VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE SÍTÍ P2P POMOCÍ NETFLOW

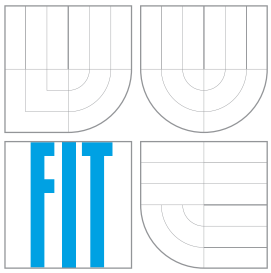
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MICHAL STARIGAZDA

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE SÍTÍ P2P POMOCÍ NETFLOW

DETECTING P2P NETWORKS USING NETFLOW

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MICHAL STARIGAZDA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PETR MATOUŠEK, Ph.D.

BRNO 2013

Abstrakt

Tento dokument se věnuje problematice identifikace P2P sítí v NetFlow záznamech. V práci jsou představeny možné metody identifikace P2P sítí a následné porovnání jejich výhod a nevýhod oproti metodě využívající technologii NetFlow. Tato metoda je založená na popise chování specifické aplikace v rámci statistik o tocích. V teoretické části práce jsou představeny charakteristické vlastnosti P2P systémů. Na základě těchto a dalších specifických vlastností je navrhnut systém detekce vybrané P2P aplikace (BitTorrent). Následně je vytvořen plugin kolektoru NfSen (využívající nástroj NFDUMP) pro detekci protokolu BitTorrent v síťovém provozu. Detekce se zaměřuje na uživatele protokolu BitTorrent s využitím NetFlow záznamů o síťových tocích bez znalosti obsahu datové části paketů.

Abstract

This thesis addresses the problem of identifying P2P networks within NetFlow traces. There are presented different methods for identifying P2P networks. Their advantages and disadvantages are discussed in compare to a method using NetFlow technology. This method is based on traffic characteristics and behaviour of specific application. The theoretical section presents characteristic features of P2P systems in general. A system based on these and more application-specific features is proposed to detect selected P2P application – BitTorrent. Finally, a NfSen plugin for detecting BitTorrent protocol within network traffic (using NFDUMP tools) is implemented. Detection targets users of BitTorrent protocol using NetFlow traces without inspecting the payload of the packets.

Klíčová slova

p2p, NetFlow, BitTorrent, detekce, plugin, NfSen, nfdump

Keywords

p2p, NetFlow, BitTorrent, detection, plugin, NfSen, nfdump

Citace

Michal Starigazda: Detekce sítí P2P pomocí NetFlow, bakalářská práce, Brno, FIT VUT v Brně, 2013

Detekce sítí P2P pomocí NetFlow

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Petra Matouška, Ph.D. Další informace mi poskytli Ing. Petr Špringl a Mgr. Martin Elich.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Michal Starigazda
15. května 2013

Poděkování

Děkuji vedoucímu mé práce panu Ing. Petrovi Matouškovi, Ph.D. za poskytnutou pomoc, konzultace a trpělivost při tvorbě této bakalářské práce. Také bych rád poděkoval Ing. Petrovi Špringlovi a Mgr. Martinovi Elichovi za jejich připomínky, návrhy a čas strávený při konzultacích.

Obsah

Kapitola 1

Úvod

V dnešnej dobe sa peer-to-peer (P2P) siete tešia vysokej popularite a miera využitia P2P aplikácií v posledných rokoch neustále narastá. Práve vďaka tejto popularite a charakteristickému postoju k (intenzívnemu) využitiu šírky pásma sa predpokladá, že prevádzka týchto P2P aplikácií (obzvlášť aplikácií pre zdieľanie súborov) reprezentuje približne 43% až 70% (podľa geografickej polohy) celkovej prevádzky na Internete [16].

Vzhľadom na podiel P2P sietí na celkovej prevádzke je dôležité poskytnúť poskytovateľom internetových služieb (ISP) prostriedky pre detekciu týchto P2P aplikácií. Avšak vzhľadom na skutočnosť, že čísla portov týchto aplikácií nie sú vždy fixné[1], sa komplikuje ich identifikácia, čo vedie k potrebe ďalších možností detekcie na základe charakteristických vlastností prevádzky P2P siete.

Predmetom tejto práce je návrh a overenie funkčnosti systému detekcie (vybraných) P2P sietí s pomocou technológie NetFlow. Práca sa člení do niekoľkých kapitol, ktoré sú organizované následovne. V kapitole 2 sa prehľadovo oboznámime s konceptmi P2P sietí, ich architektúrou a podrobnejšie sa pozrieme na jeden z najpopulárnejších P2P systémov – BitTorrent (princípy a charakteristické vlastnosti tohto protokolu). V následujúcej kapitole (kapitola 3) budú predstavené metódy detekcie P2P aplikácií. V rámci tejto časti bude opäť podrobnejšie predstavená technológia NetFlow, s ktorou je úzko spojená táto práca. Zároveň budú prezentované aj nástroje pre prácu s NetFlow – NFDUMP a NfSen. Kapitolu 4 tvorí návrh systému pre detekciu protokolu BitTorrent s pomocou sieťových tokov. Samotný návrh systému vychádza z analýzy charakteristických vlastností protokolu BitTorrent a možnosti ich identifikácie v rámci NetFlow štatistik. Na základe analýzy je vytvorený popis chovania bittorrentovej aplikácie v sieťovej prevádzke. Pri vytváraní popisu chovania sú použité aj faktory navrhnuté v predchádzajúcich prácach zaoberajúcich sa detekciou P2P prevádzky bez využitia znalosti o obsahu paketov v sieťových tokoch. Tieto faktory sú zároveň konfrontované s vytvorenou kolekciou NetFlow záznamov obsahujúcich bittorrentovú komunikáciu. Dôvodom je overenie ich platnosti aj pre aktuálnu verziu protokolu BitTorrent a použiteľnosti v systéme detekcie tohto protokolu. Navrhnutý systém sa zameriava na detekciu koncových užívateľov používajúcich bittorrentové aplikácie.

Praktickú časť práce tvorí implementácia systému pre detekciu protokolu BitTorrent vo forme pluginu nástroja NfSen na základe predchádzajúceho návrhu, ktorá je popísaná v kapitole 5. Posledná kapitola (kapitola 6) obsahuje výsledky testovania funkčnosti implementovaného systému detekcie. Cieľom tohto systému je mimo samotnej detekcie aj schopnosť spracovania dát kolektoru v reálnom čase (čas spracovania vstupu by nemalo prekročiť päť minút). V poslednej kapitole sú zahrnuté aj návrhy pre ďalší postup práce.

[1] v súčastnosti väčšina P2P aplikácií používa dynamické čísla portov

Kapitola 2

Siete peer-to-peer

V bežných sieťach typu klient-server rozlišujeme dva základne typy uzlov – centrálny server (poskytujúci určitú službu) a klientov (bežní užívatelia), ktorí sa na tento server pripájajú. Iniciátorom spojenia je klient a typicky sa jedna o komunikáciu žiadosť-odpoveď. Výhodou tejto metódy je jednoduchosť implementácie a dostupnosť súborov na sieti. Problém nastáva, pokiaľ je súbor, ktorý server poskytuje klientom, príliš veľký, alebo je príliš žiadaný. Spojenie so serverom v tomto prípade zaberá značnú časť šírky pásma a zároveň sa problém prejaví aj na zaťaženosti zdrojov serveru (súbor musí byť odoslaný všetkým klientom, ktorí ho požadujú). Ďalším problémom je práve opačná situácia, keď je nepopulárny súbor odstránený zo serveru a jeho získanie sa týmto komplikuje alebo stáva nemožným. Tieto problémy riešia P2P siete svojím prístupom k distribúcií dát a ich replikácií po sieti na požiadanie.

Narozdiel od sietí na princípe klient-server, v peer-to-peer (P2P) sieťach nerozlišujeme rolu serveru a klienta. V P2P sa jedná o spojenia medzi dvoma užívateľmi (uzlami), ktorí sú si navzájom rovný. Uzol (*peer*) zabezpečuje jak serverové tak klientské služby. Tento spôsob komunikácie využívajú rôzne aplikácie pre prenos súborov, audio a video hovorov a zdieľanie dát. Princíp sa využíva aj v hrách s viacerými hráčmi (tzv. *multiplayer* hry).
Už ako bolo spomínané, model P2P siete rieší problémy sietí klient-server tým, že uzly P2P siete sťahujú dáta rôzne medzi sebou (od jedného alebo viacerých zdrojov). Teda uzol môže v komunikácií vystupovať ako server aj ako klient zároveň. Organizácia uzlov v sieti je väčšinou plochá (tzv. flat-model) - každý užívateľ komunikuje s iným uzlom bez pomoci nejakej centrálnej autority. Avšak táto výhoda P2P sietí so sebou prináša aj problémy ohľadom správy a zabezpečenia dát prenášaných v rámci siete. Zúčastnení užívatelia P2P systému sa stávajú náchylnejší k rôznym hrozbám a porušeniu zabezpečenia.

Priebeh sťahovania v P2P je možné rozdeliť na dve fázy, resp. môžeme hovoriť o klasifikácia P2P prevádzky [20]:

- **Signalizácia** – zahŕňa vytvorenie spojenia a vyhľadávanie.
- **Dátový prenos** – kontaktovanie a sťahovanie príslušného súboru od konkrétneho uzlu.

Prvé P2P systémy (napr. Gnutella) používali k propagácií požiadavku medzi všetkých P2P užívateľov techniku kontrolovanej záplavy (*controlled flooding*) [4], čo viedlo k problémom so zaťažením šírky pásma. Novšie systémy (FastTrack, DirectConnect, novšie verzie Gnutelly) zasielajú požiadavky cielenejšie, šírka prenosového pásma sa behom fázy signalizácie využíva efektívnejšie.

Veľkosti súborov v typických systémoch pre zdieľanie dát sa v minulosti pohybovali v rozsahu cca 4-5 MB (formát MP3) až 700 MB (hodina a pol videa alebo CD image). Dnes sú už bežné aj súbory o veľkosti niekoľkých GB (DVD image). Pri prenose takýchto objemov dát je podiel signalizácie (predstavuje niekoľko stoviek bytov) zanedbateľný. Z toho vyplýva, že pre detekciu P2P sietí je zaujímavejšia časť dátového prenosu, ktorá má najväčší podiel na celkovej P2P prevádzke.

P2P aplikácie sú vo väčšine prípadov navrhované a používané pre zdieľanie súborov. V ďalšom texte tejto práce budeme uvažovať práve tento typ P2P aplikácií.

2.1 Architektúra P2P sietí

P2P systémy môžeme klasifikovať podľa úrovne centralizácie a ich sieťovej štruktúry [18].

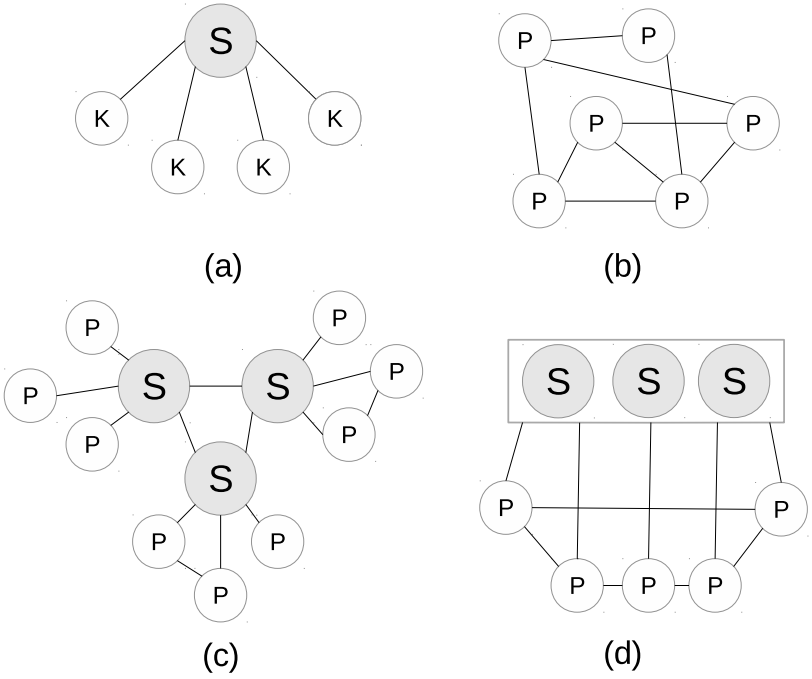
2.1.1 Klasifikácia podľa úrovne centralizácie

Rozdelenie podľa úrovne centralizácie je založené na miere účasti resp. využitia modelu klient-server v rámci zaistenia kooperácie medzi uzlami. Architektúry P2P sietí s rôznym stupňom centralizácie (popísané nižšie) sú znázornené na obrázku 2.1.

- V **decentralizovaných** architektúrach majú všetky uzly rovnaké povinnosti bez ohľadu na ich kapacitu, lokalizáciu a zdroje. Vykonávajú rovnaké úkony, v ktorých vystupujú ako server a klient zároveň. Neexistuje centrálna koordinácia. Pre užívateľov v týchto sieťach sa používa výraz *servent* (SERVer a kliENT). Táto architektúra je pomerne málo rozšírená kvôli svojej neefektívnosti – správy môžu cestovať medzi veľkým množstvom uzlov pred dosiahnutím cieľa (uzol poskytujúci požadovaný súbor).

- V **čiastočne centralizovaných** architektúrach zastávajú niektoré uzly viac povinnosti ako iné, jedná sa o lokálne servery – *super uzly* (*supernodes*), ktoré spravujú súbory zdieľané lokálnymi uzlami P2P siete a umožňujú komunikácia s ďalšími servermi. Táto architektúra vytvára hierarchiu (tzv. two-level) s lepšou výkonnosťou a škálovateľnosťou ako u decentralizovanej architektúry. Je použitá v systémoch ako FastTrack alebo Gnutella. Je nutno zdôrazniť, že servery (*super uzly*) sú volené dynamicky a neposkytujú tzv. *single point of failure* (pri výpadku je zvolený iný uzol, ktorý zastane funkciu serveru).

- **Hybridné (hybridne decentralizované)** architektúry sa vyznačujú prítomnosťou serveru (resp. niekoľkých serverov), ktorý zaisťuje koordináciu medzi uzlami a poskytuje ďalšie služby, napr. vyhľadávanie súboru. Typicky sa v P2P systémoch využívajúcich túto architektúru (napr. eDonkey) vyskytuje väčší počet serverov, distribuovaných podľa geografickej polohy a navzájom prepojených. Rozdiel medzi čiastočne centralizovanými systémom a hybridným systémom spočíva v softvérovom vybavení. *Super uzly*, aj po tom čo sú dynamický zvolené, sú stále zároveň aj bežnými užívateľmi – nevykonávajú úlohu serveru na „plný úväzok“. V hybridných systémoch sa líši klientský softvér a softvér serveru (server zaisťuje kooperáciu medzi uzlami – funguje na „plný úväzok“). Nevýhodou tohto prístupu je nižšia flexibilita a odolnosť voči chybám.



Obrázok 2.1: Rôzne stupne centralizácie. (a) Klient-server model. (b) Decentralizovaná architektúra. (c) Čiastočne centralizovaná architektúra. (d) Hybridná architektúra.

2.1.2 Klasifikácia podľa sieťovej štruktúry

Pre odlíšenie operácií P2P protokolu na aplikačnej vrstve od chovania siete na nižších vrstvách[1], sa prepojenia medzi uzlami v P2P sieti označujú ako P2P vrstva [4].
Topológia tejto vrstvy je dynamická, reaguje na časté príchody a odchody uzlov do a zo siete. Môže sa jednať o ad-hoc topológiu, nezávislú na fyzickej štruktúre siete a obsahu užívateľských staníc, alebo môže byť organizovaná pre efektívnejšie využitie siete alebo rýchlejšie vyhľadávanie. Pri existencii asociácie topológie s obsahom uzlu hovoríme o štrukturovanom P2P systéme. Podľa toho môžeme rozdeliť P2P systémy do troch skupín.

- **Neštrukturované** – umiestnenie dát je úplne nezávislé na topológii P2P vrstvy. Bez znalosti umiestnenia súboru prebieha jeho vyhľadávanie náhodne, požiadavkami na niekoľko ďalších *serventov* či sa u nich nenachádza požadovaný súbor. Títo serventi sa môžu ďalej pýtať svojich susedov, atď. Vo všeobecnosti majú neštrukturované siete problémy so slabou výkonnosťou vyhľadávania požadovaných súborov, škálovateľnosťou a neefektívnym využitím siete. Napriek tomu sú tieto siete pomerne rozšírené. Problémy škálovateľnosti a výkonnosti je možné riešiť využitím čiastočne centralizovaného alebo hybridne decentralizovaného modelu. Stále sa jedná o náhodne vyhľadávanie, ale už iba na úrovni serveru (resp. super uzlu).

