre získavanie portov od peerov

Úlohou nástroja *getports.py* je čítať torrentové súbory v zadanom adresári a postupne sa zapájať do torrentového swarmu každého torrentu. Po napojení sa do swarmu sú z tohoto swarmu získané informácie o portoch peerov, ktoré sa v tomto swarme momentálne nachádzajú. Informácie o týchto portoch sú ukladané do dopredu špecifikovaného adresára. Skript pre svoju činnosť využíva knižnicu *libtorrent* a jej funkcie, pomocou ktorých je možné sa napojiť na trackre a získavať porty jednotlivých peerov. Skript je možné spustiť s následujúcimi parametrami:

- `--torrent-dir=adresár` – Adresár odkiaľ sa budú čítať jednotlivé torrentové súbory.
- `--save-ports-file=súbor` – Súbor do ktorého sa uložia všetky porty používané peermi v swarme.
- `--number=číslo` – Počet torrentových súborov z ktorých sa budú získavať porty.
- `--retr-time=číslo` – Číslo, ktoré reprezentuje po koľkatích sekundách sa má skončit získavanie portov z jedného torrentového súboru.

5.4 Nástroj pre získavanie informácií z NetFlow záznamov

Nástroj *nfdump_analyzer.py* berie vstup z nástroja nfdump a tento vstup parsuje a následne analyzuje. Pri analýze sa zameriava na charakteristiky, ktoré boli rozoberané v kapitole 4.

Na začiatku skript zavolá nástroj nfdump do ktorého vloží dotaz ktorým sú odfiltrované všetky toky, ktoré neobsahujú protokoly typu UDP a TCP. Ďalej sú tiež odfiltrované toky ktorých IP adresy sú smerované na multicastové adresy a na adresu lokálneho broadcastu. Z výstupu programu nfdump sú získané aj informácie o časovom okne, počas ktorého boli toky zachytené. Toky ktoré ostali na výstupe programu nfdump po odfiltrovaní sú postupne prechádzané a sú z nich vyberané informácie o zdrojovej a cieľovej IP adrese, zdrojovom a cieľovom porte, priemernej veľkosti paketu, type protokolu, veľkosti prenesených dát a časové informácie o trvaní daného toku. Tieto toky sú ukladané do tabuľky IP adries, ktoré boli v tokoch nájdené. Pre každú IP adresu v tejto tabuľke sú vedené dva zoznamy tokov. V prvom zozname sú uchovávané toky, ktoré boli smerované z danej IP adresy a v druhom zozname sú toky, ktoré boli na túto IP adresu smerované. Následne sa pre každú IP adresu preráta pomer medzi tokmi, ktoré boli smerované na zdrojový a cieľový port väčší ako 1024 a ostatnými tokmi danej IP adresy. Taktiež je vypočítaný pomer medzi počtom unikátnych IP adries ktoré boli smerované na zdrojový a cieľový port väčší ako 1024 a ostatnými tokmi tejto IP adresy. Po týchto procesoch sa vykoná filtrácia IP adries, ktoré v sieti dáta len prijímali alebo odosielali a teda nebola komunikácia vedená symetrickým spôsobom.

Pre každú IP adresu sú ďalej zisťované následovné informácie:

- Koľko percent zo všetkých IP adries, ktoré sa vyskytujú v záznamoch o tokoch komunikovali s danou IP adresou.

- Počet portov pre UDP a TCP, ktoré boli použité pre komunikáciu.
- Počet tokov, ktoré boli smerované z danej IP adresy a na danú IP adresu.
- Počet tokov, na ktoré bola smerovaná UDP aj TCP komunikácia.
- Priemerný počet tokov ktoré vznikli aj skončili v určenom časovom intervale.
- Pomer nahraných a stiahnutých dát z danej IP adresy.
- Počet tokov ktoré slúžia na prenos dát.
- Koľko percent portov z priloženého vstupu, ktorý obsahuje získané porty peerov alebo trackerov, sa vyskytlo vrámci komunikácie danej IP adresy.
- Pomer komunikácie smerovanej nad zdrojový a cieľový port väčší ako 1024 oproti ostatnej komunikácií danej IP adresy.
- Pomer medzi počtom UDP tokov a počtom TCP tokov vzniknutých vrámci danej IP adresy.

