VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

VYHLEDÁVÁNÍ PODOBNOSTÍ V SÍŤOVÝCH DATECH

SIMILARITY SEARCHING IN NETWORK DATA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **JAKUB HUD**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAN WRONA**
SUPERVISOR

BRNO 2018

Abstrakt

Tato práce se zabývá analýzou záznamů o IP tocích. Záznam obsahuje metadata o IP toku na dané síti, jako IP adresy, čísla portů, čísla komunikačních protokolů a další. Cílem je návrh a implementace metrik pro určování podobností mezi NetFlow záznamy. V první části práce je popis analýzy velkého množství dat. Dále jsou předvedeny techniky monitorování sítě a technologie NetFlow. Další části práce jsou věnovány návrhu a implementaci analýzy dat pomocí DBSCANu. Součástí práce je implementace aplikace pro analýzu síťových metadat. Výslednou aplikaci je možné použít pro detekci skenování sítě v NetFlow datech, ale výsledky nejsou jednoznačné, zejména obsahují i záznamy, které k útoku nepatří.

Abstract

This bachelor thesis is interested in analyzing IP flow records. IP flow record contains IP flow metadata of specific network communication such as IP addresses, port numbers, network protocol numbers and other. Main goal is to design and implement metrices to determine similarity of NetFlow records. At the beginning of this thesis is description of how to analyze great amount of data. Next there are shown network monitoring technicies and NetFlow. Other parts of this thesis are dedicated to design and implementation of data analysis using DBSCAN algorithm. Implementation of data analysis application is also part of this thesis. As a result, the application can be used to network scan detection using NetFlow data although the results are not very clear and contain a lot of legitimate communication.

Klíčová slova

analýza dat, analýza síťových dat, data mining, DBSCAN, NetFlow„ skenování portů

Keywords

data analysis, network data analysis, data mining, DBSCAN, NetFlow, port scanning

Citace

HUD, Jakub. *Vyhledávání podobností v síťových datech.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jan Wrona

Vyhledávání podobností v síťových datech

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jana Wrony. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jakub Hud
16.5.2018

Poděkování

Děkuji především vedoucímu mé práce Ing. Janu Wronovi, za věnovaný čas a trpělivost. Dále bych chtěl poděkovat rodičům za jejich neustálou podporu.

Obsah

Kapitola 1

Úvod

Přes internet je přenášeno stále větší množství dat. Když je potřeba síť rozšířit, síť přestane fungovat nebo dojde k nějakému útoku, je potřeba mít monitorovací nástroje. Na páteřních linkách, kde protéká obrovské množství provozu, také vzniká velké množství metadata. Tato data se používají ke kontrole stavu linek, jestli nedochází k přetížení, jestli není linka pod nějakým útokem nebo jestli není komunikace jakkoli nezvyklá a není potřeba zasáhnout. Protože nasbíraných metadat je opravdu hodně, je jejich analýza velmi náročná. Z toho důvodu se neustále vyvíjejí co nejefektivnější analyzační techniky. Existují různé přístupy analýzy dat na různých úrovních. Nejdetailnější analýza, která pracuje jak s metadaty, tak i samotnými daty, dokáže odhalit téměř jakoukoli hrozbu nebo anomálii. Pracovat však se všemi daty v reálném čase není možné, protože dat je příliš mnoho a výpočetní náročnost není únosná. Proto je velmi důležité vyvíjet analyzační techniky, které pracují jen s metadaty, a to ještě třeba vzorkovanými, a přesto dokáží odhalit důležité události na síti.

Tato práce se nezabývá statistickým zpracováním síťových metadat, nýbrž hledáním podobností mezi datovými toky. Záznamy o tocích lze seskupovat do skupin podle vzájemné podobnosti. Díky tomuto přístupu je možné určit jaký charakter komunikace je typický pro určitá zařízení na síti. Zařízení v síti může být např. server nebo pracovní počítač uživatele. Dále lze říci jaké typy komunikace se na síti v daném čase vyskytují i jakou tvoří část z veškerého datového provozu na síti. Přestože předmětem analýzy jsou pouze metadata – záznamy o datových tocích, jedná se o velké množství informací. Nasbíraná metadata za několik hodin už mohou snadno dosáhnout velikosti několika desítek GB. Zpracování takového množství dat je výpočetně, a tedy i časově náročné. Proto je při návrhu analyzačních metrik třeba brát ohled na výpočetní složitost algoritmů.

Nejpodstatnější částí této práce je návrh a implementace metrik pro hledání podobnosti mezi záznamy o datových tocích. Protože komunikace na každé síti může mít jiný charakter je nezbytné, aby bylo možné proces hledání podobností parametrizovat. Celý proces musí být navrženy tak, aby při analýze různých vzorků dat s rozdílnými rozsahy hodnot byly výsledky smysluplné. Velmi také záleží, co přesně se v datech hledá nebo očekává.

Kapitola Data Mining 2 se zabývá možnostmi zpracování velkého množství dat, jakým způsobem je možné efektivně získat určitý přehled o datech ve zkoumaném vzorku. A také popisu metody shlukové analýzy DBSCAN. V další kapitole Monitorování sítě 3 jsou představeny monitorovací techniky a nástroje pro zachytávání síťových dat a metadat a pro monitorování provozu v reálném čase. V části Sledované vlastnosti datových toků 4 už jsou popsány detailně vlastnosti NetFlow záznamů, se kterými se v této práci pracuje. V kapitole Vybrané síťové útoky 5 je přehled několika nejznámějších útoků, které by mělo jít

při analýze síťových metadat odhalit. V kapitole Aplikace pro hledání podobností v síťových datech 6.1 je popis návrhu analyzační aplikace a v podkapitole Dosažené výsledky 6.3 je ukázka a zhodnocení aplikace.

Kapitola 2

Data mining

Rychlost generování stále nových dat se neustále zvyšuje. Tento trend souvisí se stálým rozšiřováním internetu a informačních technologií obecně po celém světě. Se zvětšujícími se kapacitami úložných médií je sběr a ukládání velkého množství dat jednodušší než kdy dříve. Pod pojmem Data mining se rozumí získávání hlubší znalosti ze souboru dat, kterou je možné dále využít.

Data mining je široky pojem. Jde o celý proces analýzy velkého množství dat. Vstupní data mohou být různorodá. Podle typu dat i toho jak (ne)jsou strukturovaná je potřeba navrhnout analyzační postup. Pro analýzu se nemusí využít všechny informace obsažené v rozsáhlém souboru dat. Takže je nutné určit jaké části analyzovaných dat jsou relevantní a jaké ne. Neméně důležitý je návrh/výběr analyzační metody. Cílem je typicky najít trendy, anomálie nebo také rozřadit data do skupin podle podobnosti. Konkrétně může Data mining použít např. obchodní řetězec, aby zjistil jak moc se prodává jaké zboží a v jaký den v týdnu se prodává nejvíce.

Analýza, a tedy lepší pochopení nasbíraných dat, je velmi obtížná úloha. Abychom analýzou dat získaly užitečné informace je potřeba mít správné metody na zpracování dat konkrétního charakteru a také mechanismy na ověření správnosti získaných výsledků. Dalším problémem je objem dat. Čím větší vzorek dat máme k dispozici tím přesnější výsledky můžeme očekávat. Automatické zpracování velkého množství dat, je ale náročné na výpočetní výkon i paměť. Ještě další komplikace přináší vysoká složitost nasbíraných dat, která nejsou tvořena jen jednoduchými hodnotami, nýbrž mají komplexní hierarchii, kde je potřeba brát ohled na různorodou vnitřní sémantiku (význam) různých částí dat.

Přes všechny uvedené i další překážky je Data mining hojně využíván, protože dokáže odhalit nové užitečné informace, které jsou v datech skryty. Navíc z časových důvodů nelze rozsáhlé soubory dat analyzovat ručně. V této práci je Data mining použit pro hledání specifických vzorů v síťovém provozu. Tato kapitola vysvětluje princip Data miningu a představuje některé analyzační metody.

2.1 Metrické prostory

V metrických prostorech se pracuje s množinou bodů a metrikou (vzdálenostní funkcí), která počítá vzdálenost mezi každými dvěma body. O datech můžeme přemýšlet jako o souřadnicích bodů. Pomocí vzdálenostní funkce je pak možné zkoumat jaké body jsou si blíže a jaké jsou od sebe dále, tzn. hledáme v datech trendy, anomálie, shluky – provádíme Data mining.

Definice 1 *Metrický prostor je tvořen množinou X a reálnou funkcí $d : X \times X \to \mathbb{R}_+$ splňující následující vlastnosti.*

(M1) d je nulová právě tehdy, když $x = y : d(x, y) = 0 \equiv x = y$.

(M2) d je symetrická: $d(x, y) = d(y, x)$

(M3) d splňuje trojúhelníkovou nerovnost: $d(x, y) + d(z, y) \leq d(x, z)$

Takovou funkci d nazýváme metrikou, vzdálenostní funkcí, nebo jen vzdáleností [28].

Pro jednu množinu X může existovat více metrik. Příkladem jednoduché metriky je euklidovská vzdálenost.

2.2 Datové objekty

Datové objekty jsou abstrakcí reálných objektů. Datovým objektem může být např. pacient nebo zaměstnanec v databázi nemocnice, nebo kniha v knihovně. Při analyzování dat se hledají zajímavé vzory nebo závislosti právě mezi datovými objekty. Každý datový objekt je tvořen atributy. Atributy mohou být různých typů [25]:

- Kategorické atributy - Hodnotami kategorických atributů jsou názvy nebo symboly. V informačních technologiích se o takových hodnotách říká, že jsou výčtového typu. Hodnoty tohoto typu nelze nijak smysluplně seřadit. Kategorickým atributem může být např. *barva vlasů* u datového objektu *osoba*. A hodnoty kterých může nabývat mohou být *černá*, *hnědá*, *blond*.

- Binární atributy - Mohou nabývat jen dvou různých hodnot, typicky 0 nebo 1. Hodnota 0 může značit, že daný atribut chybí, hodnota 1 že je přítomen. Například datový objekt *pacient* může mít binární atribut *kuřák*, který nabývá hodnoty 1, když pacient kouří a 0 pokud nekouří.