- **Štrukturované** – sa objavujú hlavne v akademickej oblasti. V týchto systémoch je topológia úzko spojená s obsahom staníc resp. obsah staníc je spätý s topológiou. Súbory (alebo ukazovatele na súbory) sú uložené v špecifických lokáciách v P2P systéme a existuje prostriedok pre mapovanie identifikátorov súborov na ich umiestnenie. Využitím distribuovanej smerovacej tabuľky (väčšinou hašovacie tabuľky – DHT)

[1] vrstvy ISO/OSI modelu

je možné smerovať otázky na príslušne stanice (užívateľa) s oveľa vyššou efektivitou ako u predchádzajúceho typu. Nevýhodou je náročnosť udržiavania smerovacej tabuľky vzhľadom na časté príchody a odchody uzlov v rámci systému a mapovanie kľúčových slov na unikátne identifikátory súborov.

- **Voľne štrukturované** – kompromis medzi neštrukturovanými a štrukturovanými sieťami. Existuje mapovanie medzi umiestnením súboru a topológiou, ale nie je úplne špecifikované a môže dôjsť k chybe vyhľadávania.

2.2 Význam detekcie P2P sietí

Popularita rôznych P2P aplikácií je spojená s ich významným vplyvom na výkonnosť siete (dokážu zaťažiť sieť stovkami až tisíckami spojení a zaplaviť linku veľkým počtom paketov [17]). Niektoré odhady stavajú podiel P2P prevádzky až na 50% celkovej prevádzky Internetu [4]. Neočakáva sa, že by sa tento trend v blízkej budúcnosti zmenil. Dopad P2P prevádzky sa odráža na vyťaženosti kapacity liniek navrhnutých na asymetrickom predpoklade o prevádzke v sieti (užívatelia sťahujú viac ako odosielajú), zatiaľ čo P2P prevádzka je symetrická. To vyplýva z faktu, že uzol P2P vrstvy môže vystupovať ako server a aj ako klient.

Potreba detekcie P2P prevádzky je teda daná potrebou poskytovateľov služby internetu (ISP) klasifikovať prevádzku. Na základe výsledkov klasifikácie a vytvorení predstavy (modelu) o sieti môžu ISP implementovať technológie umožňujúce zaručenie korektnosti výkonnosti pre webovú prevádzku a aplikácie bežiace v reálnom čase.

Ďalším dôležitým hľadiskom týkajúcim sa potreby detekcie je legálnosť sťahovaného obsahu prostredníctvom P2P aplikácií zameraných práve na tento účel (systémy pre zdieľanie súborov). Teda samotné P2P aplikácie nie sú nelegálne ale distribúcia súborov (filmy, hudba, hry, knihy,...) ich prostredníctvom porušuje autorské práva. Detekcia je v záujme jednak spoločností pre ochranu autorských práv a jednak opäť v záujme ISP, keďže v minulosti sa objavili súdne procesy, v ktorých boli obvinený zo zodpovednosti za toto nelegálne šírenie súborov. Čo podporuje ich záujem na detekcií a obmedzení P2P užívateľov.

Na druhú stranu si treba uvedomiť, že pre užívateľa sa ako lepšia varianta môže javiť poskytovateľ, ktorý neobmedzuje svoje služby úplným blokovaním P2P aplikácií alebo obmedzovaním prenosovej rýchlosti. Napriek tomu sa mnoho poskytovateľov uberá touto cestou, čím sa vyzdvihuje dôležitosť zdokonaľovania technik detekcie P2P aplikácií.

Reakciou na zdokonaľovanie technik detekcie P2P prevádzky je vývoj metód, ktoré túto detekciu znemožnia zo strany P2P protokolov a aplikácií. Podstatnou zložkou tejto snahy je vývoj metód pre šifrovanie prevádzky P2P sietí, čo vedie k následujúcim efektom:

- Spomínané znemožnenie detekcie zo strany ISP.
- Ochrana proti odposluchávaniu a tzv. útokom „človek uprostred“ (*Man in the middle*).
- Prekonanie cenzúry a znemožnenie detekcie nelegálnej distribúcie dát.
- Anonymita (šifrovanie toto umožňuje iba na pomerne nízkej úrovni)[2].

[2] Anonymita P2P systémov sa predovšetkým týka troch aspektov: skrytia identity poskytovateľa informácie, skrytia identity príjemcu a znemožnenia ich spárovanie. Typické riešenie spočíva v použití siete prostredníkov (friend-to-friend siete a s nimi súvisiace pojmy, napr. *onion routing* – cibuľové smerovanie, viď http://www.lupa.cz/clanky/anonymni-peer-to-peer-site/).

2.3 BitTorrent

2.3.1 Popis protokolu

BitTorrent je protokol pre zdieľanie a distribúciu súborov v sieti Internet. Protokol je založený na princípe P2P [22] a v súčastnosti to je jeden z najpopulárnejších, a teda aj najpoužívanejších P2P protokolov. Využíva prístup „*tit for tat*“, kde sa uzol sťahujúci súbor musí podieľať na jeho odosielaní, aby bol dodržaný koncept zdieľania zdrojov v rámci P2P siete. Hlavnou výhodou BitTorrentu je práve schopnosť užívateľov podieľať sa na zdieľaní súboru bez toho, aby vlastnili úplnú kópiu. Protokol bol navrhnutý Bramom Cohenom v roku 2001, v rovnakom roku bol aj vydaný.

BitTorrent umožňuje identifikáciu obsahu pomocou URL a je navrhnutý pre hladkú integráciu s webom [7]. Výhoda oproti čistému HTTP sa prejavuje pri konkurentnom sťahovaní rovnakého súboru, kde klienti môžu odosielať súbor medzi sebou navzájom, čo umožňuje zdroju súboru poskytovať podporu veľkému množstvu klientov pri relatívne malom zvýšení zaťaženia. Je alternatívou klient-server architektúry, umožňujúcou redukciu dopadu distribúcie veľkého množstva dát na server a sieť samotnú. Vďaka svojím princípom dokáže BitTorrent pracovať efektívne aj pri nízkej šírke prenosového pásma a zároveň šetrí hardvérové zdroje. Základná terminológia protokolu BitTorrent:

- **Torrent (.torrent súbor)** – jedna sa o jeden súbor, ktorý obsahuje metadáta všetkých súborov, ktoré je prostredníctvom neho možné sťahovať. Obsahuje názvy súborov, ich veľkosť, kontrolné súčty, počet segmentov, do ktorých je konkrétny súbor rozdelený a veľkosť segmentu, adresu trackeru. Týmto pojmom sa označujú aj dáta stiahnuté pomocou protokolu BitTorrent. Ukážka obsahu `.torrent` súboru (kódovanie *Bencode*[3]):
  `d8:announce41:http://bttracker.debian.org:6969/announce7:comment35:"Debian CD from cdimage.debian.org"13:creation datei1361635092e4:info d6:lengthi679192576e4:name36:debian-6.0.7-ia64-xfce+lxde-CD-1.iso12:piece lengthi524288...`

- **Tracker** – server spravujúci `.torrent` súbory, ktorý zároveň koordinuje spoluprácu klientov na sťahovaní súboru (informuje klientov o ostatných klientoch). Komunikácia medzi klientom a trackerom prebieha cez protokol HTTP, resp. HTTPS. Cez tracker sa neprenášajú žiadne dáta torrentu, využíva sa iba k signalizácií, viď sekciu 2. V súčasnosti existuje mnoho takýchto serverov (verejných aj privátnych).

- **Peer** – užívateľský uzol, klientska inštancia BitTorrentu.

- **Leech (pijavica)** – užívateľ s nekompletnou kópiou sťahovaného súboru. Po úspešnej kompletizácií sa stáva seedom (resp. seederom).

- **Seed (semeno, osivo)** – užívateľský uzol, ktorý obsahuje kompletnú kópiu sťahovaného súboru. Tieto dáta poskytuje ostatným uzlom behom sťahovania. Uzol, ktorý má stiahnutý kompletný súbor (všetky jeho časti), sa stávam seedom. Počtom seedov je charakterizované „zdravie“ torrentu.

- **Swarm (roj)** – označenie skupiny uzlov zdieľajúcich dáta patriace do rovnakého torrentu. Úlohou protokolu BitTorrent je optimalizácia výmeny dat medzi účastníkmi

[3] https://wiki.theory.org/BitTorrentSpecification#Bencoding

vo swarme. S veľkosťou swarmu sa zvyšuje rýchlosť sťahovania (narozdiel od architektúry klient-server, kde sa pri zvýšení počtu klientov rýchlosť znižuje).

2.3.2 Spôsob šírenia dát

Sťahovanie prostredníctvom protokolu BitTorrent (štruktúra je znázornená na obrázku 2.2) prebieha typicky v následujúcich krokoch (podrobnosti možno dohľadať v špecifikácií protokolu [7]):

- Prvým krokom je vyhľadanie a stiahnutie súboru s príponou `.torrent` (obsahuje informácie potrebné k stiahnutiu príslušného súboru prostredníctvom protokolu BitTorrent) z bežného webového serveru (predpokladá sa existencia takéhoto serveru obsahujúceho metadátové súbory). Po spustení (otvorení) `.torrent` súboru prostredníctvom bittorrentového klienta nasleduje ďalší krok komunikácie.

- Komunikácia s trackerom. Tracker prijíma požiadavky od uzlov BitTorrentu a koordinuje ich spoluprácu pri sťahovaní súboru (je zodpovedný za to, aby sa účastníci sťahovania navzájom našli). Komunikácia využíva typicky protokol HTTP, kde klient zaslaním požiadavku GET oznámi trackeru svoju existenciu a poskytne IP adresu a číslo portu, na ktorom naslúcha spojeniam od ďalších užívateľov. Klient môže pre oznámenie svojej existencie (a priebežné informovanie o stave sťahovania) použiť aj *UDP tracker protocol* [19]. Tracker odpovedá zoznamom uzlov (IP adresou a portom, na ktorom naslúchajú), ktorí sa podieľajú na distribúcií požadovaného súboru.
  Alternatívou k vyhľadávaniu uzlov pomocou trackeru je mechanizmus DHT (distribuovaná hašovacia tabuľka – *distributed hash table*) [15]. V takomto systéme sa nespolieha na prítomnosť trackeru. Bittorrentový klienti využívajú DHT k uloženiu informácií o uzloch, čím sa umožní sťahovanie súboru bez asistencie trackeru. Klient využívajúci DHT musí najprv kontaktovať tzv. zavádzací uzol. Týmto sa pripojí do DHT swarmu a od zavádzacieho uzlu získa niekoľko prvých uzlov k zaplneniu DHT. Následne sa počiatočné uzly kontaktujú s využitím UDP protokolu a zaháji sa vyhľadávanie ďalších, ktorí zdieľajú požadovaný súbor. Pri nájdení takéhoto uzlu sa vytvorí TCP spojenie medzi ním a bittorrentovým klientom pre prenos dát.

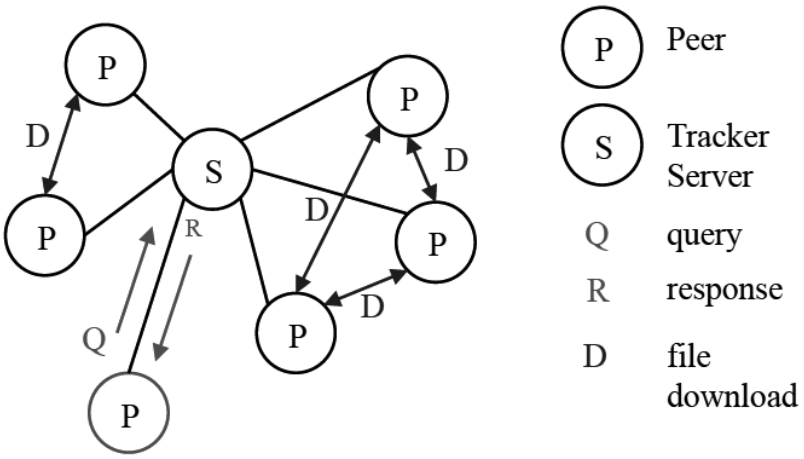
- Klienti sa naďalej periodicky hlásia trackeru a oznamujú mu stav sťahovania. Zároveň prebieha komunikácia medzi nimi navzájom – sú priamo spojený. V tejto fáze sa využíva *BitTorrent peer protocol*, ktorý operuje nad TCP alebo UDP. Typicky sa ku kontaktovaniu uzlov používa protokol UDP a pre samotný prenos dát je použitý protokol TCP. Avšak UDP môže byť použitý aj v rámci celej komunikácie (ako aj TCP), teda aj samotného prenosu dát. To je možné pri použití protokolu uTP[4].

- Výmena dat medzi uzlami BitTorrentu prebieha sťahovaním jednotlivých časti súboru (blokov). Súbor (dáta) je rozdelený na rovnako veľké časti, typicky 256kB, 512kB a 1MB (preferovaná veľkosť u verzie protokolu 3.2). Pre každú časť je vytvorený kontrolný súčet prostredníctvom SHA1 algoritmu. Pri sťahovaní sa využíva metóda

[4] uTorrent Transport Protocol (`http://www.bittorrent.org/beps/bep_0029.html`) – open source protokol bežiaci nad UDP, ktorý zefektívňuje využitie šírky pásma a redukuje problémy vznikajúce pri prenose. Cieľom je maximalizácia priepustnosti siete a zároveň znižovanie odozvy a zahltenia siete. Je navrhnutý tak, aby sám spomaľoval prenos pokiaľ je sieť preťažená, a tak predchádzal problémom. uTP je user-friendly k sieti, keďže využíva celú šírku pásma iba v prípade, že o ňu nestoja žiadne iné aplikácie. Vo výsledku použitie uTP zrýchľuje sťahovanie pre užívateľov a znižuje vyťaženosť siete (pozitívum pre ISP).

„*rarest-first*“ (najvzácnejšie prvé), ktorá zvýšením redundancie najmenej sťahovaných časti dát redukuje fatálnosť následkov výpadku seedu (užívatelia nie sú schopný dokončiť kompletizáciu súbor).

V zmysle popísanej klasifikácie P2P sietí a popisu protokolu BitTorrent by sa P2P systém založený nad týmto protokolom dal zaradiť medzi P2P siete postavené nad hybridnou architektúrou (prítomnosť trackeru), ktorý zároveň využíva aj prvky decentralizovanej architektúry (DHT mechanizmus lokalizácie uzlov). Vzhľadom na sieťovú štruktúru sa jedná o voľne štrukturovaný P2P systém.



Obrázok 2.2: Štruktúra BitTorrentu (prevzaté z [14]).

2.3.3 Využitie protokolu BitTorrent

Tento protokol predstavuje nenáročnú alternatívu pre zdieľanie a distribúciu súborov, resp. produktov, jednotlivcov alebo organizácií. Dôvodom je nenáročnosť na požadovaný hardvér (server pripojený do siete poskytujúci služby *trackeru*) a priepustnosť dátového spojenia. Následne je už potrebné iba vytvoriť a zverejniť metadátový súbor torrentu (`.torrent`).

Prakticky sa protokol BitTorrent bežne využíva pre prenos najrôznejších dátových súborov – hudby, filmov, hier a aktualizácií (napr. Blizzard Downloader od firmy Blizzard Entertaiment).

Avšak vo všeobecnosti sa dá o BitTorrente hovoriť ako o „*release system*“. Pri vydaní populárneho súboru, napr. rôzne linuxové distribúcie, existuje široká skupina užívateľov, ktorá sa zaujíma o tento súbor (*hot-file*). S počtom užívateľov narastá veľkosť *swarmu*, čo sa odráža na vysokej rýchlosti sťahovania. Po nejakej dobe ale počet súčasných sťahovaní klesá (je to dané poklesom záujmu o daný súbor). To sa prejaví poklesom rýchlosti sťahovania.

Kapitola 3

Detekcia P2P sietí

Detekcia P2P sietí spočíva v odlíšení príslušných spojení od zvyšku prevádzky. Na základe výsledkov je ďalej možné identifikovať konkrétnych užívateľov P2P systému, resp. je možné postupovať opačne – detekovať užívateľov a následne príslušnú P2P prevádzku.

V tejto sekcií si predstavíme niekoľko prístupov k identifikácií sietí P2P, kde sa zameriame na prezentáciu ich výhod a nevýhod oproti prístupu, ktorý využíva technológiu NetFlow, pretože náš detektor bude založený práve na využití tejto technológie.

3.1 Detekcia na základe čísiel portov

Väčšina P2P aplikácií pre zdieľanie súborov používa špecifické čísla portov. Vzdialený užívateľ zasiela požiadavok na tento port a odpoveď sa odosiela cez vytvorené spojenie. V opačnom prípade, pokiaľ jeden P2P uzol zasiela požiadavok inému, tak prvý použije náhodne číslo portu na lokálnom stroji a pripája sa k známemu číslu portu na vzdialenom.

Detekcia na základe čísiel portov je jednoduchá metóda bežne používaná v minulosti. K samotnej detekcií P2P spojenia a jeho následnému zablokovaniu postačuje iba náhľad na čísla portov v zachytenom TCP alebo UDP pakete z príslušného spojenia. Problémom je, že dnešné P2P aplikáciu už často nepoužívajú predvolené čísla portov. Využívajú náhodne generované čísla portov, aby sa zabránilo detekcií. Niektoré programy užívateľov priamo vyzývajú, aby nepoužívali predvolené čísla portov[1]. Ďalšou metódou, ktorá má zabrániť detekcii P2P prevádzky, je maskovanie s využitím čísiel portov iných široko využívaných služieb ako napr. HTTP (80), HTTPS (443) alebo FTP (21).