Po skončení analýzy sú zistené hodnoty charakteristík uložené do dopredu špecifikovaného súboru vo formáte XML.

Vstupné parametre skriptu sú následovné:

- `--read-dir=adresár` – Adresár, odkiaľ sa budú čítať záznamy o tokoch. Spracované budú všetky súbory v danom adresári.
- `--read-file=súbor` – Súbor, ktorý obsahuje záznamy o tokoch ktoré sa budú spracovávať.
- `--peer-ports-file=súbor` – Súbor, ktorý obsahuje porty peerov.
- `--tracker-ports-file=súbor` – Súbor, ktorý obsahuje porty trackerov.
- `--xml-output=súbor` – Súbor, do ktorého sa bude ukladať výstup analýzy. Dáta budú do súboru ukladané vo formáte XML.

K dispozícií sú taktiež parametre ktoré umožňujú meniť proces a výstup analýzy. Jednotlivé parametre je možné kombinovať a tak na finálny výstup súboru dostať len IP adresy s požadovanou hodnotou charakteristiky:

- `--time-interval=číslo` – Číslo udáva po akom intervale bude počítaná hodnota priemerného počtu tokov vzniknutých za zadaný interval.
- `--listening-threshold=číslo` – Nastavuje prah pod akým budú vyhľadávané naslúchajúce porty vrámci danej IP adresy. Ako naslúchajúce porty teda budú brané porty na ktoré sú smerované toky ktorých počet je nad hranicou zadaného prahu.
- `--average-connections=číslo` – Na výstup sa zapíšu výsledky IP adries, ktorých priemerný počet tokov vrámci zadaného intervalu je väčší alebo rovný zadanej číselnej hodnote.
- `--upload-download-ratio=číslo` – Na výstup sa zapíšu výsledky IP adries ktoré majú nahrávací a sťahovací pomer väčší alebo rovný ako je zadaná číselná hodnota.

- `--peers-percentage=číslo` – Na výstup sa zapíšu výsledky IP adries ktoré majú percentuálny výskyt portov peerov vrámci komunikácie danej IP adresy väčší alebo rovný ako je zadaná číselná hodnota.
- `--trackers-percentage=číslo` – Na výstup sa zapíšu výsledky IP adries ktoré majú percentuálny výskyt portov trackerov vrámci komunikácie danej IP adresy väčší alebo rovný ako je zadaná číselná hodnota.
- `--ip-address=ip adresa` – Do XML súboru sa zapíšu výsledky len požadovanej IP adresy.
- `--udp-tcp-ports=číslo` – Na výstup sa zapíšu výsledky IP adries ktoré majú počet portov, na ktoré bola smerovaná UDP aj TCP komunikácia väčší alebo rovný ako je zadaná číselná hodnota.
- `--ip-diversity-ratio=číslo` – Na výstup sa zapíšu výsledky IP adries ktorých pomer unikátnych IP adries nad číslom portu 1024 a IP adries pod týmto portom budú väčšie alebo rovné ako je zadaná číselná hodnota.
- `--large-data-flows=číslo` – Na výstup sa zapíšu výsledky IP adries, ktoré budú obsahovať toky prenášajúce dáta, ktoré sú väčšie alebo rovné ako je zadaná číselná hodnota.
- `--filter-p2p` – Počas analýzy budú odfiltrované IP adresy ktorých toky neobsahujú komunikáciu, ktorá je smerovaná na zdrojový a cieľový port väčší ako 1024.
- `--filter-bittorrent` – Počas analýzy budú striktne odfiltrované IP adresy ktorých toky neobsahujú porty z rozsahu 6881-6889.

Kapitola 6

Testovanie

V tejto kapitole bude rozobraný postup testovania funkčnosti a výkonu navrhnutého nástroja, ktorý napomáha k detekcií seedboxov. Najskôr budú uvedené testovacie vzorky na ktorých testovanie daného nástroja prebiehalo a následne bude uvedený postup a výsledky testovania.

6.1 Testovacie vzorky

Testovanie nástroja prebiehalo na počítači na ktorom boli umiestnené vzorky, ktoré obsahovali záznamy o tokoch. Vzorky boli rozdelené na dve časti. Prvá časť obsahovala vzorky o ktorých bolo známe, že v nich bola zachytená činnosť seedboxu. Sú to vzorky *seedbox_1 - seedbox_5*. Tieto vzorky sú tie isté, ktoré boli analyzované v kapitole 4. Druhá časť obsahovala 12 vzoriek, ktoré v sebe niesli záznamy o tokoch zozbierané z internátnej siete VUT. O týchto vzorkách nebolo známe či v nich bola zachytená aktivita seedboxu. Časové okno zachytávania tokov z internátnej siete VUT bolo od 27.6 do 15.8 teda približne dva mesiace.