- Numerické atributy - Jedná se o kvantitativní atributy, které jsou vyjádřeny celými nebo reálnými čísly. Rozlišují se dva typy numerických atributů interval-scaled a ratio-scaled.

  - Interval-scaled – Hodnoty mohou být kladné, záporné nebo nulové. Hodnoty je možné porovnávat a určit i jejich rozdíl. Co však nelze je porovnávat poměrnou velikost hodnot. Mějme například numerický atribut *teplota* s Celsiovou stupnicí. Bude-li jeden objekt mít teplotu 10°C a jiný objekt teplotu 5°C není problém tyto hodnoty porovnat ani určit o kolik se liší. Nula na Celsiově stupnici souvisí jen s bodem tuhnutí vody. Nula nevyjadřuje že by objekt neměl žádnou tepelnou energii nebo snad žádnou teplotu, proto může také mít i zápornou teplotu. Z tohoto důvodu nemůžeme říct že objekt s teplotou 5°C je dvakrát chladnější než objekt s teplotou 10°C. Co teprve kdyby měl jeden z objektů teplotu zápornou.
  - Ratio-scaled – Hodnoty ratio-scaled atributů mohou být kladné nebo nulové. Hodnoty mezi sebou lze porovnávat i spočítat jejich rozdíl. Na rozdíl od interval-scaled atributu můžeme porovnat i poměry mezi hodnotami tj. můžeme určit kolikrát je jedna hodnota větší/menší než jiná. Typickým příkladem je počet jednotek, třeba počet předmětů v nákupním košíku. Má-li jedna osoba v košíku

2 předměty a druhá osoba 6 předmětů, dává smysl říci, že druhá osoba má v košíku třikrát více předmětů než první.

Atributy lze dělit i jinými způsoby – obecnějšími nebo specifičtějšími. Atributy jsou zde rozdělené s ohledem na to jak se s nimi v této práci pracuje.

2.2.1 Podobnost

Podobnost objektů vyjadřuje, do jaké míry si jsou objekty podobné. Typicky nabývá reálných hodnot od 0 do 1. Přičemž hodnota 0 vyjadřuje, že jsou objekty maximálně rozdílné a hodnota 1, že jsou identické. Analogicky lze určit rozdílnost objektů, která nabývá hodnoty 0 pokud jsou objekty identické a hodnoty 1 jsou-li maximálně rozdílné. Podobnost lze jednoduše převést na rozdílnost a naopak, např. pokud je podobnost dvou objektů 0.3, jejich rozdílnost je 1 - 0.3 = 0.7. Z tohoto důvodu nezáleží na tom, zda vypočteme podobnost nebo rozdílnost datových objektů.

Pro určení rozdílnosti datových objektů je potřeba nějak porovnat jejich atributy. Můžeme porovnat všechny atributy nebo jen vybrat takové, které jsou pro analýzu zajímavé. Je zřejmé, že způsob porovnání atributů bude záležet na jejich typu. Výsledná rozdílnost objektů je pak určena z rozdílů jejich jednotlivých atributů. Pokud porovnáváme objekty, které mají všechny atributy stejného typu rozdílnost jde vypočítat podle jednoduchého vzorce. V praxi, a i v této práci, se ale setkáváme s objekty, které mají atributy různých typů. V takovém případě je potřeba vypočítat rozdílnost hodnot každého atributu zvlášť a tyto dílčí rozdílnosti pak promítnout do výpočtu celkové rozdílnosti mezi objekty.

U hodnot atributu kategorického typu můžeme porovnávat jen jestli jsou hodnoty stejné nebo rozdílné. Rozdílnost tak bude 0 nebo 1. Může se také stát, že nechceme jen porovnávat, jestli jsou kategorické hodnoty stejné nebo ne, ale pro některé dvojice hodnot bude rozdílnost vyšší než pro jiné. V takovém případě bude výsledkem porovnání hodnota z rozsahu 0-1, vyjadřující rozdílnost hodnot porovnávaných atributů. Například můžeme porovnávat hodnoty atributu *barva vlasů* s tím, že nám bude částečně záležet i na odstínu. Rozdílnost barvy *hnědá* a *černá* bude třeba 0.6, protože sice jde o různé barvy, ale jsou obě tmavé, zatímco rozdílnost barev *černá* a *blond* bude maximální – 1 [25].

Porovnání interval-scaled numerických atributů není pro tuto práci důležité, takže se podíváme jen na ratio-scaled atributy. Vždy když bude v této práci zmíněn numerický atribut bude se tím myslet ratio-scaled. Nejjednodušší přístup k určení rozdílnosti hodnot numerického atributu je namapovat rozsah hodnot atributu ve zkoumaném vzorku na interval <0,1>.

$$d(i,j) = \frac{d(i,j) - min(X)}{max(X) - min(X)}$$

kde $d(i,j)$ je rozdílnost hodnot atributu objektů i a j, $min(X)$ a $max(X)$ je minimální a maximální hodnota atributu ve zkoumaném vzorku datových objektů.

Toto mapování je ale triviální a problém nastává, když je maximální hodnota příliš velká oproti běžným hodnotám atributu. Například pro atribut *velikost* měřenou v bajtech u objektu reprezentujícího datový soubor se snadno stane, že většina souborů bude mít velikost v řádu několika kB, ale najde se pár souborů s velikostí několik GB. Takže maximální velikost bude několik miliard bajtů. Z tohoto důvodu dojde k namapování intervalu nula až několik miliard na interval 0-1 a malé rozdíly ve velikosti budou úplně zanedbané. Při

porovnání např. souborů o velikostech 100 kB a 400 kB vyjde jejich rozdílnost téměř nulová, protože rozdíl 300 kB je nepatrný vůči maximální hodnotě několika miliard bajtů. Tento problém lze řešit namapováním určitého intervalu v hodnotách atributu na přesně daný pod interval z intervalu <0,1>. Například u atributu *velikost* můžeme všechny hodnoty od 0 do 999 kB namapovat na interval <0;0,5>, hodnoty v řádu MB mapovat na interval (0,5;0,75> a hodnoty v řádu GB mapovat na interval (0,75;1>. Potom se určí absolutní rozdíl mezi namapovanými hodnotami.

Když už jsou vypočtené rozdílnosti jednotlivých atributů objektů, celkovou rozdílnost můžeme vypočítat podle Rovnice 2.1 [25]:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \tag{2.1}$$

kde p je počet atributů objektů i a j, $d_{ij}^{(f)}$ je rozdíl mezi hodnotami atributu f a $\delta_{ij}^{(f)}$ je váha atributu f.

Vahami atributů je možné zařídit, aby se některé atributy na výsledné rozdílnosti podílely více nebo naopak méně. V této práci se používají váhy v rozsahu 0-1, ale obecně jde jako váhy používat i vyšší hodnoty.

2.3 Shluková analýza

Shluková analýza třídí objekty do skupin (shluků) a to tak, že objekty v jedné skupině jsou si podobnější než objekty z různých skupin. Při použití shlukové analýzy nad NetFlow daty by měl každý shluk obsahovat toky, které jsou si podobné ve vybraných analyzovaných vlastnostech. Například pokud se při analýze NetFlow záznamů zaměříme jen na zdrojové a cílové IP adresy a čísla portů, každý shluk by měl obsahovat jednosměrnou komunikaci mezi dvěma určitými klienty na určitých portech.

2.3.1 DBSCAN (Density-based spatial clustering of applications with noise)

Přednosti metod založených na hustotě obecně je, že dokáží nalézt shluky téměř jakýchkoli tvarů. DBSCAN je metoda založená na hustotě datových bodů v n dimenzionálním prostoru. To znamená, že se snaží najit shluky, které mají v každé své části hustotu datových bodů větší než nějaký limit. Počet dimenzi n je dán počtem analyzovaných atributů objektů. Objekty jsou v prostory zobrazeny jako datové body.

Algoritmu DBSCANu je potřeba zadat dva parametry. Jedním parametrem je ϵ, který udává poloměr okolí každého bodu. Okolí bodu b je oblast s poloměrem ϵ a středem v bodě b. Hustotou okolí bodu b rozumíme počet bodů v této oblasti (včetně bodu b). Dalším parametrem je *minPts*, který stanovuje minimální hustotu, na základě které se rozhoduje, jestli je nějaký bod středovým bodem či nikoli. Jako středový bod je rozpoznán bod, jehož hustota okolí je větší nebo rovna parametru *minPts*.

Algoritmus hledání shluků DBSCAN pracuje následujícím způsobem. Na začátku se všechny body ze vstupní množiny datových bodů D označí jako nenavštívené. DBSCAN

pak náhodně vybere jeden nenavštívený bod b, označí ho za navštívený a zkontroluje, jestli je hustota okolí bodu b větší nebo rovna *minPts*. Pokud není označí bod b jako šum. Pokud je hustota jeho okolí větší nebo rovna *minPts*, vytvoří shluk A, přidá bod b do shluku A a všechny body z okolí bodu b přidá do množiny kandidátních bodů N. Potom DBSCAN postupně vybírá body z množiny N, které nepaří do žádného shluku a přidává je do shluku A. V každé iteraci výběru bodu z množiny N vybere vždy jeden bod b', označí ho za navštívený, pokud není přiřazen jinému shluku, přidá ho do shluku A, a opět zkontroluje, jestli je hustota jeho okolí větší nebo rovna *minPts*. Pokud ano, přidá body z okolí bodu b' do množiny N a proces opakuje výběrem dalšího bodu z množiny N. Když už v množině N nejsou žádné body je shluk A kompletní a DBSCAN náhodně vybere další ještě nenavštívený bod z množiny D a celý postup opakuje. Níže je algoritmus DBSCAN zapsaný v pseudokódu, viz 1.