Napriek tomu, že táto metóda vykazuje nedostatky pri detekcií P2P spojení, stále je použiteľná pri klasifikácií tradičných služieb a protokolov ako DNS, HTTP, FTP, ICMP. Detekcia týchto a ďalších služieb využívajúcich tzv. „*well known*“ čísla portov vykazuje vysokú úspešnosť.

Je nutné poznamenať, že aj pri použití náhodných portov môže byť uzol P2P siete detekovaný tradičnou metódou na základe špecifických portov. Pri použití náhodného čísla portu sa môže spojiť s inými uzlom, ktorý naslúcha na predvolenom porte. Takže hoci táto metóda samotná nevykazuje najlepšie výsledky pri detekcií P2P prevádzky, môže byť použitá v kombinácií s inou metódou (napr. analýzou sieťových tokov a popisom chovania) pre zvýšenie jej presnosti.

[1] http://wiki.vuze.com/w/Select_port_for_Vuze#Which_ports_Vuze_uses_by_default.3F

3.2 Aktívne preliezanie

Aktívne preliezanie (*active crawling*) [18] je metóda skúmania P2P systémov aktívnou účasťou v tomto systéme. Crawler je modifikovaný klient vybraného P2P systému, ktorý sa doň pripája bežnou cestou a získava informácie (IP adresy, čísla portov, ďalšie metadáta získane prostredníctvom P2P protokolu) o čo najväčšom počte uzlov. Na základe tohto prístupu nedochádza iba k identifikácií konkrétnych užívateľov P2P, ale tiež umožňuje vytvorenie uceleného obrazu o chovaní P2P siete. Samotné preliezanie musí byť čo najrýchlejšie, aby bol vytvorený obraz o systéme validný, pri dlhom čase operácií by nemusel odpovedať skutočnosti (užívatelia sa pripájajú a odpájajú bez toho, aby boli zaznamenaný). Výhody a nevýhody prístupu:

- Výhodou je že, se systém poskytuje IP adresy užívateľov P2P s vysokou presnosťou, využíva *direct probing*. Ale z toho vyplývajú aj nevýhody prístupu.
- Sondovanie (*probing*) a preliezanie (*crawling*) musia byť efektívne, aby sa obmedzila náročnosť na zdroje (CPU, pamäť, zdroje siete) pri vysokom počte uzlov (čo je bežné) v sledovanej P2P sieti.
- Táto metóda tiež vyžaduje podrobnú znalosť príslušného P2P protokolu. Problém nastáva, pokiaľ sa nejedná o open-source protokol, alebo je použité šifrovanie.
- Detektor založený na tomto prístup musí držať krok s vývojom použitého P2P protokolu.
- Za nevýhodu tejto metódy sa dá považovať aj jej príliš „*aktívny*“ prístup k samotným procesom na sledovanej sieti. Pozorovateľ by nemal ovplyvňovať sledovaný systém, čo sa nedá v tomto prípade úplne eliminovať.

3.3 Analýza obsahu paketov

Metóda analýzy obsahu paketov, tiež známa ako *Deep Packet Inspection* (DPI), je založená na kontrole obsahu paketov v reálnom čase, kde sa pokúša o identifikáciu charakteristických vzorov P2P protokolov. Pre vyhľadávanie týchto vzorov (signatúr) je ich nutné najprv vytvoriť zo známej P2P prevádzky a následne vytvoriť zoznam signatúr pre P2P protokoly, ktoré chceme detekovať. Každý paket môže byť skontrolovaný a oklasifikovaný ako P2P prevádzka (pokiaľ je nájdená zhoda so signatúrou). Niekedy nie je záujmom detekcie iba identifikácia P2P užívateľov, ale celková prevádzka P2P a jej toky (zaujímavé údaje pre poskytovateľov internetu). Prevádzka a obojsmerné toky môže dedikovaný hardvér oklasifikovať ako P2P na základe identifikácie charakteristického paketu (odpovedá signatúre P2P protokolu)[18].

Táto metóda je vo všeobecnosti dosť rozšírená, ale trpí niekoľkými nedostatkami:

- Pri použití šifrovania spojení v rámci P2P prevádzky nastávajú problémy s kontrolou obsahu paketov, čo vedie k neúspešnej detekcií.
- P2P protokoly sa vyvíjajú, preto je nutné udržiavať signatúry aktuálne.
- Po aktualizácií zoznamu signatúr je potrebné aktualizovať aj všetky sieťové prvky využívajúce tieto vzory, čo je často netriviálne pre hardvérové implementácie (dedikovaný hardvér potrebný na vysokorýchlostných sieťach kvôli kontrole veľkého počtu paketov v reálnom čase).

- Náročnosť dedikovaného hardvéru na výpočetné zdroje, čo ovplyvňuje aj jeho cenu.

3.4 Analýza sieťových tokov

Podstatou detekcie analýzou sieťových tokov je použitie štatistik o tokoch (*flow statistics*), resp. pripojeniach, namiesto kontroly obsahu jednotlivých paketov. Cieľom je identifikácia podozrivých aktivít medzi užívateľmi v sledovanej sieti a odhalenie užívateľov generujúcich prevádzku odpovedajúcu popisu chovania sledovanej aplikácie. K tomu je vytvorený klasifikačný model a ďalšie heuristiky na základe vzťahov medzi rôznymi štatistickými vlastnosťami tokov a príslušnej aplikácie, ktorá ich generuje.

Výhody metódy:

- možnosť detekcie šifrovaných prenosov (oproti predchádzajúcim metódam),
- nemôže dôjsť ku konfliktu so zákonom (zachovanie listového tajomstva), keďže sa vôbec nepristupuje k dátovej časti obsahu paketu,
- nižšie technické požiadavky na hardware oproti metóde DPI.

Problémy nastávajú pri pokuse o zovšeobecnenie popisu chovania pre všetky P2P aplikácie. Každá sa správa inak, hoci existujú niektoré spoločné vlastnosti. Tento problém neostáva iba pri rozdieloch v implicitnom chovaní aplikácií, pretože ich chovanie je parametrizovateľné aj zo strany užívateľa prostredníctvom príslušného programu (klienta), kde už prakticky každý takýto klient umožňuje nastavenie používaných portov, obmedzenie počtu aktívnych spojení, stupeň anonymity, šifrovanie, atď.

Nutno poznamenať, že aj po vytvorení popisu chovania vybraného P2P systému, ktorý chceme detekovať (v našom prípade to bude BitTorrent), sa jeho chovanie môže meniť s každou novou verziou protokolu. Chovanie môže byť v prípade BitTorrentu ovplyvnené aj zahltením siete a (ne)dostupnosťou požadovaného súboru.

Situáciu komplikuje aj prítomnosť aplikácií (streaming, hry, P2P TV) založených na P2P architektúre, ktoré sa chovajú podobne ako P2P aplikácie pre zdieľanie súborov. Objavuje sa potreba riešenia problému s *false positives* (programov, ktoré sú chybne detekované aj keď slúžia k iným účelom ako je zdieľanie súborov).

Spoločné vlastnosti P2P protokolov (vychádzajú z [13]):

- **Celkový počet spojení**
  Užívateľ P2P siete sa môže väčšinou prezradiť veľkým počtom TCP spojení (resp. UDP tokov) z klientského počítača. Spravidla sa jedná o 1 až N spojení na jeden sťahovaný súbor, kde N odpovedá počtu segmentov, do ktorých je súbor rozdelený. Počet segmentov na jeden súbor je daný podielom celkovej veľkosti súboru a veľkosti segmentu. Vezmime si za príklad súbor o veľkosti 1400 MB a veľkosť jedného segmentu 2 MB. Dostávame počet segmentov 700, čo je maximálny počet aktívnych sťahovaní na súbor. Z princípu P2P systémov (výmeny dát) vyplýva, že užívateľ bude pravdepodobne sťahovať, resp. odosielať, niekoľko súborov, čím sa dostávame na niekoľko tisíc spojení z jedného klientského počítača.
- **Obojsmerné spojenia**
  Je to vlastnosť vychádzajúca už zo spomínaného princípu výmeny dát v P2P systémoch. Uzol, ktorý sťahuje súbor zároveň niečo iné poskytuje ostatným uzlom P2P

systému. Dá sa predpokladať podobnosť v množstve odoslaných a prijatých dát. To znamená, že dáta sa prenášajú v spojení oboma smermi, alebo existuje ďalšie spojenie pre prenos dat smerom von.

- **Súčasné využitie protokolov TCP a UDP**
  Väčšina P2P aplikácií využíva tieto protokoly súčasne. Príkladom je protokol BitTorrent, viď sekciu 2.3.1.

- **Veľký počet neúspešných spojení**
  Narozdiel od serverov v architektúre klient-server, užívatelia P2P systému svoje počítače zapínajú a vypínajú oveľa častejšie rovnako ako aj samotné aplikácie pre zdieľanie súborov. To môže viesť k pokusom o kontaktovanie nedostupného uzlu. Môžme predpokladať, že nejaký uzol sa o to budú pokúšať opakovane po nejakú dobu [8].

- **Maximálne veľkosti paketov**
  P2P aplikácie pre zdieľanie súborov sa snažia posielať čo najväčšie pakety (typicky pomocou TCP) pri samotnom prenose súborov. Zaujímavý je fakt, že tato snaha sa neprejavuje u aplikácií pre VoIP alebo hier kvôli zabezpečeniu dostatočnej odozvy.

- **Čísla portov nad 1024**
  Dnešné P2P aplikácie používajú pre svoje potreby náhodne čísla portov. Ich spoločnou vlastnosťou je to, že sa vždy jedná o hodnoty nad 1024 (pre zdrojový aj cieľový port). Toto tvrdenie platí pre tú časť P2P komunikácie, ktorá sa nemaskuje za inú známu službu (HTTP, HTTPS), pokiaľ P2P aplikácia využíva takúto metódu k predchádzaniu detekcie.

- **Konektivita užívateľov P2P aplikácií**
  Konektivitou sa v kontexte tejto práce myslí počet spojení (odchádzajúcich alebo prichádzajúcich), ktoré užívateľ P2P aplikácie udržiava s inými uzlami P2P systému. Typicky počet unikátnych vzdialených uzlov je odlišný pre každý P2P systém (závisí od použitej architektúry P2P siete). Počet týchto konkuretných spojení so vzdialenými uzlami sa môže líšiť aj v rámci rovnakého P2P systému, kde je táto nekonzistencia spôsobená použitím rozdielnych klientov. Každý klient môže implementovať iný maximálny počet súčasne udržiavaných spojení a táto hodnota môže byť zároveň parametrizovateľná užívateľom.

Samotné štatistiky o tokoch, nad ktorými je následne možno vykonávať analýzu prevádzky na sieti, sa vytvárajú s využitím technológií pre monitorovanie tokov. Najznámejšou technológiou (použitá pri tvorbe nášho detektoru) je technológia IP tokov NetFlow popísaná v nasledujúcom texte.

3.5 NetFlow

NetFlow je otvorený protokol vyvinutý firmou Cisco (popísaný v [6]) pre monitorovanie sieťových tokov, čím umožňuje správcom siete a manažérom získavať prehľad o prevádzke na sieti v reálnom čase. Termínom NetFlow sa označuje celá oblasť činnosti technológie, ktorá je zložená z troch častí, a to exportu záznamov o tokoch, zberu dát (záznamov) a ich analýzy [9].

Podstatou celej technológie je tok (*flow*), ktorý je v NetFlow definovaný[2] ako jednosmerná postupnosť paketov so zhodnou cieľovou a zdrojov IP adresou, zdrojovým a cieľovým portom, protokolom (TCP, UDP, ICMP, IGMP), IP typom služby (ToS[3]) a identifikátorom rozhrania. Naviac sa zaznamenáva doba vzniku, doba trvania (časové značky začiatku a konca IP toku), počet prenesených paketov a bytov, atď. (neukladá sa vlastný obsah paketov).

Tok vzniká v prípade, že je zachytený paket, ktorý nepatrí do žiadneho z už existujúcich aktívnych tokov. Ďalšie pakety tohto toku sú automaticky rozpoznané a spolu s úvodným paketom tvoria zdroj NetFlow dát. Je potrebné si uvedomiť, že obojsmerná komunikácia medzi dvoma užívateľmi na sieti je vždy popísaná najmenej dvoma tokmi.

Zásadnou výhodou technológie NetFlow je jej aplikovateľnosť aj na siete o rýchlosti 10Gb/s a siete s veľkou prevádzkou, na ktorých by bolo nasadenie inej technológie komplikované, ba až nemožné.

3.5.1 Architektúra NetFlow

Základné prvky systému:

- **Exportér** – zariadenie pripojené k monitorovanej linke analyzujúce prechádzajúce pakety. Zo získaných informácií sa vytvárajú záznamy o tokoch (tvorba štatistik), resp. sa aktualizujú už existujúce v pamäti NetFlow cache. Exportérom je typicky sieťové zariadenie typu router alebo L3 switch, tiež sa môže jednať o dedikovaný hardware (pasívnu sondu), alebo softvérovú implementáciu. K samotnému exportu tokov (tok je považovaný za ukončený a následne je vyradený z cache pamäte) dochádza pri splnení jednej z následujúcich podmienok:
  - Detekcia konca toku (obdržanie príznakov TCP FIN alebo RST).
  - Zaplnenie pamäte NetFlow cache.
  - Vypršanie neaktívneho časovača po nejakej dobe neaktivity prevádzky.
  - Vypršanie aktívneho časovača. V tomto prípade sa tok uzatvára po dobe definovanej aktívnym časovačom od výskytu prvého paketu toku. Ďalší paket, ktorý by bol inak súčasťou uzatvorené toku, sa stáva prvým paketom nového toku. Takto sa môžu dlhé toky rozpadať na niekoľko časovo kratších tokov.

  Expirované toky sú následne spojené do UDP datagramov (prístup je jednoduchší a rýchlejší ako u TCP, kde je každý paket potvrdzovaný) a vyexportované na zariadenie slúžiace ako kolektor.
- **Kolektor** – zariadenie zaznamenavajúce NetFlow záznamy odoslané exportérom. V praxi zbiera kolektor NetFlow záznamy z viac ako jedného exportéra. Takto získane informácie o tokoch sú uložené v databáze. Nad nimi beží aplikácia, ktorá umožňuje ich filtrovanie, agregáciu a vizualizáciu (tabuľky, grafy).

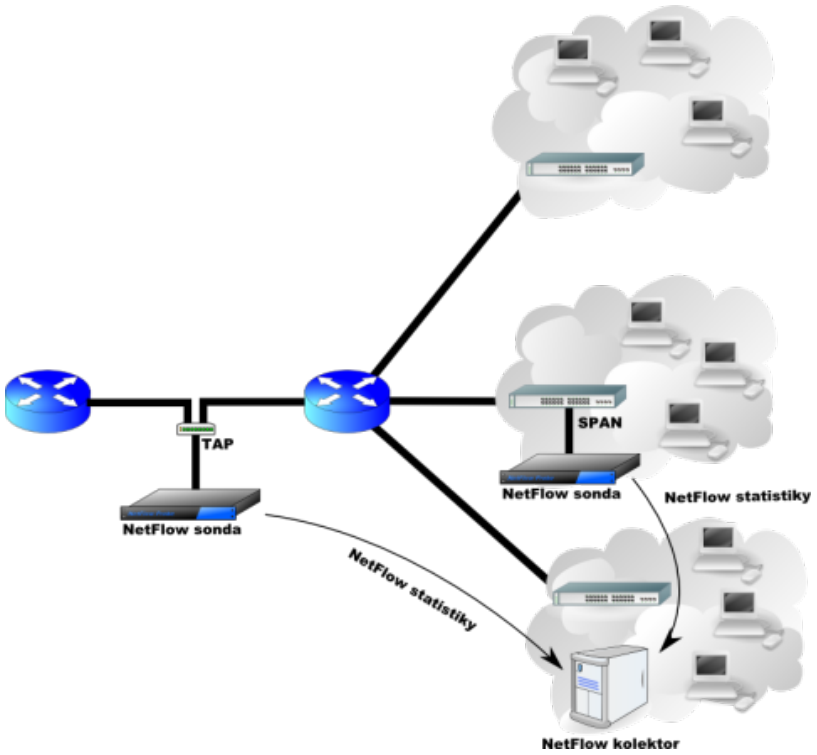
V **tradičnej architektúre** [12] sa predpokladajú na pozícií exportérov smerovače (routery), na ktorých ako doplnková služba prebieha výpočet NetFlow štatistik. Nevýhodou tohto usporiadania je vysoká cena podobných zariadení a dopad výpočtu NetFlow štatistik

[2] Obecná definícia: Postupnosť paketov so spoločnou vlastnosťou prechádzajúca bodom pozorovania za určitý časový interval [6].