6.2 Popis testovania

Nad jednotlivými záznamami, v ktorých bola zaznamenaná seedbox aktivita, bol spustený nástroj nfdump_analyze. Časový interval počas ktorého sa sledujú vzniknuté toky, bol nastavený na 5 minút. Počas doby ktorej nástroj vykonával prepočet charakteristík, boli sledované údaje o vyťaženosti procesora, pamäťových nárokov a tiež doba spracovávania dát týmto nástrojom. Tieto údaje sú vypísané v tabuľke 6.1.

Vzorka	**Rýchlosť spracovania**	**Pamäťová náročnosť**	**Vyťaženie CPU**
seedbox_1	55s	443 MiB	100.0%
seedbox_2	4s	48 MiB	72.0%
seedbox_3	10s	70 MiB	99.9%
seedbox_4	4s	39 MiB	80.0%
seedbox_5	3s	42 MiB	100%

Tabuľka 6.1: Údaje o nárokoch nástroja pri analýze jednotlivých vzoriek.

Najväčší vplyv na dobu vykonávania programu má extrakcia dát z programu nfdump, na ktorú bol použitý regulárny výraz. Taktiež nastavenie intervalu sledovania tokov ovplyv-

ňuje dĺžku vykonávania programu. Pri menšom intervale je výpočet priemernej hodnoty vzniknutých tokov pre danú IP presnejší ale na úkor časovej náročnosti.

Ďalej bol nástroj spustený nad adresárom kde sa nachádzali dáta z internátnej siete VUT. Tento adresár bude ďalej reprezentovaný pod názvom *vutdata*. Kvôli veľkému časovému rozsahu počas ktorého boli dáta zbierané bola hodnota intervalu sledovania tokov nastavená na 360 minút. Údaje o nárokoch počas spracovania týchto dát sú uvedené v tabuľke 6.2. Výsledné hodnoty sú dôsledkom uchovávania veľkého množstva tokov pre každú IP adresu, ktorá sa v daných záznamoch nachádza. Analýza taktiež sleduje pre každú IP adresu priemerný počet vzniknutých tokov v časovom rozmedzí dvoch mesiacov a toto má tiež vplyv na výsledný čas spracovania.

Vzorka	**Rýchlosť spracovania**	**Pamäťová náročnosť**	**Vyťaženie CPU**
vutdata	16m	960 MiB	100.0%

Tabuľka 6.2: Údaje o nárokoch nástroja pri analýze adresára v ktorom sa nachádzajú záznamy o tokoch z internátnej siete VUT.

Kapitola 7

Záver

Cieľom tejto bakalárskej práce bolo zistiť, akými spôsobmi je možné vykonať detekciu seedboxov, ktoré sa vyskytujú vrámci siete BitTorrent a navrhnúť a implementovať nástroj, ktorý by bol schopný túto detekciu realizovať.

Pre zistenie týchto spôsobov bolo nutné vykonať analýzu pre nájdenie charakteristických vlastností seedboxov, pretože ohľadom tejto témy nebola doposiaľ spracovaná odborná literatúra v takej miere, ktorá by sa dostatočne zaoberala touto problematikou. Na základe zistených charakteristík pozorovaných seedboxov bol vytvorený nástroj, ktorý je nadstavbou nástroja nfdump, ktorý tieto charakteristiky sleduje. Taktiež boli k tomuto nástroju implementované ďalšie nástroje, ktoré sa vrámci tejto práce využívali pre zisťovanie a zbieranie informácií o prvkoch v sieti BitTorrent. Informácie, ktoré tieto nástroje produkujú môžu byť prepojené s nástrojom, ktorý sleduje charakteristiky seedboxov a takto zlepšiť proces detekcie.