Algoritmus 1: DBSCAN [25]

```
označ všechny body jako nenavštívené;
repeat
    náhodně vyber nenavštívený bod b;
    označ bod b jako navštívený;
    if v okolí bodu b je alespoň minPts bodů then
        vytvoř nový shluk C a přidej do něj bod b;
        body z okolí bodu b přidej do množiny N;
        for all b' ∈ N do
            if bod b' je nenavštívený then
                označ bod b' jako navštívený;
                pokud je v okolí bodu b' alespoň minPts bodů,
                přidej tyto body do množiny N;
            end if
            pokud bod b' ještě nenáleží žádnému shluku přidej ho do shluku C;
        end for
        shluk C je kompletní;
    else
        označ bod b jako šum;
    end if
until nezbývají žádné ještě nenavštívené body;
```

Prakticky je DBSCANu ještě potřeba zadat vzdálenostní funkci. Pomocí této funkce se bude v algoritmu počítat vzdálenost mezi jednotlivými body. Vzdálenostní funkce musí výslednou vzdálenost dvou datových bodů vypočítat z jejich atributů. Musí nějakým způsobem spočítat rozdíly mezi všemi atributy jednoho bodu a odpovídajícími atributy druhého bodu. Pak musí tyto dílčí rozdíly zkombinovat do výsledné vzdálenosti mezi body. Pokud jsou všechny atributy porovnávaných bodů numerického typu můžeme body umístit do euklidovského prostoru a vzdálenost mezi nimi počítat jako euklidovskou vzdálenost podle vzorce 2.2 [25].

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2} \tag{2.2}$$

kde x_{i1} je hodnota prvního atributu datového bodu i, $(x_{i1} - x_{j1})$ je rozdíl mezi odpovídajícími atributy datových bodů i,j a $d(i, j)$ je výsledná euklidovská vzdálenost bodů i a j.

Datové body mají často i nenumerické atributy. Takové body už nelze zobrazit v euklidovském prostoru. Mějme například atribut kategorického typu, který může nabývat šesti různých hodnot a tyto hodnoty spadají do třech různých kategorii (dvě hodnoty do každé kategorie). Když budeme požadovat, aby vzdálenost (rozdíl) mezi hodnotami ze stejné kategorie byla nenulová a zároveň aby vzdálenost mezi jakýmikoli dvěma hodnotami z různých kategorií byla stejná, není možné toto zařídit v euklidovském prostoru bez přidání dalších dimenzí. Datové body se tedy mapují do n dimenzionálního prostoru neeuklidovského typu. Proto už nelze vzdálenost mezi body počítat jako euklidovskou vzdálenost. Místo toho je vhodné pro výpočet vzdálenosti použít 2.1.

kde p je počet atributů objektů i a j, $d_{ij}^{(f)}$ je rozdíl mezi hodnotami atributu f a $\delta_{ij}^{(f)}$ nabývá hodnotu 1 pokud je možné určit rozdíl mezi atributy f. V případě, že hodnota atributu f u jednoho (případně obou) objektů chybí, nebo jsou hodnoty neporovnatelné, je $\delta_{ij}^{(f)}$ rovna 0.

Složitost je ovlivněna především vyhledáváním bodů v okolí zpracovávaného bodu. Hledání vzdáleností mezi body je možné urychlit použitím R* stromů a dosáhnout tak časové složitosti $\mathcal{O}(n \log n)$ [23]. Bez použití R* stromů nebo jiné pomocné struktury, je časová složitost algoritmu v nejhorším případě $\mathcal{O}(n^2)$.

Kapitola 3

Monitorování sítě

Monitorování sítě spočívá ve zjišťování stavu síťových zařízení a služeb. Monitorování sítě je důležité pro síťového správce. Využívá se k detekci napadených počítačů, které vykazují podezřelé chování a k detekci úniku dat. Dále se používá k profilování zařízení. Profilování spočívá v tom, že správci poskytuje informace o komunikaci daného zařízení v čase [15]. Zařízení při svém běžném provozu používá například pouze určitá čísla portů, komunikuje jen v určitých časových intervalech apod. Profilování neslouží jen k detekci nebezpečného nebo podezřelého chování, ale také pochopení topologie sítě a její vytížení. Díky tomu lze pak provést kroky k lepšímu rozložení zátěže na síti.

3.1 Pasivní monitorování

Při pasivním monitorování sítě se sbírají především logovací a stavové informace ze síťových zařízení např. poštovního serveru nebo FTP serveru, nebo se zachytávají informace o paketech procházejících sítí. Sbírání informací může probíhat pomocí protokolu *syslog* nebo asynchronními SNMP zprávami typu *trap*, zasíláním NetFlow dat aj.[30]. Pokud se síťový administrátor rozhodne zachytávat provoz na sítí, je možné jej přímo analyzovat v reálném čase, nebo pokud to z technických důvodů není možné, lze analýzu provést později.

3.2 Aktivní monitorování

U aktivního monitorování se síťový administrátor aktivně dotazuje na stav síťových prvků nebo pravidelně testuje dostupnost síťových služeb. Pravidelné aktivní testování bývá automatizované a jen v případě nedostupnosti služby dojde k upozornění síťového administrátora. Dotazování síťových prvků probíhá pomocí zpráv ICMP, protokolu SNMP aj. [30].

3.3 Simple Network Management Protocol (SNMP)

SNMP je protokol, který funguje na principu zasílání zpráv různým zařízením na síti. Lze se dotazovat na stav zařízení, nebo zařízení i konfigurovat. Pomocí SNMP lze například změnit konfiguraci směrovače nebo zjistit stav tiskárny. Kdyby se síťový administrátor nebo SNMP agent neustále dotazoval na stav různých zařízení, značně by tím zatěžoval linku SNMP provozem. Proto je možné v zařízení nastavit, aby při dosažení určité podmínky samo odeslalo tzv. *trap* zprávu poslouchajícímu agentovi [27]. Pro doručování zpráv se používá nespolehlivý protokol UDP. Pokud se agent dotáže zařízení na jeho stav a zpráva se po cestě

ztratí, může se agent dotázat znovu. Problém nastává, pokud dojde ke ztrátě *trap* zprávy od zařízení k agentovi, přijetí zprávy se totiž agentem nijak nepotvrzuje. Výhodou SNMP je standardizace, díky které je podporován velkým množstvím zařízení.

Nevýhodou SNMP je, že administrátor může zjistit např. informace o počtu přenesených paketů na síťové kartě nebo směrovači, ale už nezjistí, kdo komunikoval s kým, o jaký typ komunikace šlo apod.

3.4 Deep Packet Inspection (DPI)

Metoda DPI oproti zachytávání NetFlow záznamů, nezpracovává jen hlavičku IP datagramu, jejíž obsah je nezbytný pro správné směrování dat. DPI může pracovat s daty na vrstvách 2–7 modelu ISO/OSI. Pokud není komunikace šifrovaná dokáže analyzovat i obsah zprávy na aplikační vrstvě. DPI se používá zejména těmito způsoby:

- Ve zprávách aplikační vrstvy hledá určité skupiny znaků, nebo testuje data na shodu s regulárními výrazy. Tímto způsobem je možné DPI použít i pro cenzurování, nebo uzpůsobení směrování datagramů podle druhu dat – textová data, video atd. [29].
- Pokouší se najít podobnosti mezi kontrolovaným paketem a pakety používaných při známých bezpečnostních útocích [29].
- Někteří poskytovatelé internetu používají statistická data nasbíraná pomocí DPI pro lepší cílení reklamy, zjištění návyků uživatelů a kontrolu vytíženosti linek

Tato analýza každého paketu je výpočetně velmi náročná, proto se používá vzorkování, nebo se použije úplně jiná metoda monitorování síťového provozu. DPI je vhodné použít, pokud potřebujeme blíže určit chybu na síti, nebo zkontrolovat jaká konkrétní data odesílají síťové prvky. Oblíbeným nástrojem pro analýzu jednotlivých paketů je Wireshark [11].

3.5 Technologie NetFlow

Netflow je protokol vyvinutý společností Cisco Systems. Používá se ke sledování datových IP toků na síti. Zachytávání metadat na síti spočívá v zaznamenávání metadat o každém paketu, který projde určitou částí sítě. Díky získaným informacím lze analyzovat a vhodně zobrazit datový provoz náležící jednotlivým zařízením. Nasbíraná data je možné agregovat a sledovat statistiky provozu na celé síti, nebo její části. Dnes se často využívá ke sledování toho, jaké síťové aplikace jsou jednotlivými zařízeními používány v daném čase. Netflow je hodně využíváno poskytovateli internetu, kteří potřebují účtovat poplatky za připojení a také mít přehled o dění na síti a vytížení sítě. NetFlow je mocným nástrojem při odhalování síťových útoků jako je DoS, falešný DHCP server, podvržené DNS odpovědi aj.

Jednou z výhod NetFlow je, že nasbírané datové toky obsahují pouze metadata o síťovém provozu, nikoli samotná přenesená data. Objem záznamů o tocích tak je jen zlomkem objemu přenesených dat. Protože záznamy o tocích obsahují zdrojové a cílové IP adresy, porty atd. nedá se říci, že jsou anonymní. Pro případného útočníka mohou být záznamy z delšího časového horizontu zajímavé. Může tak např. zjistit jaké síťové prvky v sítí existují, jaká je topologie sítě nebo jaké datové toky jsou právě aktivní.

První verzí mechanismu NetFlow byla verze 1. Dále existují tyto verze [5]:

- Verze 5. Od této verze obsahují hlavičky paketů při exportu toků sekvenční čísla pro možnost detekce ztráty dat. Součástí toků jsou nově čísla autonomních systémů získaná z protokolu BGP.
- Verze 7. Není kompatibilní s Cisco směrovači. Používá se jen v přepínačích série Cisco Catalyst 5000 vybavených NetFlow feature card (NFFC).
- Verze 8. Přidává agregaci nasbíraných dat přímo ve směrovači. Díky tomu je objem dat exportovaných na kolektor významně menší.
- Verze 9. Zavádí export toků založený na šablonách, které definují, jaká data exportované toky obsahují.
- IPFIX někdy také označována jako NetFlow v10. Mimo jiné umožnuje jako součást toků exportovat informace, které se běžně posílají pomocí zpráv syslog a také informace, běžně sbírané pomocí SNMP. Další významnou novinkou je podpora proměnné délky polí v paketech, takže je možné exportovat např. URL adresu [16].