[3] Type of Service

na celkový výkon zariadenia. V lacnejších zariadeniach s podporou NetFlow sa využíva vzorkovanie vstupu (deterministické alebo náhodné [6]), tzn. že pre výpočet štatistik je použitý iba každý n-tý paket. Týmto sa však znižuje presnosť merania a narastá pravdepodobnosť neúspechu pri odhaľovaní bezpečnostných incidentov.

Riešením je využitie **modernej architektúry**, ktorá využíva pasívne sondy[4] – špecializované zariadenia pre monitorovanie a export NetFlow štatistik. Sondy je možné zapojiť, narozdiel od smerovačov v tradičnej architektúre transparentne do ľubovoľného bodu v sieti. Monitorované dáta nie sú nijak ovplyvnené sondou, preto sú pasívne. Ďalšou výhodou moderného usporiadania je spôsob exportu štatistik na kolektor, obvykle dedikovanou linkou, čím sa neovplyvní monitorovaná linka [21]. Schéma takejto architektúry je zobrazená na obrázku 3.1.



Obrázok 3.1: Schéma modernej architektúry (prevzaté z [21]).

3.5.2 Verzie NetFlow

Prvou rozšírenou verziou protokolu NetFlow bola verzia 5 [2]. Dnes sa už používa aj verzia 9, ktorá má na rozdiel od predchádzajúcej verzie danú svoju štruktúru šablónou, čím sa stáva flexibilnejšou (možnosť pridávať položky bez zmeny exportného formátu). NetFlow v9 tiež poskytuje úplnú podporu protokolu IPv6. Na základe tejto verzie vznikol nový protokol Internet Protocol Flow Information Export (IPFIX) [6] ako snaha o zjednotenie metód pre prácu s IP tokmi.

[4]Príkladom takejto výkonnej sondy v ČR je FlowMon od firmy INVEA-TECH.

3.5.3 NFDUMP a NfSen

V tejto kapitole sa pozrieme na systém NfSen ako jedno z kvalitných open-source riešení softvérových NetFlow nástrojov[5]. Na tomto systéme je založená implementácia nášho detektoru popísaná ďalej v texte.

NFDUMP je balíček nástrojov určený pre zber a spracovanie NetFlow dát. Balíček je distribuovaný pod BSD licenciou a určený pre Unixové systémy. Všetky nástroje balíku NFDUMP podporujú NetFlow verzie 5, 7 a 9, konkrétne sa jedná o:

- **nfcapd (netflow capture daemon)** – démon, ktorý sa chová ako kolektor a zaznamenáva NetFlow dáta zo siete. Následne ich ukladá do súborov po n-minútových úsekoch (typicky 5 minút), potom vytvára nový súbor. Súbory sú pomenované podľa názvu démona a časovej značky v tvare *nfcapd.RRRRMMddhhmm* a sú uložené v binárnom tvare (napríklad *nfcapd.201301081120* značí interval 11:20 - 11:25 dňa 8.1.2013). Program transparentne zachytáva dáta všetkých podporovaných verzií NetFlow. Pre každý prúd NetFlow dát (pre každý exportér) je potrebná samostatná inštancia procesu.
- **nfdump** – je nástroj pre spracovanie dát uložených v súboroch programom `nfcapd`. Jedná sa o utilitu s rozhraním príkazového riadku. Dokáže generovať top N štatistiky a filtrovať zobrazované dáta. Využíva podobnú syntax ako nástroj `tcpdump`. Nástroj umožňuje výber úrovne detailu pri výpise a nastavenie vlastného formátu zobrazovaných dát, čo je užitočné pri použití v skriptoch.
- **nfprofile** – nástroj, ktorý umožňuje spracovávať a filtrovať súbory vytvorené a uložené nástrojom `nfcapd`. Filtrovanie prebieha na základe špecifikovanej sady filtrov (profilu) a výstup sa ukladá do súborov pre ďalšie spracovanie.
- **nfreplay** – umožňuje načítať NetFlow dáta zo súborov vytvorených nástrojom `nfcapd` a posielať ich po sieti iným užívateľom.

NfSen (NetFlow Sensor) je ďalší nástroj pre prácu s NetFlow dátami. Jedná sa o grafickú webovú nadstavbu (*frontend*) nad balíkom nástrojov NFDUMP. Najdôležitejšie funkcie NfSenu [10]:

- Vizualizácia NetFlow dát (tokov, paketov a bytov) s využitím RRD (Round Robin Database)[6].
- Ľahká navigácia medzi NetFlow dátami.
- Spracovanie NetFlow dát vo vybranom časovom úseku.
- Vytváranie reálnych a historických profilov pre sledovanie dátových tokov.
- Vyvolanie poplachu (*alert*) na základe definovaných podmienok.
- Možnosť tvorby vlastných pluginov pre spracovanie NetFlow dát v pravidelných intervaloch.

[5] Príkladom ďalších alternatív je nástroj nTop postavený nad knižnicou libpcap. Nástroje predstavujúce softvérové riešenia NetFlow sondy sú napríklad nProbe, fProbe, softflowd.

[6] Open-source priemyselný štandard s vysokým výkonom logovania dát a vykresľovania časových radov. Jednoducho integrovateľný do skriptov shellu, perlu, pythonu a ďalších.

Jadro systému je napísané v programovacom jazyku Perl a riadi backendové funkcie, ktoré nie sú vizualizované (štartovanie démona nfcapd apod.). Užívateľská správa systému (frontend systému) je vytvorená v programovacom jazyku PHP.

Systém pluginov

NfSen umožňuje rozšírenie funkcionality pomocou pluginov (zásuvných modulov). Každý plugin sa delí na dve časti:

- **backendové pluginy** – umožňujú rozšírenie funkcionality NfSenu o definovateľné poplachy (alerty) a o funkcie umožňujúce vykonávanie užívateľom definovaných operácií nad NetFlow dátami a to priamo na vyžiadanie alebo periodicky. Backendové pluginy sú rovnako ako väčšina NfSenu napísané v jazyku Perl a majú pevne definovanú štruktúru a rozhranie (povinné funkcie a ich návratové hodnoty).
- **frontendové pluginy** – poskytujú rozhranie pre komunikáciu medzi užívateľom a backendovým pluginom. Teda sú úplne závislé na príslušnom backendovom plugine, bez ktorého nemôžu existovať (prítomnosť frontendovej časti nie je podmienkou, plugin môže fungovať aj ako čisto backendový). Jedná sa o dynamické webové stránky napísane v jazyku PHP.

Komunikácia medzi frontendovým a backendovým pluginom NfSen kolektoru prebieha prostredníctvom komunikáčného kanálu `nfsend.comm` [1]. Funkcionalitu rozhrania zaisťujú moduly `Nfcomm.pm` pre backendovú časť a `nfseutil.php` pre frontendovú časť.

3.6 Zhrnutie

V prvej časti kapitoly sme sa oboznámili s typickými metódami detekcie P2P prevádzky. Pre ďalšiu časť práce je podstatná metóda detekcie na základe analýzy sieťových tokov, ktorá sa v porovnaní s inými metódami a ich nedostatkami (nespoľahlivosť metódy DPI pri šifrovanej prevádzke a primitívnosť detekcie na základe čísiel portov) javí ako najprijateľnejšia. Na princípe tejto metódy je založený návrh detekcie protokolu BitTorrent v sieťovej prevádzke.

Na základe vlastností bittorrentovej prevádzky popísaných v predchádzajúcich prácach a konfrontovaných s vlastnými kolekciami záznamov o sieťových tokoch (obsahujúcich bittorrentovú prevádzku) je navrhnutý a demonštrovaný nástroj pre detekciu tohto P2P systému.

V ďalšej časti sme si predstavili technológiu NetFlow a jej možnosti monitorovania sieťovej prevádzky a zberu dát (záznamov o tokoch). Dôvodom je stanovená podmienka práce, že všetky dáta potrebné k identifikácií P2P prevádzky musia byť dostupné v NetFlow záznamoch, čo umožní aplikovať proces detekcie P2P aktivity aj bez znalosti obsahu paketov alebo v prípade, že je tento obsah šifrovaný.

Kapitola 4

Detekcia protokolu BitTorrent

V tejto kapitole navrhneme systém pre detekciu protokolu BitTorrent, resp. identifikáciu užívateľov využívajúcich služby BitTorrentu, vo vybranom segmente siete (segment je daný maskou podsiete). Návrh je založený na analýze sieťových tokov a popise chovania sledovaného P2P systému. Všetky sledované parametre použité pri detekcií sú súčasťou štatistik o tokoch získaných technológiou NetFlow (popísanej v predchádzajúcej kapitole), alebo sú spočítateľné zo záznamov o jednotlivých tokoch.

Samotný výber parametrov, na základe ktorých sa pokúsime odhaliť užívateľov protokolu BitTorrent v sieti, vychádza jednak z obecných vlastností P2P sietí popísaných v sekcií 3.4 a ďalších vlastností špecifických pre chovanie protokolu BitTorrent, vychádzajúcich z [5, 8, 14]. Na základe zistených hodnôt sa určí skóre pre každú kandidátnu IPv4 adresu zo sledovaného segmentu siete (podsiete). Skóre odpovedá pravdepodobnosti, že sa jedná o IP adresu užívateľa protokolu BitTorrentu. Metodika výberu konkrétnych kandidátnych adries je popísaná ďalej v texte. Výpočet skóre je založený na štatistikách o tokoch pre príslušnú adresu.

4.1 Popis vytvorených kolekcií NetFlow dát

V rámci získavanie štatistik o tokoch v bittorrentovej komunikácií bolo vytvorených niekoľko kolekcií záznamov o tokoch. Záznamy boli vytvárané počas sťahovania rôznych testovacích súborov (linuxových distribúcií) protokolom BitTorrent. Pri sťahovaní boli použitý bittorrentový klienti Vuze[1] a Transmission[2].

Kolekcie boli vytvorené na vlastnom notebooku a pre vytvorenie NetFlow záznamov bola použitá softvérová implementácia NetFlow sondy *softflowd*[3]. Vytvorené kolekcie dát obsahujú okrem tokov bittorrentovej komunikácie aj toky základných služieb a protokolov (HTTP, DNS, ICMP,...), ktorých počet som sa v čase vytvárania záznamov pokúsil minimalizovať (kolekcie NetFlow záznamov a príslušné `.torrent` súbory sú priložené na CD v prílohe).

Základný popis kolekcií je v tabuľke 4.1, kde hodnota *L port (Listening port)* predstavuje nastaviteľné číslo portu v použitých klientoch. Na tomto porte bittorrentový klient naslúcha prichádzajúcim spojeniam a zároveň ho využíva ku kontaktovaniu uzlov BitTorrentu v roji (*swarme*). Klasifikácia bittorrentovej prevádzky vo vytvorených kolekciách spočívala vo

[1] `http://www.vuze.com/` verzia 4.7.0.2
[2] `http://www.transmissionbt.com/` verzia 2.77
[3] `http://www.mindrot.org/projects/softflowd/` verzia 0.9.9

č.	Sťahovaný súbor	L port	Toky		BitTorrentové toky	
			celkom	porty >1024 (%)*	celkom (%)**	port <1024
1.	[T] Debian	51413	5455	3284 (60.20%)	3274 (99.70%)	11
2.	[T] Fedora	51413	5487	4859 (88.55%)	4715 (97.04%)	18
3.	[T] Ubuntu	51413	4744	4000 (84.32%)	3212 (80.30%)	11
4.	[T] Mint	51413	850	746 (87.76%)	680 (91.15%)	1
5.	[V] Debian	51000	6800	5746 (84.50%)	5682 (98.89%)	6
6.	[V] Fedora	51001	6700	5649 (84.31%)	5534 (97.96%)	3
7.	[V] Ubuntu	51002	13591	10697 (78.71%)	10510 (98.25%)	17
8.	[V] Mint	51003	14078	9356 (66.46%)	9229 (98.64%)	15
9.	[T] hot-file A	51413	13770	10946 (79.49%)	10901 (99.59%)	37
10.	[T] hot-file B	51413	24118	23122 (95.87%)	22353 (96.67%)	33

Klient použitý pri sťahovaní testovacieho súboru:
[V] – Vuze [T] – Transmission

* Pomer tokov s portami>1024 k celkovému počtu
** Pomer tokov k počtu tokov s portami>1024

Tabuľka 4.1: Popis kolekcií NetFlow dát obsahujúcich komunikáciu protokolu BitTorrent.

vytvorení zoznamu adries uzlov BitTorrentu, ktorí boli kontaktovaný cez tento port. Ďalej bol každý tok medzi IPv4 adresou klienta (lokálny uzol BitTorrentu) a adresou zo zoznamu (vzdialený uzol) klasifikovaný ako bittorrentová prevádzka.

V rámcii kolekcií je okrem sťahovania linuxových distribúcií zachytené aj sťahovanie tzv. *hot-files* – horúcich súborov (viď sekciu 2.3.3), ktoré sú charakteristické zvýšeným počtom užívateľov podieľajúcich sa na ich distribúcií. Tieto súbory boli sťahované prakticky hneď po zverejnení ich metadátových súborov (`.torrent`).

V následujúcom texte budú rôzne tvrdenia o špecifických vlastnostiach chovania BitTorrentu v sieťovej prevádzke konfrontované s popísanými kolekciami dát. Pri tejto konfrontácií budú podrobnejšie zobrazené vlastnosti tokov v kolekciách.

4.2 Sledované parametre tokov

V tejto sekcií sa zameriame na parametre tokov špecifické pre protokol BitTorrent.

Na základe spoločnej vlastnosti P2P sietí o použití čísiel portov nad 1024 (sekcia 3.4) budem ďalej v texte uvažovať iba toky, kde zdrojový aj cieľový port majú hodnotu nad 1024 (pokiaľ to nebude explicitne uvedené inak). Týmto sa obmedzujeme iba na sledovanie *dátovej fázy* P2P komunikácie, viď sekciu 2.

Platnosť tejto vlastnosti pre protokol BitTorrent je znázornená v tabuľke 4.1. Z nameraných hodnôt tiež vyplýva, že v priemere až 96% tokov s portami nad 1024 tvoria toky bittorrentovej komunikácie a iba zanedbateľné množstvo tokov tejto komunikácie má aspoň jeden port menší ako 1024 (menej ako jedno percento). Preto budeme v následujúcom texte a pri konfrontácií našich kolekcií s očakávanými hodnotami považovať všetky toky s portami nad 1024 zároveň za bittorrentovú komunikáciu.

Pomer tokov s portami nad 1024 k celkovému počtu tokov

P2P aplikácie je možné v sieťovej prevádzke charakterizovať vysokým počtom tokov s číslami portov nad 1024, narozdiel od klasických aplikácií (HTTP, FTP), ktoré tieto porty pou-

žívajú zriedkavo. Pre protokol BitTorrent je toto tvrdenie znázornené opäť v tabuľke 4.1, z ktorej vyplýva, že počet tokov s portami nad 1024 (v podstate komunikácia BitTorrentu) predstavuje v priemere 81% celkového počtu tokov.

Čísla portov

Problémy so sledovaním špecifických portov P2P protokolu za účelom jeho detekcie sú zhrnuté v sekcií 3.1. Avšak aj pri použití dynamických portov môže bittorrentový klient naviazať spojenie so vzdialeným užívateľom, v ktorom bude cieľový port spadať do rozsahu portov špecifických pre sledovaný protokol (čím je klient aktívnejší, tým narastá počet spojení a zároveň šanca, že dôjde k takémuto spojeniu). Rozsah špecifických (odporúčaných) čísiel portov pre výmenu dát medzi uzlami BitTorrentu je *6881-6889* [7].

Ďalším zaujímavým portom je číslo *6969*. Opäť sa jedná o odporúčané číslo portu, cez ktoré by mal *tracker* komunikovať s uzlami BitTorrentu. Väčšina trackerov používa iné čísla portov (často sa zámerne používa port 80 kvôli maskovaniu prevádzky), ale aj tak môže byť výskyt tohto portu ďalším vodítkom k detekcii užívateľov BitTorrentu.

Špecifické čísla portov sa nepodieľajú na výpočte skóre kandidátnej adresy. Napriek tomu sa v rámci štatistík zaznamenávajú a ich použitie (v špeciálnych prípadoch) je popísane ďalej.

Pre BitTorrent platí obecná vlastnosť využívania portov s hodnotou nad 1024, ale väčšina týchto portov je mimo rozsah 6881-6889. V práci J.Hadviga [11] sa uvádza ako jeden z rysov komunikácie BitTorrentu využitie čísiel portov až nad *10 240* na obidvoch stranách spojenia v približne 88% tokov. Tomuto tvrdeniu odpovedá aj meranie nad našimi kolekciami dát, ktorého výsledky sú v tabuľke 4.2.