Vrámci pokračovania v tejto práci by bolo vhodné, ak by sa tento nástroj optimalizoval alebo by sa implementoval do jazyka, ktorý je bližší hardwaru, prípadne by sa mohli výsledné informácie získané pri procese analýzy graficky zobrazovať. Taktiež pri pokračovaní riešenia tejto problematiky odporúčam vykonať detailnejšiu analýzu seedboxov pre zistenie ďalších možných charakteristík, ktorá by naznačovala ich prítomnosť v sieti.

Literatúra

[1] *NFDUMP*. [Online; navštíveno 15.05.2018].
URL http://nfdump.sourceforge.net/

[2] Buford, J. F.; Yu, H.; Lua, E. K.: *P2P Networking and Applications*. Morgan Kaufmann Publishers, 2009, ISBN 978-0-12-374214-8.

[3] Claise, E. B.: *RFC 3954 — Cisco Systems NetFlow Services Export Version 9*. [Online; navštíveno 15.05.2018].
URL https://www.ietf.org/rfc/rfc3954.txt

[4] Cohen, B.: *The BitTorrent Protocol Specification*. [Online; navštíveno 15.05.2018].
URL http://www.bittorrent.org/beps/bep_0003.html

[5] France, R.: *Create your own seedbox to download torrent remotely with Transmission*. [Online; navštíveno 15.05.2018].
URL https://howtoraspberrypi.com/create-own-seedbox-download-torrent-remotely-transmission/

[6] Gong, Y.: *Identifying P2P users using traffic analysis*. [Online; navštíveno 15.05.2018].
URL https://www.symantec.com/connect/articles/identifying-p2p-users-using-traffic-analysis

[7] Gordon, W.: *What is Private BitTorrent Tracker, and Why Should I Use One?* [Online; navštíveno 15.05.2018].
URL https://lifehacker.com/5897095/whats-a-private-bittorrent-tracker-and-why-should-i-use-one

[8] Guide, S.: *What is seedbox?* [Online; navštíveno 15.05.2018].
URL http://seedboxgui.de/guides/what-is-a-seedbox/

[9] Harris, M.: *P2P File Sharing: What is it And is it Legal?* [Online; navštíveno 15.05.2018].
URL https://www.lifewire.com/what-is-p2p-file-sharing-and-is-it-legal-2438571

[10] Henry, A.: *Most Popular BitTorrent Client: uTorrent*. [Online; navštíveno 15.05.2018].
URL https://lifehacker.com/5813348/five-best-bittorrent-applications/1705622513

[11] Jalsovszky, Z.: *Rozpoznání uživatelů P2P sítí na základě analýzy síťového provozu*. Bakalářska práce, VUT FIT v Brně, 2009.

[12] Kuna, L.: *Možnosti anlýzy IP toků v OS Linux*. Vědecká práce, 2009.

[13] Letý, P.: *Detekce peer-to-peer komunikace*. Bakalářska práce, VUT FIT v Brně, 2014.

[14] Lori, R.: *Understanding BitTorrent Through Real Measurements*. Seminar, 2014.

[15] McDowell, M.; Wrisley, B.; Dormann, W.: *Risks of File-Sharing Technology*. [Online; navštíveno 15.05.2018].
URL https://www.us-cert.gov/ncas/tips/ST05-007

[16] Netflow: Netflow — Wikipedia, The Free Encyclopedia. 2018, [Online; navštíveno 15.05.2018].
URL https://en.wikipedia.org/wiki/NetFlow

[17] Pardi, T. Z.: *Mitigate DDoS attacks with P2P*. [Online; navštíveno 15.05.2018].
URL https://streembit.github.io/2017-01-04-DDoS/

[18] Sanders, C.: *Analýza sítí a řešení problémů v programu Wireshark*. Computer Press, 2012, ISBN 978-80-251-3718-5.

[19] Schindler, S.: *Analysis of BitTorrent Trackers and Peers*. Bakalářska práce, Freidrich-Alexander-Universität, 2015.

[20] Warren, T.: *Microsoft to deliver Windows 10 updates using peer-to-peer technology*. [Online; navštíveno 15.05.2018].
URL https://www.theverge.com/2015/3/15/8218215/microsoft-windows-10-updates-p2p

Príloha A

Obsah CD

Priložené CD obsahuje nasledujúce súbory:

- Bakalársku prácu vo formáte PDF
- Zdrojové súbory tejto práce
- Zdrojové súbory nástroja
- Manuál k nástroju
- Testovacie dáta ktoré boli v práci zahrnuté