3.6 Architektura NetFlow

Tok je jednosměrná posloupnost paketů procházejícími sledovacím síťovým zařízením s určitými stejnými údaji [14]. Unikátní n-tice údajů, která tok jednoznačně odlišuje od ostatních toků může být složena z následujících údajů [27]:

- Zdrojová IP adresa
- Cílová IP adresa
- Zdrojový port
- Cílový port
- Číslo IP protokolu
- Typ služby
- Vstupní rozhraní

Pro odlišení toků se nemusí použít všech sedm uvedených vlastností, stačí jen výběr z nich, záleží na implementaci. Toky samozřejmě mohou obsahovat mnoho dalších údajů, jako počet přenesených bytů, počet přenesených paketů, čísla autonomních systémů apod. Tyto údaje však mohou být pro několik toků totožné.

Zařízení, které je v síti připojené a zaznamenává metadata o paketech, které přes něj procházejí se nazývá exportér. Exportérů bývá po síti rozmístěných více. Exportéry si ve své vnitřní paměti tvoří záznamy o datových IP tocích a nasbíraná data odesílají do kolektoru.

K exportu toku do kolektoru dochází v následujících případech [27]:

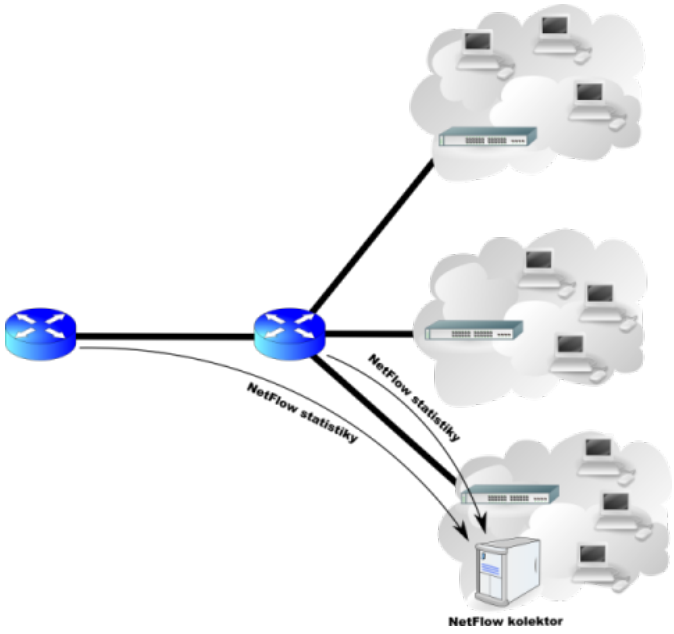
- Neaktivní timeout – Poslední zachycení paketu náležícímu k danému toku došlo před dobou danou neaktivním timeoutem.
- Aktivní timeout – Tok je stále aktivní, ale došlo k vypršení doby dané aktivním timeoutem.

- Detekce konce toku – Při TCP spojení lze detekovat konec toku přijetím paketu s příznakem RST nebo FIN.
- Zaplnění paměti exportéru

Kolektor je typicky server, uchovávající všechny NetFlow záznamy. Obvykle je součástí kolektoru i sada aplikací pro zpracování a následnou vizualizaci zachycených záznamů.

Mezi nejpoužívanější protokoly pro přenos dat mezi exportérem a kolektorem patří NetFlow v5 a NetFlow v9. Data jsou přenášena protokolem UDP, takže když dojde k jejich ztrátě nelze je získat zpět. Detekovat, že ke ztrátě vůbec došlo je možné od NetFlow verze 5, díky sekvenčním číslům. Přenos není žádným způsobem šifrovaný. Z daného exportéru lze data odesílat pouze na jeden kolektor. Tento kolektor může data distribuovat dále, ale ztratí se tím informace o originálním zdroji nasbíraných dat, tu může původní kolektor přenést jiným způsobem, nebo adresu odesílatele zfalšovat a vydávat se za původní zdroj exportovaných dat [27].

Jako NetFlow exportéry je možné využít směrovače, které exportování NetFlow záznamů podporují. Takovéto směrovače jsou nicméně poměrně drahé a kromě směrování, musejí provádět ještě výpočet NetFlow statistik, což má negativní dopad na jejich výkon. Snížení nároků na výkon exportéru se dosahuje použitím vzorkování, kdy se zaznamenává jen jeden náhodný paket vybraný z okna o velikosti n paketů [13]. Nevýhodou vzorkování je však snížení přesnosti záznamů.

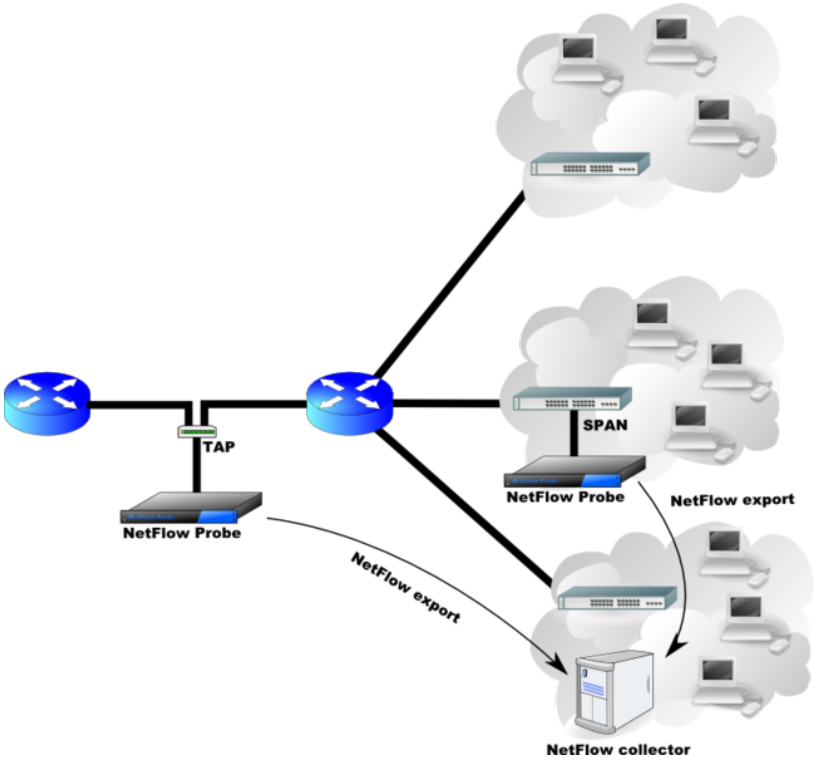


Obrázek 3.1: Architektura NetFlow [3]

3.7 Specializované NetFlow sondy

Protože nároky na NetFlow exportéry stále rostou používají se pasivní sondy, které na rozdíl od přepínačů nebo routerů pouze ukládají NetFlow záznamy. Odeslání zachycených dat do

kolektoru může probíhat přes dedikovanou linku. Díky dedikované lince přenos NetFlow záznamů nijak neovlivňuje ostatní datový provoz na síti a je odolnější vůči útokům.



Obrázek 3.2: Architektura NetFlow s pasivními sondami [9]

3.8 Softwarové sondy

Softwarové sondy běží na standardních serverech. Mohou být specializované na zachytávání NetFlow záznamů pouze určitého typu komunikace např. HTTP, P2P, voice-over-IP atd. Jejich výhodou je především cena. Nad nasbíranými daty mohou sondy ještě před odesláním provádět určitou agregaci. Jedním z open-source řešení je nProbe vyvíjené společností ntop [8].

3.9 NetFlow záznam

Záznam protokolu NetFlow v9 obsahuje minimálně tyto informace:

- Zdrojová IP adresa
- Cílová IP adresa
- Zdrojový aplikační TCP/UDP port
- Cílový aplikační TCP/UDP port
- IP adresa příštího hopu

- Číslo vstupního fyzického rozhraní
- Číslo výstupního fyzického rozhraní
- Počet paketů v toku
- Počet bytů v toku
- Čas začátku toku
- Čas konce toku
- Číslo IP protokolu
- Type of service (ToS)
- Sjednocené TCP příznaky
- Číslo zdrojového autonomního systému
- Číslo cílového autonomního systému
- Maska zdrojové podsítě
- Maska cílové podsítě
- Další příznaky

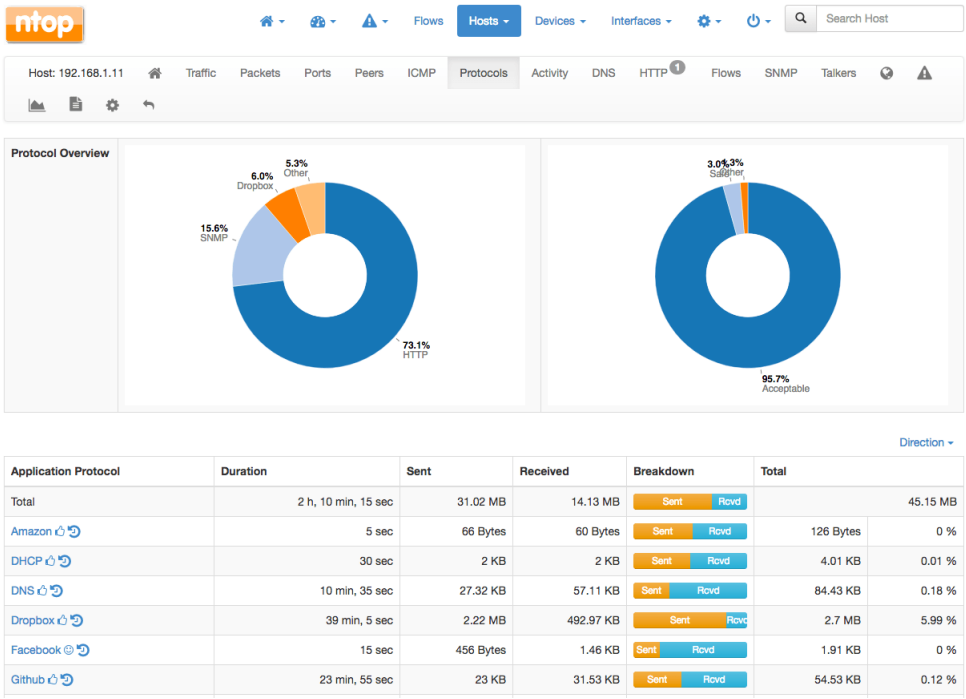
3.10 ntopng

Nástroj ntopng funguje jako kolektor záznamů a zároveň i jako analyzátor. Vytížení sítě, jednotlivé toky, grafy a statistiky zobrazuje v reálnem čase pomocí několika dashboardů ve webovém rozhraní. Při použití softwarové síťové sondy nProbe^{TM}, kterou také vyvíjí společnost ntop, lze v ntopng měnit nastavení sondy.