	Toky			**Prenesené dáta**	
č.	**celkom**	**porty >10240 (%)**	**Slonie (%)**	**celkom**	**Slonie (%)**
1.	3284	2937 (89.43%)	30 (0.91%)	716.39 MB	714.52 MB (99.74%)
2.	4859	4263 (87.73%)	192 (3.95%)	972.98 MB	969.33 MB (99.62%)
3.	4000	3568 (89.20%)	110 (2.75%)	718.52 MB	716.72 MB (99.75%)
4.	746	671 (89.95%)	88 (11.80%)	841.29 MB	840.78 MB (99.94%)
5.	5746	5133 (89.33%)	14 (0.24%)	694.08 MB	691.97 MB (99.70%)
6.	5649	5000 (88.51%)	228 (4.04%)	968.64 MB	966.38 MB (99.77%)
7.	10697	9731 (90.97%)	173 (1.62%)	846.37 MB	842.7 MB (99.57%)
8.	9356	8300 (88.71%)	131 (1.40%)	950.34 MB	947.49 MB (99.70%)
9.	10946	10148 (92.71%)	154 (1.41%)	1.46 GB	1.45 GB (99.34%)
10.	23122	21903 (94.73%)	245 (1.06%)	2.20 GB	2.18 GB (99.16%)
Priemer:		**90.13%**	**2.92%**		**99.63%**

Tabuľka 4.2: Výsledky merania veľkosti čísiel portov v tokoch a počtu sloních tokov vrátané preneseného objemu dát.

V tabuľke sú pre každú kolekciu dát zaznamenané počty tokov, v ktorých zdrojový aj cieľový portov majú hodnotu nad 10240 (a pomer k celkovému počtu tokov). Tieto toky predstavujú v priemere 90% celkovej BitTorrentovej komunikácie, čo potvrdzuje predchádzajúce tvrdenie.

Slonie toky (počet paketov v toku)

Prítomnosť *sloních tokov* (*elephant flows*) je charakteristickou vlastnosťou protokolu BitTorrent podľa [14]. Sloní tok je tok, ktorý je tvorený viac ako 100 paketmi[4].

V tejto práci sme nerozlišovali medzi TCP a UDP sloními tokmi, keďže pre prenos dát sa používajú oba protokoly a charakteristický počet paketov sa na nich vzťahuje rovnako (problém môže nastať pri nastavení aktívneho časovača pre UDP toky na príliš krátku dobu).

Počet sloních tokov tvorí menej ako 10% celkového počtu tokov komunikácie BitTorrentu a zároveň predstavuje viac ako 98% celkovej prevádzky BitTorrentu. Tomuto tvrdeniu odpovedajú nami namerané hodnoty v tabuľke 4.2.

Neúspešné spojenia

Tok je považovaný za neúspešné spojenie, pokiaľ obsahuje iba TCP príznak SYN a žiadny ACK, a zároveň je tvorený 1-3 paketmi. Jedná sa o tzv. *krátky tok*, ktorý indikuje nejakú chybu v komunikácií alebo anomáliu v kolekcií tokov. V komunikácií BitTorrentu sa môže jednať až o 22% celkového počtu tokov [8].

Pomocou tohto parameteru je prevádzka BitTorrentu dobre rozlíšiteľná od HTTP alebo FTP komunikácie. Je to dané relatívne vysokým počtom neúspešných spojení na strane BitTorrentu oproti zvyšným dvom službám. Tento nepomer sa dá zdôvodniť otváraním spojení na žiadosť aplikácie (BitTorrent) naproti otváraniu spojení na žiadosť užívateľa, ktorý to po niekoľkých pokusoch pravdepodobne vzdá.

V rámci príslušných štatistík sa okrem samotného počtu neúspešných spojení sleduje aj počet rôznych uzlov BitTorrentu, ktorý sú sledovanou adresou neúspešne kontaktovaný opakovane. Ďalej sa sledujú počty spojení v týchto opakovaných pokusoch a ich pomer k celkovému počtu neúspešných spojení. Dôvodom je predpoklad, že sa bittorrentová aplikácia nevzdáva pokusov o kontaktovanie nedostupného uzlu iba po jednom pokuse. Výsledky merania, vyplývajúceho z tohto predpokladu, sú v tabuľke 4.3.

	Neúspešne kontaktované uzly		**Neúspešné spojenia**	
č.	**celkom**	**opakovane (%)**	**celkom**	**opakovane (%)**
1.	6	6 (100.00%)	28	28 (100.00%)
2.	22	19 (86.36%)	93	90 (96.77%)
3.	72	59 (81.94%)	232	219 (94.40%)
4.	11	11 (100.00%)	35	35 (100.00%)
5.	1	1 (100.00%)	3	3 (100.00%)
6.	13	2 (15.38%)	15	4 (26.67%)
7.	30	12 (40.00%)	44	26 (59.09%)
8.	14	4 (28.57%)	18	8 (44.44%)
9.	54	32 (59.26%)	182	160 (87.91%)
10.	364	187 (51.37%)	1002	825 (82.34%)

Tabuľka 4.3: Výsledky merania výskytu neúspešných spojení a ich opakovaní.

[4]Ďalším charakteristickým tokom komunikácie BitTorrentu je tzv. *korytnačí tok* (*tortoise flow*), ktorý je definovaný dĺžkou trvania viac ako 100 sekúnd. V prípade TCP protokolu korešpondujú korytnačie toky so sloními tokmi.

Z výsledkov tohto merania je zjavná nekonzistencia medzi hodnotami získanými použitím rôznych bittorrentových klientov (viď tabuľku 4.1). Možným dôvodom je rozdielna implementácia a preferovanie odlišného transportného protokolu. Z hodnôt prezentovaných v tabuľke 4.4 je zrejme, že klient použitý pri ich vytváraní kolekcií 5.–8. uprednostňuje komunikáciu prostredníctvom UDP pred TCP (resp. konkurentnom využití UDP a TCP). To sa prejavuje menším počtom krátkych TCP tokov (tabuľka 4.3).

	Kontaktované uzly				
			Transportný protokol		
č.	**celkom**	**úspešne (%)**	**TCP∧UDP**	**TCP**	**UDP**
1.	943	839 (88.97%)	259	231	454
2.	1349	1038 (76.95%)	284	279	786
3.	964	883 (91.60%)	175	450	339
4.	241	163 (67.63%)	61	27	153
5.	2289	2040 (89.12%)	1	12	2276
6.	2227	1843 (82.76%)	6	160	2061
7.	3351	2761 (82.39%)	6	473	2872
8.	2487	2086 (83.88%)	4	269	2214
9.	1151	1045 (90.79%)	641	73	437
10.	2298	2127 (92.56%)	1092	105	1101

Tabuľka 4.4: Výsledky merania pomeru úspešne kontaktovaných uzlov a počtu uzlov kontaktovaných protokolom TCP, UDP alebo oboma zároveň.

V tabuľke 4.4 sú zároveň predstavené výsledky ďalšieho merania, v ktorom sme overovali pomer úspešne kontaktovaných uzlov a všetkých kontaktovaných uzlov. V tomto prípade sme nerozlišovali medzi použitím TCP alebo UDP protokolu. Z výsledkov vyplýva, že iba približne 85% kontaktovaných uzlov BitTorrentu reagovalo na pokus o vytvorenie spojenia.

Použité transportné protokoly

Behom dátovej fázy sa využíva pre prenos dát protokol TCP alebo UDP. Pri koexistencií TCP a UDP tokov sa objavujú aj prípady, kde klient komunikuje so vzdialeným užívateľom s využitím obidvoch protokolov. V takýchto prípadoch sú prítomne UDP toky výsledkom komunikácie mechanizmus DHT, viď sekciu 2.3.2. Počet takýchto uzlov, s ktorými je vytvorené TCP a UDP spojenie zároveň, je zobrazený v tabuľke 4.4. Opäť je zrejmá nekonzistencia v počte týchto uzlov pre jednotlivé kolekcie. Avšak už samotná existencia takýchto uzlov je ďalším vodítkom k identifikácií užívateľov BitTorrentu.

Sledovanie spojenia so známym trackerom

Myšlienkou je sledovanie spojení so známymi trackermi na predom vytvorenom zozname (zoznam najznámejších trackerov). Táto metóda bola zavrhnutá. Dôvodom je samotná identifikácia najznámejších trackerov. Zoznamy dostupné na Internete[5] často mylne označujú za trackerov webové servery, ktoré iba indexujú a poskytujú k stiahnutiu zozbierané `.torrent` súbory.

[5]napríklad http://lifehacker.com/5460636/five-best-public-bittorrent-trackers

Počet konkuretných spojení s rôznymi uzlami BitTorrentu

Počet rôznych uzlov, s ktorými klient komunikuje v rámci sledovaného intervalu, je v prípade BitTorrentu podstatne vyšší ako u iných aplikácií, resp. protokolov. Tento počet odpovedá veľkosti zoznamu uzlov v roji (*swarme*), ktorý je zaslaný klientovi trackerom (pokiaľ nie je nastavené obmedzenie pre maximálny počet). Môže sa jednať až o stovky rôznych uzlov BitTorrentu, viď tabuľku 4.4.

Počtom konkuretných spojení s rôznymi uzlami sa BitTorrentu približuje napríklad protokol Skype, kde je tato hodnota závislá na počte účastníkov v komunikácií. Podobné hodnoty ako BitTorrent by mohli nadobúdať aj užívatelia s intenzívnou webovou prevádzkou (množstvo pripojení na rozličné adresy cez HTTP, port 80). To je v našom prípade zanedbateľné, keďže uvažujeme iba komunikáciu prostredníctvom tokov s portami nad 1024 (čím sa eliminujú aj ďalšie protokoly s nižším počtom unikátnych uzlov – mail, FTP, DNS). V rámci týchto tokov by sa zvýšenou konektivitou mohli prejaviť aj aplikácie ako streaming, hry alebo iné P2P protokoly.

Táto metrika je použitá pri výbere kandidátnych adries v implementovanom systéme detekcie. Typicky sa vyberá iba niekoľko najkomunikatívnejších adries zo sledovanej podsiete. Počet týchto adries je konfigurovateľný a závisí na veľkosti sledovanej podsiete a prevádzke (množstvo a zložitosť NetFlow záznamov) v tejto sietí (je potrebné zaručiť spracovanie štatistik týchto adries v požadovanom čase).

Distribúcia vzdialených portov (remote ports distribution – RPD)

Toto kritérium sa podieľa na celkovej známke spracovávanej kandidátnej adresy a príslušné čiastkové skóre môže nadobúdať nízke hodnoty aj pri komunikácií s veľkým počtom unikátnych vzdialených adries. Samotná hodnota je závislá na počte rozdielnych vzdialených portov a počte konkurentných tokov na týchto portoch v sledovanom intervale. Počíta sa samostatne pre TCP a UDP toky. U P2P aplikácií ako BitTorrent je hodnota RPD pomerne vysoká kvôli použitiu dynamických portov pri komunikácií medzi uzlami. Zatiaľ čo u aplikácií, kde sa používa malý počet rozličných (špecifických) portov, je hodnota nízka aj pri komunikácií s väčším počtom vzdialených užívateľov (príkladom takýchto aplikácií je už spomínaný streaming alebo hry viacerých hráčov).

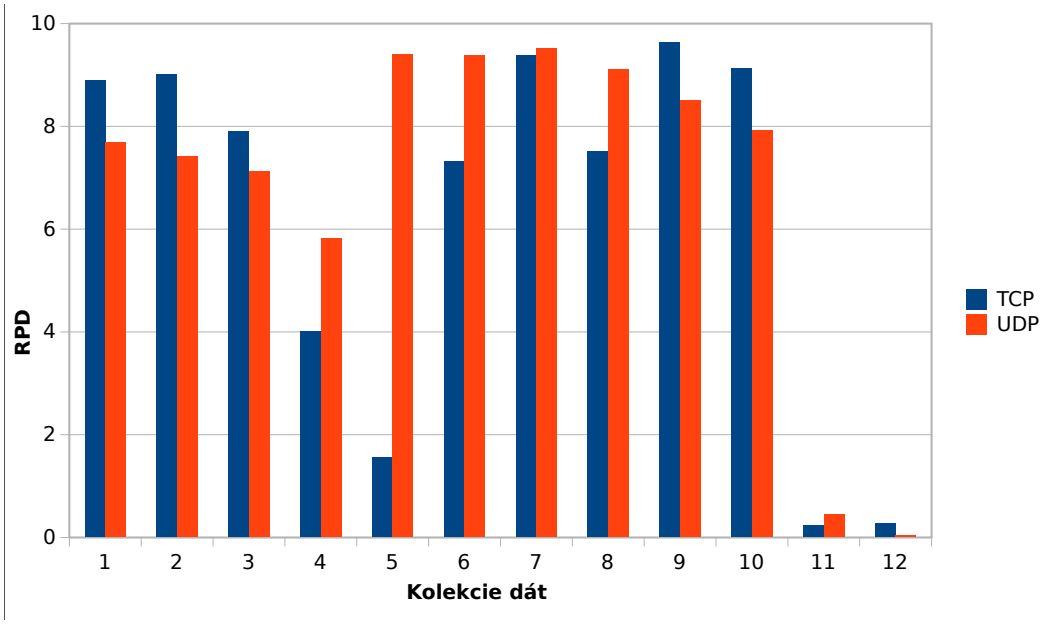
Táto metóda je popísaná v [5], kde je RPD pre konkurentné TCP, resp. UDP toky v intervale t definovaná ako:

$$D(t) = \frac{1}{n} \sum_{i=1}^{m} \log_2 \frac{n}{x_i} \tag{4.1}$$

kde n je počet konkuretných tokov v intervale t, m je počet rôznych vzdialených portov (je zrejmé, že $m \leq n$), a x_i značí počet tokov so vzdialeným číslom portu rovným $i$tému portu. Ukážka hodnôt RPD pre TCP a UDP toky je znázornená na obrázku 4.1.

Pre kontrast sú na obrázku zobrazené aj hodnoty RPD pre kolekcie dát obsahujúce streaming (kolekcia č.11) a komunikáciu multiplayer hry[6] (kolekcia č.12).

[6] League of Legends http://eune.leagueoflegends.com/



Obrázok 4.1: Hodnoty RPD pre toky v kolekciách dát.

4.3 Metódy detekcie BitTorrentu

V rámci vývoja nástroja pre detekciu BitTorrentu bolo použitých niekoľko rôznych prístupov k získavaniu a spracovávaniu štatistických údajov zo záznamov o sieťových tokoch. Samotný nástroj bol vyvíjaný ako skript v jazyku Perl a následne definovaním špecifických funkcií (povinných funkcií rozhrania) bol vytvorený plnohodnotný plugin kolektoru NfSen s frontendovým užívateľským rozhraním v jazyku PHP.

V tejto sekcii si jednotlivé prístupy stručne zhrnieme a predstavíme ich výhody a nevýhody. Posledná varianta je finálna a je základom implementovaného pluginu pre automatizovanú detekciu užívateľov BitTorrentu v sledovanej podsieti.

Všetky tri metódy pracujú s rovnakými štatistickými údajmi, ale zásadne sa líšia:

- metodikou výberu kandidátnych adries,
- využitím nástroja NFDUMP pre získanie potrebných hodnôt,
- pamäťovej a časovej náročnosti detekcie.

4.3.1 Metóda A

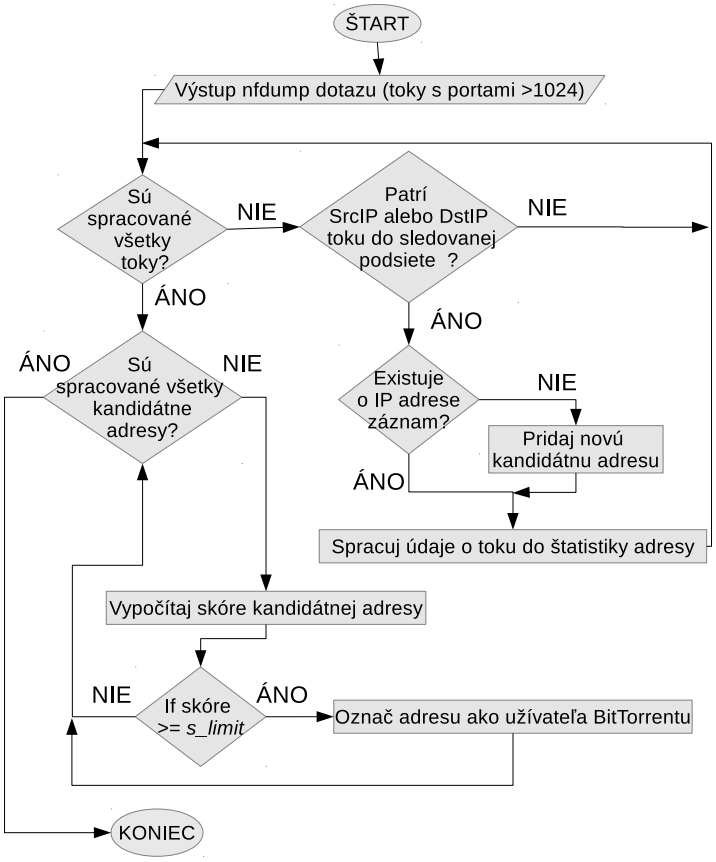
Tento prístup bol použitý v prototype nástroja a výber kandidátnych adries bol založený na sledovaní všetkých adries z podsiete. Samotný algoritmus je založený na jedinom nfdump dotaze v tvare:

```
nfdump -q -N -o "fmt:%pkt,%td,%pr,%sa,%sp,%da,%dp,%flg,%byt"
      'ipv4 and src port > 1024 and dst port > 1024'
```

Výstupom sú všetky toky s portami nad 1024. Tieto toky sa následne spracovávajú v jednom sekvenčnom prechode. Každá adresa (cieľová alebo zdrojová) v práve spracovávanom toku sa stáva kandidátnou, pokiaľ o nej ešte neexistuje záznam a patrí do sledovanej podsiete.

Pre všetky zaznamenané kandidátne adresy sa sumarizujú štatistické údaje (prenos dát v oboch smeroch, počet tokov, počet tokov s portami nad 10240, neúspešné spojenia, slonie toky,...) z tokov, v ktorých príslušná adresa figuruje.

V druhom sekvenčnom prechode nad zoznamom kandidátnych adries (kľúče asociatívneho poľa, kde hodnoty sú hodnoty spočítané v prvom prechode) sa pre aktuálnu adresu vypočíta skóre. Adresa je zaradená medzi detekovaných užívateľov protokolu BitTorrent, pokiaľ je hodnota skóre nad zadaným limitom (`s_limit`). Vývojový diagram popísaného algoritmu je na obrázku 4.2.



Obrázok 4.2: Vývojový diagram popisujúci algoritmus metódy A.

Hlavným nedostatkom je pamäťová náročnosť metódy, ktorá je zrejmá hlavne pri spracovaní dát s rádovo až miliónmi tokov. Istá výhoda tohto prístupu spočíva v jedinom nfdump dotaze a relatívne rýchlom a predvídateľnom čase spracovania. Časová zložitosť popísaného algoritmu spracovania výstupu nfdump dotazu je $O(n+k)$, kde n je počet tokov vo výstupe nfdump dotazu a k je počet kandidátnych adries.

Výkonnostné testy tejto metódy (príslušného skriptu) sú v tabuľke 4.5. Testovanie prebiehalo na vlastnom stroji[7]. Záznamy o veľkosti 556 KB odpovedajú kolekcií číslo 1. prezentovanej v predchádzajúcej sekcií (všetky kolekcie majú približne rovnakú veľkosť, preto bola vybraná iba jedna vzorka). Ďalšie záznamy sú rôzne derivácie neklasifikovanej kolekcie anonymizovaných NetFlow dát, ktoré boli použité výlučne k výkonnostným testom. Do záznamoch o veľkosti 14 MB, 37 MB a 327 MB sú zároveň nasadené záznamy o tokoch bittorrentovej komunikácie generovanej adresami 192.168.1.105 a 192.168.1.112. Pri ich testovaní sa sledoval príslušná podsieť (192.168/16), okrem záznam o veľkosti 37 MB. Tam nebola detekcia obmedzená, sledovali sa všetky adresy.

	Veľkosť dát			
	556 KB	**14 MB**	**37 MB**	**327 MB**
Čas	0.448 s	1.780 s	8.457 s	43.066 s
Pamäť	147.53 MB	180.81 MB	407.60 MB	1.63 GB

Tabuľka 4.5: Výkonnostné testy metódy A.

Na základe výsledkov výkonnostných testov môžme povedať, že táto metóda je schopná spracovávať NetFlow dáta v reálnom čase, ale na úkor zvýšenej pamäťovej náročnosti. Tá narastá úmerne počtu spracovávaných tokov (tj. závisí aj na veľkosti sledovanej podsiete). Pri testovaní boli zároveň úspešne detekované nasadené adresy generujúce bittorentovú prevádzku. Prí záznamoch o veľkosti 37 MB boli detekované aj ďalšie adresy, ktoré však pochádzajú z neklasifikovanej kolekcie dát.

4.3.2 Metóda B

V tejto metóde bol výber kandidátov založený na modifikovanom filtri nfdump dotazu navrhnutom Jakubom Čeganom v [23]. Výstupom pôvodného filtru boli toky, ktoré predstavovali inicializačnú komunikáciu medzi klientom a iným uzlom BitTorrentu pred samotným začatím sťahovania. Počet takýchto tokov udáva s akou pravdepodobnosťou je adresa podozrivá na bežiaceho BitTorrent klienta.

Vzhľadom na to, že sa vychádzalo zo špecifikácie protokolu a filter bol navrhnutý ešte v roku 2009, sa dalo predpokladať, že nebude úspešný. To sa potvrdilo nulovou úspešnosťou pri overovaní použiteľnosti tohto nfdump filtru nad našimi kolekciami dát.

Následne bol filter modifikovaný a rozšírený do tvaru:

```
'src net SUB.NET/maskbits and dst port > 1024 and src port > 1024
and ((((flags S and not flags ARPFU) or (flags A and not flags RPFUS))
                and (bpp>50 and bpp<61) and packets < 4)
               or ((src port > 6880 and src port < 6890)
              or (dst port > 6880 and dst port < 6890)))'
```

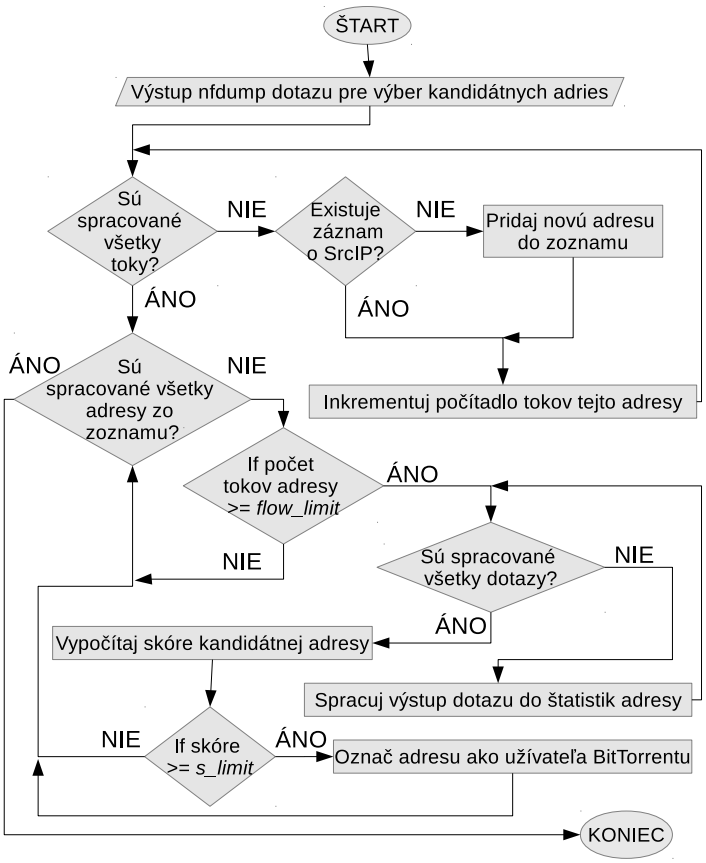
Opäť sa vyberajú toky nad 1024 zo sledovanej podsiete. Hľadajú sa charakteristické krátke TCP toky. Zároveň je filter rozšírený aj o výber tokov na špecifických portoch pre zvýšenie úspešnosti. Označme celkový počet tokov vo výstupe ako n.

[7] Vlastný stroj – Intel Core i3, 4 GB RAM.
Pri meraní pamäťových nárokov metódy bol použitý modul `Proc::ProcessTable`, http://search.cpan.org/~durist/Proc-ProcessTable-0.40/ProcessTable.pm

Na základe výstupu sa vytvorí zoznam dvojíc – adresa a počet k nej príslušných tokov. V rámci tohto zoznamu existujú kandidátne adresy (ich počet označme k) – splňujú podmienku, že počet ich tokov je viac ako `flow_limit`. V rámci výkonnostných testov bola ako `flow_limit` použitá hodnota 10.

V sekvenčnom prechode zoznamom dvojíc sa pri identifikácií kandidátnej adresy získajú štatistické údaje volaním niekoľkých nfdump dotazov (počet dotazov je d). Z výstupu dotazov a zostavených štatistík sa následne vypočítalo skóre kandidátnej adresy. Počet dotazov nad NetFlow dátami je priamo úmerný počtu kandidátnych adries (na jednu pripadá 10 dotazov). Časová zložitosť spracovania dát popísanou metódou je $O(n + k * d)$[8].

Vývojový diagram popísaného algoritmu metódy je na obrázku 4.3.



Obrázok 4.3: Vývojový diagram popisujúci algoritmus metódy B.

Veľkou výhodou oproti predchádzajúcej metóde je prakticky konštantná pamäťová náročnosť, keďže sa vždy spracováva iba jedna kandidátna adresa. Nevýhody metódy sú:

- Metodika výberu kandidátnych adries. Rozšírenie filtru pre výber týchto adries o detekciu tokov na špecifických portoch nie je ideálne (viď sekciu 3.1).
- Charakteristické krátke toky sa vyskytujú iba na začiatku komunikácie medzi uzlami BitTorrentu (pripojenie uzlu do *swarmu*). Pri dotaze nad dátami obsahujúcimi dlhodobejšiu komunikáciu (niekoľko päťminútových okien) je úspešnosť detekcie lepšia

[8]Napriek tomu, že d je multiplikatívna konštanta, som sa rozhodol ponechať ju v zápise kvôli zdôrazneniu podielu časovej zložitosti vlastných nfdump dotazov na celkovej zložitosti.

ako pri spracovaní jednotlivých okien samostatne. Môže dôjsť k situácií, že nedôjde k detekcií ani v jednom časovom okne . Takáto metóda sa stáva nepoužiteľnou pri periodickom spracovávaní tokov v kolektore.

- Časová náročnosť nie je závislá iba na počte spracovávaných tokov, ale aj na počte kandidátnych adries – akumuluje sa počet dotazov potrebných pre zisk štatistických údajov. Patová situácia nastáva v prípade veľkej kolekcie dát (dlhšia doba spracovania dotazu) s vyšším počtom kandidátov (veľký počet dotazov). V takomto prípade je nutné obmedzenie maximálneho počtu spracovávaných kandidátnych adries pre ukončenie spracovania v rozumnom čase (menej ako päť minút).

V tabuľke 4.6 sa nachádzajú výsledky výkonnostných testov metódy. Pri testovaní sú použité rovnaké záznamy ako pri metóde A.

	Veľkosť dát			
	556 KB	**14 MB**	**37 MB**	**327 MB**
Čas	0.147 s	0.521 s	8.017 s	5.123 s
Pamäť	139.0 MB	141.25 MB	141.46 MB	141.29 MB

Tabuľka 4.6: Výkonnostné testy metódy B.

Z výsledkov testovania je zrejmé, že metóda dokáže spracovávať záznamy o tokoch v reálnom čase a má nízku časovú a pamäťovú náročnosť. Dokonca bola úspešná aj pri detekcií užívateľov BitTorrentu, rovnako ako metóda A (vrátane detekcie užívateľov z neklasifikovaných záznamov). Avšak pri aplikovaní tejto metódy na kolekcie 5.–8. z predchádzajúcej sekcie (tabuľka 4.1) sa prejavili nedostatky metódy. Pri nastavení hodnoty `flow_limit` na 10 bola úspešnosť detekcie nad jednotlivými nfcapd súbormi (predstavujú päťminútové okno záznamov) veľmi nízka – adresa generujúca bittorrentovú prevádzku sa neobjavovala medzi kandidátnymi adresami. Výsledky tohto merania sú v tabuľke 4.7.

č.	**Počet intervalov**	**Počet detekcií**
5.	5	0
6.	4	1
7.	5	1
8.	5	0

Tabuľka 4.7: Výsledky metódy B pri spracovaní kolekcií 5.–8. po jednotlivých intervaloch.

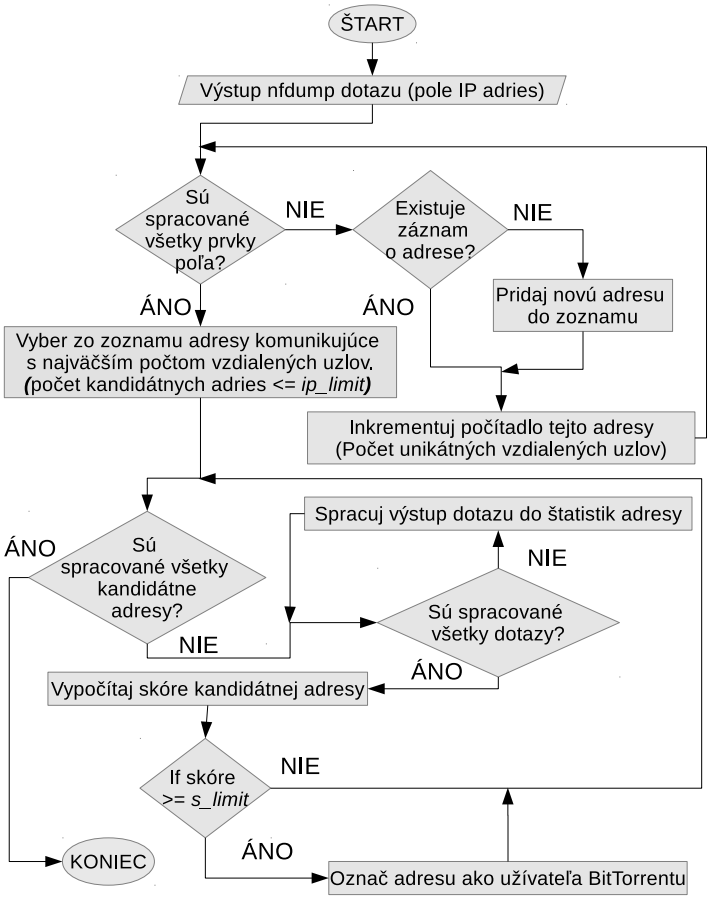
4.3.3 Metóda C

Finálna metóda, v ktorej je výber kandidátov založený na ich konektivite (počet unikátnych adries, s ktorými komunikujú súčasne). Nfdump dotaz pre výber týchto adries je:

```
nfdump -r nfcapd.timeslot -a -A srcip,dstip -o "fmt:%sa"
'src net SUB.NET/maskbits and src port > 1024 and dst port > 1024' -q -N
```

Výstupom dotazu je pole adries zo sledovanej podsiete. Počet výskytu rovnakej adresy v poli udáva počet vzdialených uzlov, s ktorými sa pokúšala nadviazať spojenie.

Následne sa spracováva iba niekoľko adries komunikujúcich s najväčším počtom vzdialených uzlov (pri periodickom spracovaní v kolektore je implicitná hodnota 10). Výber maximálneho počtu (`ip_limit`) je parametrizovateľný užívateľom (pri nastavení by sa mala zohľadniť očakávaná veľkosť vygenerovaných NetFlow dát, aby bolo zaručené dokončenie spracovania v reálnom čase). Schéma tejto metódy je zobrazená na vývojovom diagrame na obrázku 4.4.



Obrázok 4.4: Vývojový diagram popisujúci algoritmus metódy C.

Výsledky výkonnostných testov metódy sú zobrazené v tabuľke 4.8. Boli použité rovnaké záznamy ako pri testovaní výkonnosti predchádzajúcich metód.

	Veľkosť dát			
	556 KB	**14 MB**	**37 MB**	**327 MB**
Čas	1.694 s	3.806 s	8.101 s	46.541 s
Pamäť	139.24 MB	152.31 MB	162.99 MB	152.23 MB

Tabuľka 4.8: Výkonnostné testy metódy C.

Na základe výsledkov výkonnostných testov môžme konštatovať, že metóda je schopná spracovávať generované NetFlow dáta v reálnom čase. Časová a pamäťová náročnosť je porovnateľná s predchádzajúcimi metódami a je to kompromis medzi pamäťovou náročnosťou metódy B (sekvenčné spracovanie kandidátnych adries) a časovou náročnosťou metódy A

(sekvenčné spracovanie výstupov komplexnejších dotazov). Časová zložitosť metódy je teda $O(n+k*d)$, kde n je počet tokov vo výstupe prvého dotazu, k je počet kandidátnych adries ($k \leq$`ip_limit`) a d je počet dotazov.

Pri testovaní neboli vždy detekované nasadené adresy užívateľov BitTorrentu (narozdiel od predchádzajúcich metód), pretože sa nedostali medzi kandidátne adresy (limit bol nastavený na hodnotu 10). Všetky miesta boli obsadené adresami z neklasifikovanej kolekcie dát.