Mezi nejvýznamnější analyzační a monitorovací funkce patří Top N statistika, řazení toků podle zadaných kritérií, statistiky týkající se protokolu TCP (příznaky, přenesené byty, pakety), statistika používaných transportních protokolů, monitorování SNMP zařízení a statistika aplikačních protokolů. Pro odhalení podezřelého nebo nestandardního chování slouží různá upozornění, která mohou být generována, když dojde k překročení definovaného limitu přenesených paketů nebo bytů nebo když nějaký klient odesílá velké množství TCP paketů s příznakem SYN (může se jednat o SYN flood útok) apod. Analýzu a zkoumání nasbíraných dat je možné provádět i později.

3.11 NfSen

NfSen [7] je webový front-end pro nástroj nfdump, který slouží ke zpracování NetFlow záznamů. NfSen tedy umí NetFlow data zpracovávat stejně jako nfdump, ale je možné program ovládat v grafickém webovém rozhraní a síťová data jsou vykreslena do grafů. NfSen umožňuje zpracovávat data pouze ze souboru. Pokud chceme monitorovat dění na síti neustále, můžeme nastavit, aby NfSen periodicky načítal nově zachycené toky a dále s nimi pracoval. Při odhalování potenciálně nebezpečného chování velmi pomůže, možnost nastavit různá upozornění, podobně jako u nástroje ntopng. Při agregaci a filtrování NetFlow



Obrázek 3.3: Architektura NetFlow s pasivními sondami [4]

záznamů můžeme zadat časové období, se kterým chceme pracovat. Kromě jednoduchého filtrování pomocí zaškrtávacích políček se filtrování provádí stejně jako u nástroje nfdump.

Příklad filtru: `tcp and (src ip 177.17.18.19 or dst ip 178.19.20.21)`

Když síťový administrátor uvidí v grafu prudký nárůst přenášených dat může to znamenat DOS útok. Jestli jde opravdu o útok, odkud pochází apod. pak v ideálním případě zjistí dalším filtrováním záznamů. Podobně (i když pravděpodobně s větším úsilím) může odhalit další typy útoků na síť jako např. skenování portů, falešný DHCP server a další.

Kapitola 4

Sledované vlastnosti datových toků

Tato kapitola se zabývá popisem některých důležitých vlastností datových toků. Každá vlastnost o toku vypovídá něco jiného. Při hledání anomálií nebo podobností se vždy používá kombinace několika vlastností toků. Kombinace více vlastností má logicky větší vypovídací hodnotu o charakteru síťového provozu. Při detekci různých druhů útoků se sledují různé kombinace vlastností, přičemž některé sledované vlastnosti mohou mít větší váhu než jiné. To, jaké vlastnosti si pro analýzu vybereme a jakou jim přiřadíme váhu, záleží jen na tom, co se z analyzovaných dat snažíme zjistit.

4.1 Klasifikace IP adres

Internet je velmi rozsáhlý a proměnlivý, takže není možné, aby měl každý směrovač informace o všech dalších směrovačích nebo sítích. Z tohoto důvodu se zavedlo třídní adresování, které bylo později nahrazeno beztřídním adresováním (CIDR). Přestože se už třídní adresování prakticky nepoužívá stále existují rozsahy adres vyhrazeny pro multicast, experimentální účely a další specifické použití. Pokud při zkoumání podobností mezi datovými toky narazíme na IP adresy, o kterých podle prefixu víme, že se používají na něco jiného, nebudou spolu pravděpodobně zkoumané toky souviset.

Pro pochopení, jak se IP adresy dělí do tříd, je potřeba vysvětlit pojmy *netid* a *hostid*. Všechny adresy IPv4 mají délku 32 bitů. Část adresy, která určuje síť se označuje jako *netid*. Pokud bude část *netid* dlouhá 8 bitů, mohlo by teoreticky existovat $2^8 = 256$ takovýchto různých sítí. U třídního adresování však například adresy třídy A začínají na 0. Přestože mají adresy třídy A *netid* dlouhé 8 bitů, první bit musí být nulový, takže různých sítí třídy A může existovat jen 2^7-2 = 126. Zbylých 24 bitů adresy se použije na adresování jednotlivých počítačů v dané síti. Tato část adresy se označuje jako *hostid*. Délku *netid* (a nepřímo tedy i *hostid*) vyjadřuje maska sítě. Typicky se používá zápis *IP adresa/prefix sítě*. Kde prefix sítě vyjadřuje počet jedničkových bitů v masce — počet bitů použitých pro *netid* [30].

4.1.1 Třídní adresování

Při třídním adresování adresy IPv4 rozdělují do několika tříd — viz Tabulka 4.1.

Existují jisté adresy a rozsahy adres, které jsou vyhrazeny pro speciální účely. Třeba adresy pro adresování v privátních sítích — viz Tabulka 4.2.

Další vyhrazené adresy podle dokumentů RFC jsou uvedeny v Tabulce 4.3.

Adresy z tříd A–C se označují jako unicastové adresy a jsou používány k běžnému adresování zařízení na internetu. Adresy třídy D jsou používány pro adresování multicastových

Třída	Začátek adresy	Prefix sítě	Množství použitelných adres
A	1–126	/8=255.0.0.0	167 77 214 (2^{24}-2)
B	128–191	/16=255.255.0.0	65 534 (2^{16}-2)
C	192–223	/24=255.255.255.0	254 (2^{8}-2)
D	224–239	**Multicast**	
E	240–254	**Rezervováno pro experimentální účely**	

Tabulka 4.1: Rozdělení adres IPv4 do tříd [30]

Počáteční adresa	Koncová adresa	Maska
10.0.0.0	10.255.255.255	/8=255.0.0.0
172.16.0.0	172.31.255.255	/12=255.240.0.0
192.168.0.0	192.168.255.255	/16=255.255.0.0

Tabulka 4.2: Rozsahy IP adres pro adresování v privátních sítích [30]

skupin. Dále je možné v Tabulce 4.3 identifikovat adresy a rozsahy adres vyjmuté ze tříd A–E. I tyto adresy se běžně používají např. Link Local adresy.

Příslušnost IP adresy do určité skupiny lze použít při zkoumání podobnosti mezi datovými toky na základě IP adres. Pokud jsou zdrojové nebo cílové adresy dvou toků z jiných skupin naznačuje to, že jejich podobnost je velmi malá. Pokud jsou IP adresy datových toků ze stejné skupiny neznamená to, že by mezi toky byla automaticky vysoká podobnost. Skupiny IP adres jsou rozsáhlé a většina běžné komunikace probíhá mezi unicastovými adresami, tj. ze skupin A–C. Pokud chceme dále zkoumat podobnost mezi toky, musíme se podívat na jejich další vlastnosti. V případě, že jsou IP adresy toků totožné, je zřejmé, že je jejich podobnost vysoká.

4.1.2 Beztřídní adresování (CIDR)

Beztřídní adresování, angl. CIDR (Classless Inter-Domain Routing) bylo zavedeno, aby se prostor IP adres využíval efektivněji než u třídního adresování. Oproti třídnímu adresování není prefix sítě, a tedy ani maska, pevně dána. Délku masky sítě lze zvětšovat nebo zmenšovat a tím vytvořit síť požadované velikosti. Beztřídní adresování se používá především u IP adres třídy C, nicméně lze využít i u IP adres z tříd A a B. Sítě z třídy C mohou adresovat pouze 254 počítačů. Je to proto, že mají masku /24 a na adresování počítačů zbývá už jen 8 bitů. U beztřídního adresování můžeme masku zmenšit jen na např. 22 bitů a tím vytvořit síť, na které lze adresovat 1022 počítačů. Tímto způsobem vlastně dojde ke spojení několika dříve oddělených (majících různé *netid*) sítí z třídy C a vytvoření tzv. *supersítě*. Je tedy možné vytvářet sítě různých velikostí podle potřeby [30].

U techniky CIDR je směrování složitější než u třídního adresování. Protože masku sítě nelze určit z jejího prefixu, jak tomu je u třídního adresování. Směrovače tak musí uchovávat i masku každé sítě. Aby bylo adresování efektivnější používají se pro směrování čísla Autonomních systémů a protokol BGP, viz kapitola 4.5 Autonomní systémy.

Blok adres	Použití	Označení dokumentu
0.0.0.0/8	„This host on this network“	RFC 1122
100.64.0.0/10	Shared Address Space	RFC 6598
127.0.0.0/8	Loopback	RFC 1122
169.254.0.0/16	Link Local	RFC 3927
192.0.0.0/24	IETF Protocol Assigments	RFC 6890
192.0.0.0/29	DS-Lite	RFC 6333
192.0.2.0/24	TEST-NET–1	RFC 5737
192.88.99.0/24	6to4 Relay Anycast	RFC 3068
198.18.0.0/15	Benchmarking	RFC 2544
198.51.100.0/24	TEST-NET–2	RFC 5737
203.0.113.0/24	TEST-NET–3	RFC 5737
240.0.0.0/4	Reserved	RFC 1112
255.255.255.255/32	Limited Broadcast	RFC 0919

Tabulka 4.3: Další vyhrazené adresy [17]

4.2 Protokol UDP (User Datagram Protocol)

UDP je transportní bezstavový protokol. Funguje jednoduchým způsobem, a to tak, že převezme doručovanou zprávu od aplikační vrstvy L7, zabalí ji do UDP datagramu, přidá hlavičkou a datagram předá síťové vrstvě — IP protokolu. IP protokol datagramu přidá IP hlavičku a odešle zprávu po síti. Tento způsob doručování dat se označuje jako „best-effort delivery“. Protokol UDP nezaručuje doručení přenášených dat, ani kontrolu úspěšného doručení. Je používán aplikacemi, které mají své vlastní kontrolní mechanismy (např. počítadla), kterými jsou schopné zjistit, že došlo ke ztrátě dat na síti. Hlavičky UDP datagramů mohou volitelně obsahovat kontrolní součet, takže není problém zjistit, jestli konkrétní datagram došel k příjemci nepoškozen. Díky své jednoduchosti je UDP v některých případech až o 40% rychlejší v porovnání s protokolem TCP [21].