Popis nfdump dotazov

Potrebné hodnoty pre výpočet skóre na základe štatistik sieťovej prevádzky pre kandidátnu adresu (spracováva sa po jednej) sa získavajú konštantným počtom dotazov, ktoré sú definované v následujúcom asociatívnom poli. Každý dotaz je definovaný *filtrom* a prepínačmi nástroja `nfdump`.

```
# Filter pre vyber tokov nad 1024.
my $port_filter = "src port > 1024 and dst port > 1024";

# Definicia nfdump dotazov -- dvojice [ "'filter'", "prepinace" ]
my %filters = (
        # Statistiky tokov s portami nad 1024.
        # Vystup: pocet tokov, mnozstvo prenesenych dat (in/out) v bytoch.
        out_stats   => [ "'src ip IP and $port_filter'",
                         '-s srcip -q -N' ],
        in_stats    => [ "'dst ip IP and $port_filter'",
                         '-s dstip -q -N' ],

        # Pocet tokov s portom pod 1024 (pocet riadkov).
        lowp_flow   => [ "'ip IP and port < 1025'",
                         '-o "fmt:%sa"-q -N' ],

        # Slonie toky. Vystup: pocet tokov a velkost prenesenych dat.
        elephant    => [ "'ip IP and packets > 100 and $port_filter'",
                         '-o "fmt:%byt" -q -N' ],

        # Neuspesne spojenia.
        # Vystup: pocet spojeni a pocet neuspesne kontaktovany uzlov.
        fc_aggr     => [ "'src ip IP and (flags S and not flags ARPFU)
                            and packets < 4 and $port_filter'",
                         '-A dstip -o "fmt:%fl" -q' ],

        # Konkurentne TCP a UDP spojenia s rovnakym uzlom.
        ctu_aggr    => [ "'src ip IP and $port_filter'",
                         '-a -A proto,dstip -o "fmt:%da %pr %fl" -q -N' ],

        # Toky so specifickymi portami.
        def_port    => [ "'ip IP and ( (src port > 6880 and src port < 6890)
                            or (dst port > 6880 and dst port < 6890) )'",
                         '-o "fmt:%sa" -q'],
```

```
        track_port  => [ "'src ip IP and dst port = 6969'",
                         '-o "fmt:%da" -q' ],

        # Toky s vysokymi portami, nad 10240 (pocet riadkov).
        high_port   => [ "'ip IP and src port > 10240 and dst port > 10240'",
                         '-o "fmt:%da" -q' ],

        # RPD - remote ports distribution
        # Vystup: pocet roznych vzdialenych portov a prislusnych tokov.
        # (TCP alebo UDP).
        rpd         => [ "'src ip IP and $port_filter'",
                         '-a -A proto,dstport -o "fmt:%pr %fl" -q -N' ]
);
```

Popis výpočtu skóre na základe hodnôt získaných z výstupov týchto dotazov je popísaný v následujúcej sekcii.

4.4 Identifikácia užívateľov protokolu BitTorrent

Identifikácia užívateľov protokolu BitTorrent prebieha na základe výpočtu skóre, pri ktorom sa vychádza z predchádzajúcej analýzy vlastností sieťových tokov v komunikácií protokolu BitTorrent (sekcia 4.2). Získané skóre kandidátnej adresy predstavuje pravdepodobnosť, s ktorou sa v sieťových tokoch tejto adresy nachádza aj komunikácia protokolu BitTorrent. Kandidátna adresa, ktorej získaná hodnota skóre prekročí zadanú hranicu (`s_limit`), je detekovaná ako užívateľ tohto protokolu.

Základom detekcie sú následujúce vlastností sieťových tokov. Sleduje sa niekoľko rôznych parametrov tokov a pre každý je určené tzv. *čiastkové skóre*. Tým je hodnota z rozsahu $0-10$ (reálne číslo). Táto hodnota vyjadruje mieru podobnosti s chovaním systému BitTorrent.

Výpočet jednotlivých hodnôt čiastkového skóre na základe vybraných vlastností sieťových tokov:

1. **Pomer používania portov nad 1024 (1024S)**

   ```
   pomer = porty_pod_1024 / porty_nad_1024;
   if ( 0 < pomer <= 1.0 ) skóre = 10 - pomer * 5;
   if ( 1.0 < pomer <= 2.0) skóre = (2.0 - pomer) * 10;
   ```

2. **Pomer používania portov nad 10240 (10240S)**

   ```
   pomer = porty_nad_10240 / porty_nad_1024;
   if ( 0 < pomer < 0.88 ) skóre = 10 - ((0.88 - pomer) * 12);
   if ( 0.88 <= pomer <= 1 ) skóre = 10 - ((pomer - 0.88) * 10);
   ```

3. **Neúspešné spojenia (FCS)**

   ```
   pomer_peeri = neuspesne_kontaktovany_opakovane / neuspesne_kontaktovany;
   pomer_toky = opakovane_neuspesne_pokusy / neuspesne_spojenia_celkom;
   avg = ( pomer_peeri + pomer_toky ) / 2;
   if ( avg < 1.0 ) skóre = avg * 10;
   if ( avg > 1.0 ) skóre = 10;
   ```

4. **Pomer vzdialených uzlov s konkurentým TCP a UDP spojením (TUS)**

```
pomer = uzly_TCP_i_UDP / uzly_celkom;
if ( pomer < 0.1 ) skóre = pomer * 100;
if ( pomer > 0.1 ) skóre = 10;
```

5. **Pomer sloních tokov a dát v nich (ES)**

```
pomer_tokov = slonie_toky / toky_nad_1024;
pomer_dat = slonie_data / data_celkom;
if ( 0 < pomer_tokov <= 0.1 ) skóre1 = 10 - pomer_tokov;
if ( pomer_tokov > 0.1 ) skóre1 = 10 - ((pomer_tokov - 0.1) * 12);
if ( pomer_dat >= 0.98 ) skóre2 = 10 - (pomer_dat - 0.98);
else skóre2 = 10 * pomer_dat;
skóre = ( skóre1 + skóre2 ) / 2;
```

6. **Distribúcia vzdialených portov RPD (RPDS)**

```
suma = tcp_rpd + udp_rpd;
if ( suma > 10) skóre = 10;
if ( suma < 10) skóre = suma * 1.5;
if ( skóre > 10) skóre = 10 ;
```

Celkové skóre kandidátnej adresy sa spočíta následovne:

```
max_sum = pocet_ciastkovych * 10;
celkove_skóre = ( suma_čiastkových / max_sum ) * 100;
```

V našom prípade je maximálna hodnota súčtu jednotlivých čiastkových hodnôt skóre 60 (`max_sum`). Tabuľka 4.9 zobrazuje hodnoty jednotlivých čiastkových i celkového skóre pre kolekcie NetFlow dát popísané v sekcií 4.1.

č.	**Skóre**						**Súčet (celkové skóre)**
	1024S	**10240S**	**FCS**	**TUS**	**ES**	**RPDS**	
1.	3.37	9.85	10	10	9.99	10	53.21 (88.68%)
2.	9.35	9.96	9.16	10	9.97	10	58.44 (97.40%)
3.	9.06	6.88	8.82	10	9.98	10	54.74 (91.23%)
4.	9.27	9.8	10	10	9.88	10	58.95 (98.25%)
5.	9.08	9.86	10	0.04	9.99	10	48.97 (81.62%)
6.	9.07	9.94	2.1	0.27	9.79	10	41.17 (68.62%)
7.	8.64	9.7	4.95	0.18	9.98	10	43.45 (72.42%)
8.	4.95	9.92	3.65	0.16	9.98	10	38.66 (*64.43%*)
9.	8.71	9.52	7.36	10	9.99	10	55.58 (92.63%)
10.	9.78	9.32	6.69	10	9.99	10	55.78 (92.97%)

Tabuľka 4.9: Výpočet skóre pre kolekcie záznamov obsahujúce komunikáciu BitTorrentu.

Z výsledkov tabuľky by sa dala stanoviť východzia hodnota hranice detekcie `s_limit` na číslo 64 (najnižšie celkové skóre). Avšak vzhľadom na to, že tieto výsledky boli namerané nad dátami obsahujúcimi celú bittorrentovú prevádzku behom sťahovania súboru, je nutné overiť či bude táto hranica detekcie postačujúca aj pri spracovávaní jednotlivých *nfcapd* súborov (zachytávajú záznamy o tokoch v päťminútových intervaloch). Práve nad takýmito

vzorkami dát bude prebiehať periodická automatizovaná detekcia v implementovanom plugine NfSen kolektoru.

V tabuľke 4.10 sú zaznamenané priemerné hodnoty celkového skóre získané z jednotlivých *nfcapd* súbor príslušnej kolekcie záznamov. Priemerné skóre $\bar{x}$ je dané vzťahom:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{4.2}$$

kde n je počet päťminútových intervalov (*nfcapd* súborov) obsiahnutých v príslušnej kolekcií dát a x_i je *celkové skóre* dosiahnuté v i-tom intervale.

č.	**Počet intervalov**	**Priemerné skóre ($\bar{x}$)**
1.	3	79.43
2.	3	95.24
3.	2	90.94
4.	2	64.59
5.	5	56.69
6.	4	62.31
7.	5	64.78
8.	5	*55.19*
9.	10	62.17
10.	6	83.45

Tabuľka 4.10: Priemerné hodnoty celkového skóre pri spracovaní kolekcií NetFlow dát po päťminútových intervaloch.

Na nameraných hodnotách je očividné, že sú nižšie ako v predchádzajúcom meraní. To je spôsobené veľmi nízkymi hodnotami skóre v intervaloch na začiatku, resp. na konci sťahovania, kedy síce bola bittorentová komunikácia prítomná v sieťovej prevádzke, ale prejavovala sa iba minimálne. Pre hranicu detekcie `s_limit` by sme mohli na základe výsledkov merania navrhnúť hodnotu 55.19, ktorá však ešte nie je konečná.

Na výpočte konečnej hodnoty skóre sa nepodieľajú zaznamenané toky so špecifickými portami (6881-6889, 6969). Na existenciu takýchto tokov sa prihliada iba v prípade, že celkové skóre je tesne pod hranicou `s_limit`. V takomto prípade je kandidátna adresa detekovaná ako užívateľ BitTorrentu v prípade, že overovacia funkcia vráti hodnotu `true`. Postup overovania je následovný:

```
if (0 <= pocet_tokov_6881_6889 < 5) tmp = pocet_tokov_6881_6889;
if (5 <= pocet_tokov_6881_6889) tmp = 5;

if (0 <= pocet_tokov_6969 < 5) tmp += pocet_tokov_6969;
if (5 <= pocet_tokov_6969) tmp += 5;

if ((skore_limit - celkove_skóre) < tmp ) return true;
else return false;
```

Po predstavení tejto možnosti korekcie detekcie som sa rozhodol stanoviť konečnú hranicu detekcie `s_limit` na hodnotu *60*. Táto hodnota je implicitným nastavením pluginu *BitTorrent Detector*, ktorý je predstavený v následujúcej kapitole.

Kapitola 5

Implementácia pluginu

Vytvorený plugin (zásuvný modul) kolektoru NfSen má dve časti – backendovú a frontendovú (sú to v podstate dva pluginy). Základný princíp fungovania týchto častí je naznačený v sekcií 3.5.3. Ďalšie podrobnosti o nutnej základnej štruktúre oboch pluginov sú dostupné v špecifikácií NfSenu [1]. V tejto kapitole si predstavíme oba časti implementovaného pluginu *BitTorrent Detector* (btdetect), ich funkcionalitu a rozhranie. Pri vytváraní pluginu bol použitý nástroj pre podporu tejto činnosti – `nfPlugger`[1].

5.1 Backendový plugin

Je napísaný v jazyku Perl a jeho jadro tvorí skript popísaný v predchádzajúcej kapitole, ktorý je doplnený o nutné funkcie so špecifickým účelom (inicializácia, periodické spracovanie dát, atď.) a ďalšie náležitosti (verzia NfSenu, pre ktorý je plugin určený).

Pre konfiguráciu pluginu je využitá možnosť nastavenia parametrov v konfiguračnom súbore NfSenu, ktorý umožňuje konfigurácou všetkých pluginov na jednom mieste. Parametre pluginu sú zapísané do asociatívneho poľa `%PluginConf` takto:

```
%PluginConf = (
    btdetect => {                   # meno pluginu
        s_limit  =>  60,            # potrebné skóre
        ip_limit =>  10,            # max. počet kandidátnych adries
        subnet   =>  '192.168/16',  # sledovaná podsieť
    },
);
```

Rozhranie pre komunikáciu s frontendovým pluginom predstavuje komunikačný kanál `nfsend.comm`. Typicky sa vždy jedná o odpoveď (zaslanie výsledkov) na požiadavok frontendového pluginu.

Funkcionalitu backendového pluginy predstavuje perodické spracovanie nových dát, alebo spracovanie dát na vyžiadanie (inicializované užívateľom cez frontend).

Výsledky periodickej detekcie sa zaznamenávajú do súboru `btdetect.log`. Tieto informácie využíva frontedový plugin a sú prístupné prípadným ďalším pluginom pre periodické zasielanie informácií o P2P aktivite (protokolu BitTorrent) v sledovanej sieti na vybranú emailovú adresu[2].

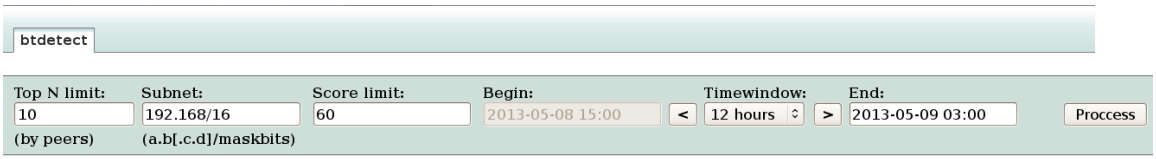
[1] https://nftest.ics.muni.cz/nfplugger/

[2] Takýto plugin nebol implementovaný. Avšak v prípade potreby by sa vlastne jednalo o čiste backendový plugin využívajúci dostupné moduly `Mail::Header` a `Mail::Internet`.

5.2 Frontendový plugin

Jedná sa o dynamické webové stránky napísané v jazyku PHP. Toto webové rozhranie backendového pluginu má tri základné časti:

1. **Hlavný panel** je zobrazený na obrázku 5.1. Umožňuje výber časového úseku, nad ktorým sa ma vykonať detekcia užívateľov BitTorrentu.[3] Zároveň je možné nastavenie sledovanej podsiete a potrebnej hranice skóre, pri ktorej bude kandidátna adresa identifikovaná ako užívateľ BitTorrentu.



Obrázok 5.1: Hlavný panel pluginu.

2. **Podrobnosti o detekovaných IP adresách**, ktoré obsahujú údaje použité pri výpočte skóre a ďalšie informatívne hodnoty (množstvo prenesených dát). Každá adresa predstavuje jeden riadok tabuľky. Ukážka podrobnosti o sieťových tokoch niekoľkých užívateľov je na obrázku 5.2 (vzhľadom na formát tejto práce bola tabuľka rozdelená na dve časti.). Táto tabuľka je spracovaným výstupom backendového pluginy, ktorý odpovedal na požiadavok zadaný v hlavnom paneli.

IP address	Peers	Peers (concurrent TCP, UDP)	Default ports (flows)		Flows (by ports)			Failed connections (ratio)	
			6881-6889	tracker	>1024	<1024 (ratio)	>10 240 (ratio)	peers	flows
192.168.1.9	388	97	502	8	3402	731 (0.21)	1698 (0.50)	1.00	1.00
192.168.1.13	636	90	145	10	2118	770 (0.36)	1322 (0.62)	0.96	0.99
192.168.1.11	1976	3	349	12	5315	718 (0.14)	4404 (0.83)	0.83	0.93
192.168.1.14	316	83	170	16	2403	591 (0.25)	1950 (0.81)	0.33	0.74

Elephants		RPD		Traffic (Bytes)			SCORE
flows (ratio)	traffic (ratio)	TCP	UDP	IN	OUT	ALL	
218 (0.06)	1.72 G (1.00)	0.66	1.51	1.68 G	36.04 M	1.72 G	79.28
147 (0.07)	825.92 M (1.00)	1.02	4.80	814.60 M	12.80 M	827.40 M	89.26
162 (0.03)	858.54 M (1.00)	1.41	6.64	844.65 M	16.90 M	861.55 M	79.44
343 (0.14)	838.84 M (1.00)	0.68	2.17	823.39 M	17.88 M	841.27 M	78.85

Obrázok 5.2: Podrobnosti o tokoch detekovaných užívateľov BitTorrentu.

3. **Detekované aktivity BitTorrentu** vo forme tabuľky. Hodnoty v tabuľke odpovedajú záznamom v súbore `btdetect.log`. Pre zobrazenie podrobnosti o vybranej adrese (resp. záznam podozrivej aktivity v danom čase) zo zoznamu je možné použiť hlavný panel. Na obrázku 5.3 je vidieť, že sa zaznamenávajú aj čiastkové hodnoty skóre z predchádzajúcej kapitoly.

[3] Pokiaľ v zadanom časovom intervale chýbajú príslušné *nfcapd* súbory, napríklad z dôvodu výpadku kolektoru, nebude nikdy detekcia úspešná, pretože príslušný nfdump dotaz skončí s chybou.

Detected BitTorrent activities:

Timeslot	IP address	Subscore						Sum (Total score)
		1024S	10240S	FCS	TUS	ES	RPDS	
201305090140 (2013-05-09 01:40)	192.168.1.9	9.24	6.66	8.70	10.00	9.67	9.23	53.5 (89.17 %)
201305090145 (2013-05-09 01:45)	192.168.1.9	9.20	4.97	3.07	10.00	9.52	6.76	43.52 (72.54 %)
201305090150 (2013-05-09 01:50)	192.168.1.9	9.57	3.52	5.67	10.00	9.61	6.47	44.84 (74.73 %)
201305090220 (2013-05-09 02:20)	192.168.1.11	9.79	8.61	1.74	0.71	9.97	10.00	40.82 (68.02 %)
201305090225 (2013-05-09 02:25)	192.168.1.11	9.78	9.67	1.43	0.08	9.99	10.00	40.95 (68.26 %)
201305090225 (2013-05-09 02:25)	192.168.1.14	9.62	9.45	4.41	10.00	9.39	6.47	49.34 (82.23 %)
201305090230 (2013-05-09 02:30)	192.168.1.11	9.77	9.35	1.66	0.19	9.97	10.00	40.94 (68.23 %)
201305090230 (2013-05-09 02:30)	192.168.1.14	9.33	8.93	3.75	6.59	8.63	4.56	41.79 (69.64 %)
201305090235 (2013-05-09 02:35)	192.168.1.13	9.35	7.43	7.77	10.00	9.95	10.00	54.5 (90.83 %)
201305090240 (2013-05-09 02:40)	192.168.1.13	9.40	4.44	4.24	10.00	9.76	8.00	45.84 (76.41 %)

Obrázok 5.3: Zoznam detekovaných aktivít protokolu BitTorrent.

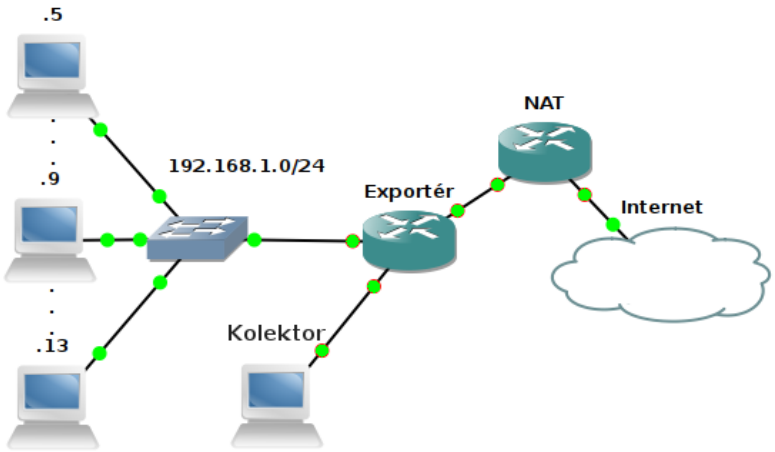
Kapitola 6

Testovanie a možnosti rozšírenia

6.1 Metodika testovania a výsledky

Implementovaný plugin bol nasadený v prostredí virtuálneho serveru s kolektorom FlowMon od firmy Invea-Tech, ktorý bežal na operačnom systéme CentOS. Systém sa nepodarilo nasadiť do siete s reálnou prevádzkou preto bol pri overovaní funkčnosti pluginu použitý alternatívny postup:

1. V počítačovom laboratóriu bola vytvorená testovacia sieť, ktorej prevádzka bola zaznamenávaná do NetFlow záznamov. Ako exportér bol použitý smerovač. Znázornenie zapojenia je na obrázku 6.1 (jedna sa o tradičnú architektúru zapojenia, viď sekciu 3.5.1).



Obrázok 6.1: Schéma zapojenia zariadení v testovacej sieti.

2. Na počítačoch v tejto sietí bola generovaná bittorentová a iná prevádzka, u ktorej bolo podozrenie, že by mohlo dojsť k falošnému poplachu (*false positive*). Popis generovanej prevádzky je v tabuľke 6.1 (pri generovaní bittorrentovej prevádzky boli použitý klienti Vuze, Transmission a BitTorrent[1]). Rôzne typy prevádzky neboli kombinované na základe predpokladu, že užívateľ bittorentového klienta nepoužíva ďalšie aplikácie (stream, online hry viacerých hráčov), ktorých kvalita odozvy by bola ovplyvnená

[1] http://www.bittorrent.com/ verzia 7.8

zníženou dostupnosťou šírky pásma. Vytvorené NetFlow záznamy tejto prevádzky boli exportované na kolektor (`nfcapd` démon), ktorý bežal na vlastnom notebooku (záznamy sú priložené na CD v prílohe).

3. Vytvorené *nfcapd* súbory boli následne skopírované na virtuálny server. Aby boli zaradené do hierarchie súborov kolektoru v prostredí virtuálneho serveru, bol použitý nástroj `nfreplay`, ktorým boli NetFlow dáta z našich súborov nahrané na kolektor vo virtuálnom prostredí.

4. Nad týmito dátami bola overená funkčnosť detektoru protokolu BitTorrent.

IP adresa	**Typ prevádzky**	**Program (klient)**	**Počet detekcií**
192.168.1.5	streaming	VLC	0
192.168.1.8	multiplayer hra	League of Legends	0
192.168.1.9	BitTorrent	BitTorrent	3
192.168.1.11	BitTorrent	Vuze	3
192.168.1.13	BitTorrent	BitTorrent	2
192.168.1.14	BitTorrent	Transmission	2

Tabuľka 6.1: Popis generovanej prevádzky v laboratóriu a úspešnosť detekcie.

Z výsledkov tabuľky vyplýva, že systém úspešne detekoval bittorrentovú prevádzka (všetkých užívateľov) opakovane, vo viacerých päť minútových intervaloch, počas ktorých dochádzalo k najintenzívnejšiemu zdieľaniu (sťahovaniu) súborov. Podrobnosti o konkrétnych hodnotách sú na obrázkoch 5.2 a 5.3 v predchádzajúcej kapitole. Na druhej strane nedošlo k žiadnemu falošnému poplachu.

Plugin bol dodatočné otestovaný na úspešnosť rozpoznanie prevádzky generovanej najznámejším bittorrentovým klientom uTorrent[2], ktorú rovnako ako v predchádzajúcom prípade úspešne identifikoval. Zároveň bol konfrontovaný s najväčším kandidátom na vyvolanie falošného poplachu – prevádzkou P2P TV, ktorá bola taktiež úspešne nedetekovaná. Porovnanie dosiahnutých čiastkových skóre pre uTorrent a P2P TV je v tabuľke 6.2.

Aplikácia	**Skóre**						**Súčet (celkové skóre)**
	1024S	**10240S**	**FCS**	**TUS**	**ES**	**RPDS**	
uTorrent	9.24	6.66	8.70	10	9.67	9.23	53.50 (89.17%)
P2P TV	0.00	2.35	0.00	0.00	7.69	10	20.04 (33.40%)

Tabuľka 6.2: Skóre prevádzky generovanej BitTorrentom (klient uTorrent) a P2P TV.

Z výsledkov tabuľky vyplýva, že hoci aplikácia P2P TV konkuretne komunikuje s viacerými uzlami P2P siete, udržiava narozdiel od BitTorrentu podstatne stabilnejšie a dlhodobejšie spojenia. To sa prejaví poklesom neúspešných spojení a nižším počtom tokov na vysokých portoch. Zároveň neudržiava konkurentné TCP a UDP spojenie s rovnakým uzlom. To všetko sa prejavilo v nízkom celkovom skóre.

[2] `http://www.utorrent.com/` verzia 2.0.2

Zhrnutie výsledkov

Na základe výsledkov vytvorených nad dostupnými dátami, sme došli k záveru, že navrhnutý systém dokáže automatizovane a korektne detekovať protokol BitTorrent v sieťovej prevádzke. Zároveň je systém založený na metóde (sekcia 4.3.3), ktorej výkonnostne testy nám dokázali, že dokáže spracovávať relatívne veľké množstvo NetFlow v reálnom čase pri správnej konfigurácií (maximálny počet kandidátnych adries). Výsledky potvrdili správnosť sledovaných parametrov tokov pri samotnom návrhu systému.

6.2 Možnosti rozšírenia a ďalšieho postupu

Možnosti rozšírenia pluginu spočívajú v sledovaní ďalších charakteristických vlastností prevádzky protokolu BitTorrent (veľkosti paketov v TCP alebo UDP sloních tokoch, resp. sledovanie UDP paketov mechanizmu DHT používaných k lokalizácií uzlov [3]) a rozšírenia popisu chovania. Toto by však viedlo k nárastu zložitosti a zvýšeniu potrebného počtu nfdump dotazov pre každú kandidátnu adresu. Počet dotazov je už teraz pomerne vysoký (konštantne 10 na kandidátnu adresu). Z čoho vyplýva ďalšia možnosť rozšírenia, a to výpočet prednostne čiastkových skóre, ktoré majú typicky vždy najlepšie výsledky (napr. EFS, RPDS). Pokiaľ by získaná hodnota nedosiahla požadovaný limit tak by sa vôbec nepokračovalo ďalšími dotazmi, čím by sa zredukovala časová náročnosť.

Pri ďalšom postupe na práci sa ako zaujímavá možnosť javí využitie strojového učenia k popisu modelu a klasifikácií chovania BitTorrentu v sieťovej prevádzke. Pre ďalšie pokračovanie v práci považujem tiež sa nevyhnutné odskúšanie vytvoreného systému na reálnej sieti.

Kapitola 7

Záver

Táto práca sa zaoberala možnosťou detekcie P2P sietí na základe analýzy sieťových tokov a následnou detekciou užívateľov vybraného P2P systému, ktorých sieťová prevádzka odpovedá popisu chovaniu sledovaného P2P protokolu. Na začiatku práce boli predstavené základné princípy a vlastnosti P2P sietí. Podrobnejšie bol popísaný najrozšírenejší P2P systém pre zdieľanie súborov – BitTorrent, na detekciu ktorého sa zameriava táto práca. Ďalej boli predstavené ďalšie metódy detekcie P2P sietí a porovnanie ich výhod a nevýhod oproti metóde založenej na vytvorení popisu chovania z charakteristických vlastností P2P aplikácie v sieťovej komunikácií. Táto metóda využíva informácie o sieťových tokoch bez znalosti obsahu paketov, v súvislosti s čím bola predstavená technológia NetFlow.

V jadre práce boli popísané a analyzované charakteristické vlastnosti chovania protokolu BitTorrent v sieťovej komunikácií, z ktorých bol vytvorený popis chovania použitý pri detekcií užívateľov tohto protokolu. Platnosť tvrdení o jednotlivých vlastnostiach bola overená na kolekciách NetFlow záznamov o tokoch, v ktorých bola zachytená bittorrentová prevádzka. Na základe analýzy boli vytvorené tri metódy detekcie, ktoré pristupovali rôznymi spôsobmi k výberu kandidátnych adries (podozrivých na bežiaceho BitTorrent klienta) a využitiu vytvorených nfdump dotazov pre zisk potrebných štatistik o tokoch kontrolovanej adresy. Všetky metódy vykazovali dobré výsledky pri detekcii BitTorrentu, avšak vzhľadom na pomer časovej a priestorovej náročnosti bola ako jadro vytvoreného systému použitá posledná. Táto metóda vyberá kandidátne adresy na základe zvýšenej konektivity (počet unikátnych vzdialených uzlov, s ktorými kandidátna adresa komunikuje súčasne).

Testovanie a overenie funkčnosti vytvoreného systému detekcie (plugin kolektoru NfSen – *FlowMon kolektor* firmy Invea-tech) prebiehalo v rámci testovacej siete vytvorenej v počítačovom laboratóriu, keďže sa nepodarilo získať prístup k pasívnej sonde v reálnej sieti. Z výsledkov testovania nad dostupnými dátami, bolo zistené, že systém úspešne detekuje aktivitu BitTorrentu v sieťovej prevádzke bez výskytu falošných poplachov.

Možnosti rozšírenia nástroja spočívajú v zaradení ďalších charakteristických vlastností (napríklad veľkosti paketov a UDP toky DHT mechanizmu navrhnuté Ahmedom Bashirom [3]) komunikácie BitTorrentu do vytvoreného popisu chovania a optimalizácie použitej metódy detekcie. Súčasťou ďalšieho postupu na práci by malo byť overenie funkčnosti nasadením do reálnej siete a pravidelne aktualizácie popisu chovania pre novšie verzie sledovaného protokolu pre zvýšenie presnosti detekcie a zníženie rizika chybnej klasifikácie.

Obmedzením vytvoreného systému detekcie, resp. použitej metódy využívajúcej popis chovania sledovanej aplikácie v sieťovej prevádzke, je úzke zameranie na konkrétnu aplikáciu. Pre klasifikáciu ďalších P2P systémov je potrebné navrhnúť iné metódy a zameriavať sa na charakteristické vlastnosti príslušnej P2P aplikácie.

Literatura

[1] *NfSen - Netflow Sensor* [online]. 2011 [cit. 2013-01-22]. Dostupné z: <http://nfsen.sourceforge.net/>.

[2] *Cisco IOS NetFlow* [online]. [cit. 2013-01-22]. Dostupné z: <http://www.cisco.com/web/go/netflow>.

[3] Bashir, A. *Classifying P2P Activities in Netflow Records: A Case Study (BitTorrnet & Skype).* Carleton University, Engineering, Electrical and Computer, 2012. Disertační práce.

[4] Buford, J., Yu, H. a Lua, E. K. *P2P Networking and Applications.* Burlington: Morgan Kaufmann, 2009. ISBN 978-0-12-3742148.

[5] Cheng, W.-q., Gong, J. a Ding, W. Identifying file-sharing P2P traffic based on traffic characteristics. *The Journal of China Universities of Posts and Telecommunications.* 2008, roč. 15, č. 4. s. 112–120.

[6] Claise, B. *Cisco Systems NetFlow Services Export Version 9. RFC 3954.* 2004.

[7] Cohen, B. *The BitTorrent Protocol Specification* [online]. 2008 [cit. 2013-01-21]. Dostupné z: <http://www.bittorrent.org/beps/bep_0003.html>.

[8] Collins, M. P. a Reiter, M. K. Finding peer-to-peer file-sharing using coarse network behaviors. In *Proceedings of the 11th European conference on Research in Computer Security.* Berlin, Heidelberg: Springer-Verlag, 2006. s. 1–17. ESORICS'06.

[9] Elich, M. *Rozšíření NetFlow kolektoru NfSen o detekci síťových anomálií.* Brno: Masarykova univerzita, Fakulta informatiky v Brně, 2009. Diplomová práce.

[10] Haag, P. *User Documentation nfdump & NfSen* [online]. [cit. 2013-01-22]. Dostupné z: <http://www.first.org/conference/2006/papers/haag-peter-papers.pdf>.

[11] Hadvig, J. *Identifikácia protokolu BitTorrent v sieťovej prevádzke.* Brno: Masarykova univerzita, Fakulta informatiky, 2010. Bakalářská práce.

[12] Hlavatý, I. *Ochrana datové sítě s využitím NetFlow dat.* Brno: FIT VUT v Brně, 2011. Diplomová práce.

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

[14] Liang, C. a Jian, G. Analysis of BitTorrent Flow Behavior on Large-Scale Networks. *Journal of Southeast University (Natural Science Edition).* 2008, roč. 38. s. 390–395.

[15] Loewenstern, A. *DHT Protocol* [online]. 2008 [cit. 2013-04-22]. Dostupné z: <http://www.bittorrent.org/beps/bep_0005.html>.

[16] Schulze, H. a Mochalski, K. *Internet Study 2008/2009* [online]. 2009 [cit. 2013-02-28]. Dostupné z: <http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf>.

[17] Silvester, J. *Analyzátor spamu a p2p toku dat.* Praha: Fakulta elektrotechnická, České vysoké učení technické v Praze, 2011. Bakalářská práce.

[18] Soldani, C. *Peer-to-Peer Behaviour Detection by TCP Flows Analysis.* University of Liege, Computer Sciences, 2004. Disertační práce.

[19] Spek, O. van der. *UDP Tracker Protocol for BitTorrent* [online]. 2008 [cit. 2013-04-28]. Dostupné z: <http://www.bittorrent.org/beps/bep_0015.html>.

[20] Vrba, J. *Detekce P2P sítí.* Brno: FIT VUT v Brně, 2009. Bakalářská práce.

[21] Wikipedia. *NetFlow* [online]. 2009 [cit. 2013-01-21]. Dostupné z: <http://en.wikipedia.org/wiki/NetFlow>.

[22] Wikipedia. *BitTorrent (protocol)* [online]. 2013 [cit. 2013-01-21]. Dostupné z: <http://en.wikipedia.org/wiki/BitTorrent>.

[23] Čegan, J. *Ochrana datové sítě s využitím NetFlow dat.* Brno: FIT VUT v Brně, 2009. Bakalářská práce.

Příloha A

Obsah CD

- Text práce vo formáte PDF a zdrojové súbory tejto práce.
- Zdrojové súbory implementovaného systému (vrátane pokynov k inštalácií).
- Kolekcie NetFlow záznamov obsahujúcich komunikáciu protokolu BitTorrent.
- Testovacie dáta (Netflow záznamy obsahujúce bittorentovú a inú prevádzku).