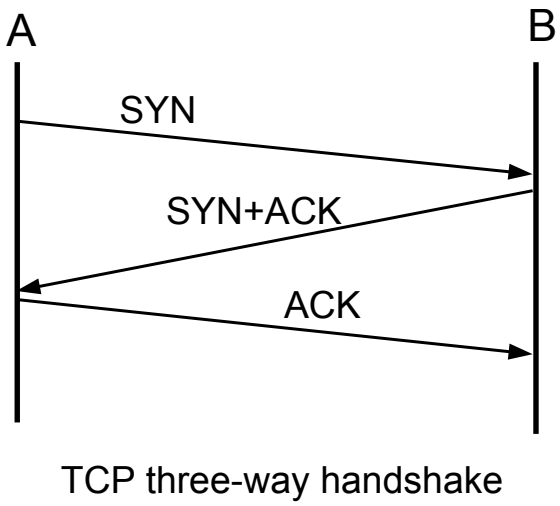
4.3 Protokol TCP (Transmission Control Protocol)

TCP je transportní protokol modelu TCP/IP. Protokol TCP používají aplikace, které vyžadují úspěšné doručení dat k příjemci. Protokol vytváří logické spojení mezi dvěma komunikujícími stranami na síti. Dokud je spojení otevřené komunikující strany si vyměňují nejen data, ale i potvrzovací zprávy o přijetí paketů s určitým sekvenčním číslem. Pokud se nepodaří některý paket doručit — sekvenční čísla přijatých paketů nejdou za sebou — provede se pokus o opětovné doručení ztraceného paketu. Protokol také používá časovač, kterým kontroluje, jestli netrvá odpověď na zprávu příliš dlouho a spojení není přerušené [21].

Před prvním navázáním spojení protokolem TCP se provádí tzv. three-way handshake. Princip činnosti mechanismu je následující [21]:

- Ověří, že je příjemce dosažitelný
- Ověří, že příjemce naslouchá na daném portu
- Odesílatel a příjemce si vymění počáteční sekvenční čísla paketů, od kterých budou dále pakety číslovat

Jaké TCP příznaky jsou nastaveny paketům při three-way handshaku je vidět na obrázku 4.1.



Obrázek 4.1: Three-way handshake [10]

V hlavičce TCP paketu se používá 6 bitů pro příznaky (TCP flags). V hlavičce je rezervováno více místa, aby šlo další příznaky přidat později. Příznaky slouží k indikaci aktuálního stavu TCP spojení, nebo nesou další informaci o přenášených datech. Význam příznaků viz Tabulka 4.4.

Příznak	Význam
URG (Urgent)	Indikuje, že pole „Urgent Pointer“ v hlavičce paketu obsahuje ukazatel na místo v datové části paketu. Od tohoto místa by příjemce měl začít číst data.
ACK (Acknowledgment)	Vztahuje se k poli „Acknowledgment Number“, které příjemci říká, jaké další pořadové číslo paketu (Sequence number) má očekávat.
PSH (Push)	Na straně odesílatele i příjemce jsou vyrovnávací paměti, kde pakety čekají na zpracování. Při nastavení příznaku PSH příjemce nesmí ukládat data do vyrovnávací paměti, ale rovnou je předávat aplikační vrstvě. Používá se při ukončení spojení, kdy by přijatá data měla být kompletní. Celý obsah vyrovnávací paměti se předá aplikaci k dalšímu zpracování.
RST (Reset)	Úplně uzavírá spojení. Používá se také pro odmítnutí spojení.
SYN (Synchronize)	Používá se pro synchronizaci pořadových čísel (Sequence number) v handshake procesu při ustavování spojení.
FIN (Finish)	Indikuje, že přenos dat byl dokončen. Tento příznak explicitně neuzavírá spojení, ale pokud obě komunikující strany odešlou paket s tímto příznakem a navzájem si odpoví správným ACK paketem spojení je ukončeno.

Tabulka 4.4: Příznaky TCP paketu [21]

Příznaky protokolu TCP hrají roli při odhalování skenování. Když se útočník snaží připojovat na různé IP adresy a různé porty, většinou používá stále stejně nastavené TCP příznaky. Tato skutečnost pomůže při jednoznačné identifikaci skenování.

4.4 Klasifikace čísel portů transportních protokolů

Pro rozlišení jednotlivých služeb používajících transportní protokol se používají různá čísla portů. Čísla portů jsou 16bitová čísla. Porty se podle číselného rozsahu dělí do třech skupin [20]:

- Známé porty (well known ports) – čísla portů v rozsahu 0–1023, jsou vyhrazené pro nejběžnější služby na internetu
- Registrované porty – v rozsahu 1024–49151, registrují se u mezinárodní organizace IANA
- Dynamické a soukromé porty – v rozsahu 49152–65535, používají se pro dynamické přidělování aplikacím komunikujícím na internetu, nejsou pevně vázány na určitou aplikaci nebo službu

Známé porty jsou přidělovány na základě procesu „IETF Review“ nebo „IESG Approval“ mezinárodní skupinou IETF. Registrované porty jsou přidělovány organizací IANA opět na základě procesu „IETF Review“, „IESG Approval“ nebo „Expert Review“. Procesy registrace portů jsou popsány v dokumentu RFC 8126 [19]. Dynamické porty nejsou nijak pevně přidělovány nebo registrovány [20]. Bez předchozí registrace u organizace IANA, by se známé i registrované porty neměly používat. Přesto se lze na internetu občas setkat s jiným, než registrovaným použitím známých portů. Většinou se pro určitou službu používají stejná čísla portů u protokolů TCP i UDP. Jen ve vzácných případech nepatří stejné číslo portu u protokolů TCP a UDP ke stejné aplikaci nebo službě, např. port 520 [21].

Čísla portů mají při zkoumání podobností mezi datovými toky velký význam. U portů z rozsahu 0–1023 jsme schopni identifikovat internetovou službu. U registrovaných portů je možné odhadnout komunikující aplikaci, pokud je aplikace s daným číslem portu registrována. V případě, že je zdrojovým portem port z dynamického rozsahu a cílovým portem port ze známého rozsahu, pravděpodobně jde o komunikaci klienta, který přistupuje k nějaké internetové službě. Tímto způsobem pak můžeme např. najít útočníka, který se snaží komunikovat na velkém množství různých portů.

4.5 Autonomní systémy

Protože směrovače nemohou uchovávat ve směrovacích tabulkách informace o všech veřejných sítích na internetu. Byl zaveden hierarchický systém autonomních systémů (dále jen AS). Pod jedním AS je skupina sítí a směrovačů se stejnou směrovací politikou. Autonomním systémem může být např. AS konkrétního poskytovatele internetu. Je rozdíl mezi vnějším směrováním, tzn. mezi autonomními systémy a vnitřním směrováním v rámci jednoho AS. Směrovače na hranicích AS směrují data do dalších AS. K tomuto směrování mezi AS se používá protokol BGP. Hraniční směrovače v pravidelných intervalech šíří dalším hraničním směrovačům jiných AS informace o tom, k jakým sítím se mohou připojit a přes jaké AS budou data posílána. Hraniční směrovače si vyměňují zprávy obsahující prefixy sítí (a délku prefixu) do kterých mohou směrovat data [26].

AS je identifikován 16bitovým nebo 32bitovým číslem. Tato čísla přiděluje regionální internetový registr např. RIPE NCC. Při hledání podobností v síťových datech můžeme podle čísel AS dvou IP adres zjistit, jestli jsou adresy pod správou jednoho poskytovatele internetu, nebo patří jedné organizaci.

Kapitola 5

Vybrané síťové útoky

5.1 Skenování portů

Skenování portů se používá pro zjištění toho, jaké porty jsou otevřené pro navázání spojení. Nejjednodušší forma skenování je, když se útočník snaží připojit postupně na všechny porty na zařízení oběti. Takový způsob skenování je snadno odhalitelný a lze jej zablokovat. Časem vzniklo mnoho nápaditějších technik skenování portů, které jsou mnohem nenápadnější. Útočník většinou používá skenování portů proto, aby se dozvěděl něco o zařízení oběti. Může to být např. verze operačního systému, způsob reagování na nezvyklé pakety nebo jaké síťové služby jsou na zařízení oběti spuštěny a představují potenciální zranitelnost. Informace získané skenováním je pak možné použít pro plánování/provedení dalšího útoku. Útočník typicky na jednotlivá skenovaná zařízení odesílá jen jeden paket. Podle způsobu skenování nastavuje paketům různé TCP příznaky viz dále.

Při horizontálním skenování se útočník pokouší navázat spojení s různými IP adresami, ale vždy na stejném portu. Zjišťuje tím, na které IP adrese běží služba, která je pro daný port typická. Tuto službu se pak pravděpodobně pokusí napadnou. Příklad typického horizontálního skenování vyfiltrovaného pomocí nástroje nfdump[6] je vidět v Tabulce 5.1.

Proto	Src IP Addr:Port	Dst IP Addr:Port	Packets	Bytes	Flags
TCP	252.68.10.88:47858 ->	58.226.107.101:7040	1	40	....S.
TCP	252.68.10.88:47858 ->	58.226.107.79:7040	1	40	....S.
TCP	252.68.10.88:47858 ->	109.155.100.95:7040	1	40	....S.
TCP	252.68.10.88:47858 ->	252.57.153.244:7040	1	40	....S.

Tabulka 5.1: Horizontální sken

Útočníkem je IP adresa `252.68.10.88:47858`, ze které přichází pakety na různé cílové adresy. Skenovaným cílovým portem je port `7040`. Že se s velkou pravděpodobností skutečně jedná o skenování je vidět také z toho, že každý tok je tvořen jen jedním paketem a jediný nastavený TCP příznak je SYN viz. kapitola 5.1.1 SYN Scanning.

Vertikální skenování je obdobné jako horizontální, jen se místo stejného portu na různých IP adresách skenují různé porty na stejné IP adrese. Kombinace horizontálního a vertikálního skenování se označuje jako blokové skenování, při kterém se postupně na každé vybrané IP adrese skenuje určitý rozsah portů.

5.1.1 SYN Scanning

Tento typ skenování je také známý jako „half-open scan“, protože zcela neotevírá TCP připojení. Jak už název napovídá SYN scanning odesílá oběti paket s nastaveným jediným příznakem a tím je SYN. Přesně stejným způsobem probíhá první fáze three-way handshaku při pokusu o navázání TCP spojení, viz kapitola 4.3 Protokol TCP. Pokud oběť odešle jako odpověď paket s příznaky SYN/ACK — pokračování three-way handshake — je skenovaný port jistě otevřený. Útočník pak pošle paket s příznakem RST, čímž připojení ukončí, aby nedošlo k zahlcení oběti. K dokončení three-way handshake tak nedojde [22].

5.1.2 FIN, X-mas, NULL scanning

Jiný způsob skenování portů než SYN scanning, je FIN, X-mas, NULL sncanning. Princip této metody spočívá v odeslání paketu oběti, který je nezvyklý a z pohledu TCP/IP modelu nedává smysl. Pokud je port, na který útočník takový paket odešle otevřený, žádná odpověď zpět nepřijde. Pokud je port uzavřený systém by měl podle dokumentu RFC 793 [12] odeslat odpověď ve formě paketu s nastaveným příznakem RST. Paket odeslaný útočníkem může být trojího typu, podle nastavených příznaků [22]:

- FIN scaning – nastaven je pouze příznak FIN
- X-mas scaning – nastaveny jsou příznaky FIN, URG, PSH (odtud označení X-mas – příznaků je tolik, že připomínají světýlka na vánočním stromečku)
- NULL scanning – není nastaven žádný příznak

5.2 DoS (Denial of Service)

DoS je jeden z nejjednodušších útoků. Jeho cílem je znemožnit oběti navazovat nová spojení a komunikovat na síti. DoS může fungovat tak, že zahltí oběť požadavky, vytíží její síťové pásmo na maximum, takže dojde k znemožnění legitimní komunikace. Jiný typ DoS útoku cílí na konkrétní zranitelnost v zařízení oběti, např. útočník odešle pouze jeden paket, který je sestaven tak, aby došlo k neočekávanému stavu aplikace nebo jádra systému přijímajícího paket. Pokud útočník neposílá pakety oběti pouze ze svého počítače, ale zapojí do útoky i další zařízení, označuje se útok jako DDoS (Distributed Denial of Service). DoS a DDoS útoky je poměrně snadné odhalit v NetFlow datech, nicméně pokud zahlcující pakety přichází z velkého množství zdrojů je těžké, někdy až nemožné, odfiltrovat komunikaci útočníka od té legitimní.

5.2.1 SYN flood

SYN flood útok využívá skutečnosti, že TCP protokol je stavovým protokolem. Zařízení, která komunikují pomocí protokolu TCP si musí uchovávat informaci o stavu všech aktuálních spojení. Scénář útoku je takový, že útočník zaplaví oběť TCP pakety s nastaveným příznakem SYN, jako by chtěl navázat spojení viz. Three-way handshake viz kapitola 4.3 Protokol TCP. Útočník v paketu uvede jako zdrojovou adresu neexistující nebo neplatnou adresu. Když oběť na falešnou zdrojovou adresu odpoví a čeká na paket s příznakem ACK nikdy nedostane odpověď. Připojení se nedokončí a oběť stále vyčkává, dokud

nedojde k vypršení časovače, což však trvá poměrně dlouhou dobu. Útočník zasílá SYN paketů tak velké množství, že dojde k zaplnění tabulky pro polootevřená spojení na straně oběti. Výsledkem je, že oběť nemůže navázat žádná další spojení [22].

Z popisu útoku vyplývá, že útočníka lze z NetFlow záznamů identifikovat podle toho, že odesílá velké množství TCP paketů s nastaveným příznakem SYN.

Kapitola 6

Aplikace pro hledání podobností v síťových datech

Aplikace, která je výsledkem této práce pracuje s datovými toky NetFlow. Porovnává vlastnosti toků a rozřazuje je do shluků. Pro shlukovou analýzu je využit algoritmus DBSCAN. V konfiguračním souboru se nastavují váhy atributů datových toků a parametry pro DBSCAN. Podle použité konfigurace jsou výsledkem zpracování shluky datových toků. Cílem je nalézt shluky obsahující provoz, který je součástí nějakého síťového útoku.

6.1 Návrh aplikace

Aplikace je napsaná v programovacím jazyce C, podle standardu C99. Aplikace načítá datové toky z NetFlow souboru typu *nfcapd*. K přečtení vstupního souboru a extrakci atributů toků se kterými se bude dále pracovat je využita knihovna *libnf* [18]. Vstupní soubor může obsahovat pro každý tok velké množství informací, takže program načte jen vybrané atributy toků do paměti. Ještě před načítáním toků se přečte konfigurační soubor. Konfigurovat lze:

- Váhy atributů a ϵ a *minPts* pro DBSCAN.
- Počet toků, které se mají ze vstupního souboru načíst.
- Jestli se mají toky načítat postupně nebo provést vzorkování napříč celým souborem.
- Které metriky se pro porovnávání toků mají použít, například zdrojové IP adresy, porty a počet přenesených paketů.

Součástí zachycených sítových dat nemusí být čísla autonomních systémů, takže se dohledávají ve volně dostupné databázi GeoLite2 [1] společnosti MaxMind [2]. K prohledávání databáze je použita knihovna vyvíjená také společností MaxMind, která je k tomu určená. Stejným způsobem se dohledávají i přibližné souřadnice zdrojových/cílových IP adres. Výstupem aplikace je sada textových souborů pro každý nalezený shuk, ve kterých jsou statistické informace o tom, jaké hodnoty atributů se ve shluku vyskytují. Každý atribut je vypsán do vlastního souboru. Takže například shluk 1 může mít soubory *src_addr* a *src_port*, ve kterých jsou různé zdrojové IP adresy a zdrojové porty a počet kolikrát se daná hodnota ve shluku vyskytuje. Tyto soubory slouží pro tvorbu histogramů a případně další automatické zpracování, ale je velmi nepřehledné z nich vyčíst charakter shluku, neboli

čím se shluk vlastně odlišuje od ostatních, co je pro něj typické. K tomu slouží výstupní soubor *db_clusters.out*, ve kterém jsou vypsané informace o prvních 20 tocích z každého shluku. Toky jsou vypsané v podobném formátu jako při použití nástroje *nfdump*, příklad je v Tabulce 6.1.

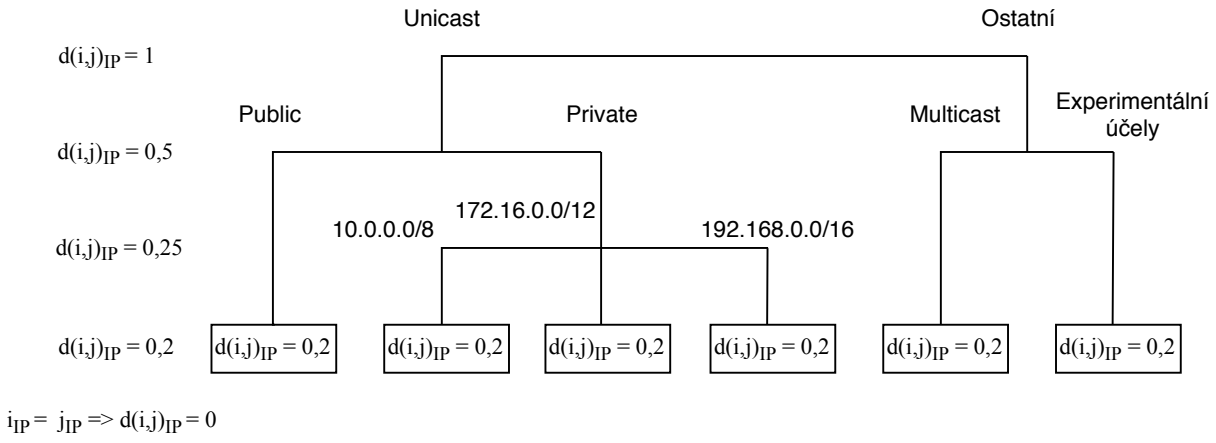
cluster_id	src_addr:port	dst_addr:port	prot	bytes	pkts	src_country	dst_country	src_as	dst_as	tcp_flags
0	82.158.8.249:443	57.189.132.243:25968	6	1089	6	ES	BE	6739	2647	.AP.SF
0	82.158.8.249:443	57.189.132.243:20101	6	660	6	ES	BE	6739	2647	.AP.SF
0	82.158.8.249:443	57.189.132.243:26233	6	1192	6	ES	BE	6739	2647	.AP.SF
0	82.158.8.249:443	57.189.132.243:16955	6	1341	6	ES	BE	6739	2647	.AP.SF

Tabulka 6.1: Příklad souboru s výpisem záznamů ve shlucích

6.2 Porovnání NetFlow záznamů

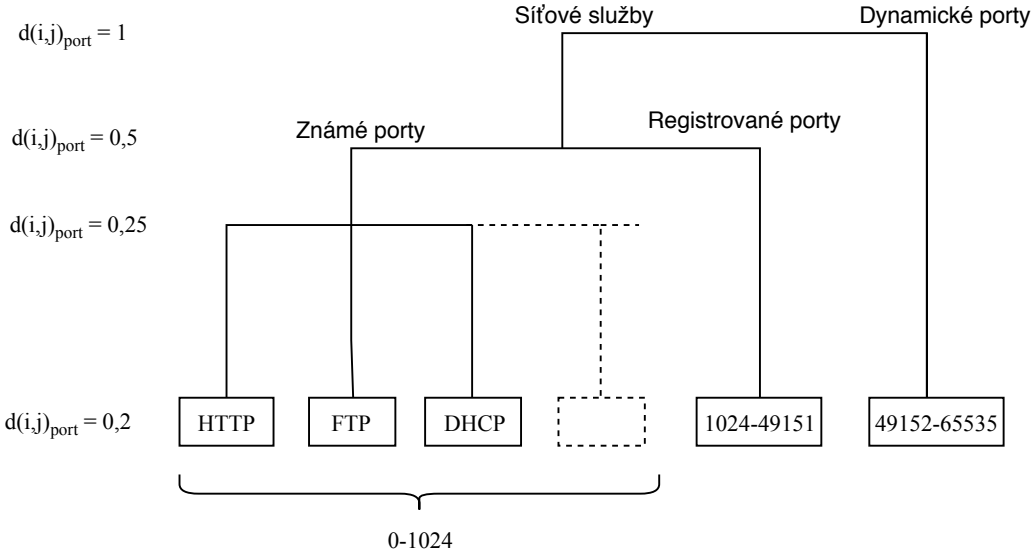
Vzdálenostní funkce použitá v algoritmu DBSCAN používá pro určení vzdálenosti mezi dvěma NetFlow záznamy různé metriky pro různé atributy. Jaké atributy budou zahrnuty do výpočtu vzdálenosti je možné nastavit v konfiguračním souboru. Celkem je možné pracovat s následujícími atributy: zdrojová/cílová IP adresa, zdrojový/cílový port, transportní protokol, počet přenesených bytů a paketů, přibližné souřadnice zdrojové/cílové IP adresy, číslo zdrojového/cílového AS, TCP příznaky. Níže je vysvětleno, jak se jednotlivé atributy porovnávají.

- IP adresy – Adresy se porovnávají na základě příslušnosti do jedné z kategirií, viz Diagram 6.1. Pokud jsou adresy totožné je výsledná vzdálenost rovna 0.



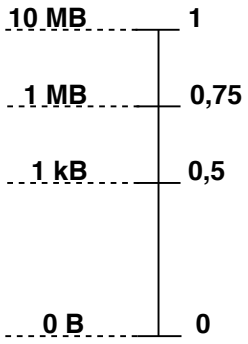
Obrázek 6.1: Vzdálenostní funkce pro IP adresy - Podle toho na které úrovni se adresy liší je výsledkem různá vzdálenost $d(i,j)_{IP}$

- Čísla portů – Porty jsou rozděleny jak je znázorněno na Diagramu 6.2. Přičemž známé porty jsou rozděleny do 17 tříd podle aplikací a služeb, které je běžně využívají.



Obrázek 6.2: Určení vzdálenosti mezi porty

- Transportní protokol, počet přenesených paketů, číslo AS, TCP příznaky – U těchto hodnot se jen porovnává, jestli jsou pro oba toky stejné nebo rozdílné. Pokud jsou stejné je vzdálenost $d(i,j)_f = 0$, pokud rozdílné $d(i,j)_f = 1$, kde f je jeden z vyjmenovaných atributů.
- Počet přenesených bajtů – Protože jde o numerický atribut, zkoumá se, jak blízké si jsou porovnávané hodnoty. Nejprve se počet bajtů namapuje na interval <0;1>. Mapování není rovnoměrné aby nedošlo k zanedbání rozdílů mezi výrazně nižšími hodnotami než je maximální hodnota v datech, více je vysvětleno v kapitole 2.2.1. V tomto případě je velikost prvního toku převedena na desetinné číslo, velikost dalšího toku také (viz Diagram 6.3) a pak je vypočten absolutní rozdíl desetinných hodnot, který je výsledkem porovnání velikosti toků.



Obrázek 6.3: Mapování velikosti v bytech na interval <0;1>

- Souřadnice IP adres – Porovnává se zvlášť vzdálenost zdrojových IP adres a cílových IP adres. Nejprve se vypočte vzdálenost IP adres v kilometrech a toto číslo se opět namapuje na interval <0;1>. Jako maximální hodnota je brána vzdálenost 12756 km což je maximální vzdálenost mezi body na Zemi. Takže výpočet rozdílnosti geografické polohy IP adres je proveden podle Rovnice 6.1.

$$d(i,j)_{GEO} = \frac{1}{12756} \times \text{geografická vzdálenost} \tag{6.1}$$

Pro nalezení shluků v načtených záznamech je použit algoritmus shlukové analýzy DBSCAN. Implementace DBSCANU v C je převzata od autora Gagarine Yaikhom [24] na GitHubu, kde se počítá s body v 3D prostoru. Algoritmus je upraven tak, aby používal vzdálenostní funkci, která provádí výpočet s proměnlivým počtem vybraných atributů NetFlow záznamů. Když už jsou zjištěné vzdálenosti atributů toků i a j, vypočítá se výsledná rozdílnost toků podle Rovnice 2.1.

6.3 Dosažené výsledky

Experimenty byly prováděny na datech zachycených na jedné z páteřních linek internetu. Za 5 minut bylo zachyceno asi sedm milionů NetFlow záznamů do jednoho *nfcapd* souboru. Z tohoto souboru se analyzovalo vždy sto tisíc záznamů vybraných vzorkováním z celého souboru. Problémem metody je exponenciální složitost výpočtu. Nejnáročnější částí výpočtu je vzdálenostní funkce. Počet volání této funkce roste s druhou mocninou počtu zpracovávaných toků. Pokud chceme analyzovat např. tisíc toků, bude se vzdálenostní funkce provádět milionkrát.

Vybrány byly pouze záznamy s transportním protokolem TCP, protože ten je pro útoky používán nejčastěji. Aplikaci lze použít pro hledání horizontálních i vertikálních skenů. Pro skenování je charakteristické, že se jeden útočník snaží připojit na spoustu obětí. Čím se toky patřící k útočníkovi liší jsou cílové adresy a porty. Co je naopak pro toky společné je zdrojová IP adresa, zdrojový port, počet přenesených paketů a TCP příznaky. Vzdálenostní funkce DBSCANu tedy počítala s těmito atributy.

$$d(i,j) = \frac{\delta_{src\_IP} * 0 + \delta_{dst\_IP} * 0 + \delta_{src\_port} * 0 + \delta_{dst\_port} * 0.2 + \delta_{flg} * 0 + \delta_{pkts} * 0}{\delta_{src\_IP} + \delta_{dst\_IP} + \delta_{src\_port} + \delta_{dst\_port} + \delta_{flg} + \delta_{pkts}}$$

Pokud budeme porovnávat dva toky, které mají všechny zkoumané atributy stejné až na cílové IP adresy (možný horizontální sken), bude čitatel v rovnici výpočtu rozdílnosti toků 2.1 vypadat takto: $\delta_{src\_IP}*0+\delta_{dst\_IP}*0.2+\delta_{src\_port}*0+\delta_{dst\_port}*0+\delta_{flg}*0+\delta_{pkts}*0$. Hodnota 0.2 je minimální vzdálenost hodnot u IP adres, ostatní vzdálenosti budou nulové, protože jsou hodnoty atributů stejné. Zvolená váha δ_{dst_IP} byla 0.8, ale šlo by použít i jiné číslo. Důležité je, že celý čitatel vychází 0.8 a ve jmenovateli zlomku je součet všech vah atributů. To znamená, že se nemění v závislosti na tom, jaké atributy mají zrovna toky společné nebo ne. Váha zdrojové IP adresy i zdrojových/cílových portů je zvolena stejně na 0.8. Váhy atributů pakety a TCP příznaky mají hodnotu 1, aby při jejich neshodnosti došlo k významnému navýšení rozdílnosti toků. Když se budou zkoumané toky lišit jen v cílovém portu, bude vzdálenost hodnot atributu cílový port také rovna nejméně 0.2. Váha tohoto atributu je stejná jako v případě cílové IP adresy, takže i rozdílnost toků bude

totožná. To stejné platí i pro zdrojové IP adresy a zdrojové porty. Když vypočítáme, jaká v tomto případě výsledná rozdílnost bude, vyjde přibližně 0.04. Pokud se toky budou lišit v paketech nebo TCP příznacích vyjde jejich rozdílnost minimálně přibližně 0.255. Když tedy pro DBSCAN nastavíme $\epsilon = 0.041$ nemůže se stát, že by v jednom shluku byly toky, které mají rozdílné TCP příznaky nebo počet paketů.

Výsledkem analýzy bylo 122 shluků, kde naprostá většina shluků je pouze legitimní komunikace např. jednoho klienta s webovým serverem. Nicméně 23 shluků vykazuje známky horizontálního skenování. Vertikální skenování nebylo nalezeno žádné. Ukázka shluku, který by mohl obsahovat toky patřící ke skenu je v Tabulce 6.2. Oproti legitimní komunikaci se liší v tom, že nejsou použity známé porty např. 443 pro HTTPS, ale registrované nebo dokonce dynamické porty. Dalším významným ukazatelem, že by se skutečně mohlo jednat o sken je, že záznamy v toku mají nastaveny jediný z TCP příznaků – SYN. Dále i počet přenesených paketů v každém toku je pouze jeden paket, což je pro skenování také typické.

cluster_id	src_addr:port	dst_addr:port	prot	bytes	pkts	src_country	dst_country	src_as	dst_as	tcp_flags
8	130.27.149.177:56396	59.223.67.170:3010	6	40	1	US	CN	0	0	....S.
8	130.27.149.177:56396	193.162.221.214:3010	6	40	1	US	DK	0	25384	....S.
8	130.27.149.177:56396	44.151.141.92:3010	6	40	1	US	US	0	7377	....S.
8	130.27.149.177:56396	108.146.232.130:3010	6	40	1	US	US	0	20057	....S.

Tabulka 6.2: Horizontální sken

Útočník nemusí cílit na náhodné IP adresy ale pokoušet se skenovat adresy patřící do jednoho AS. Pro nalezení takových shluků stačilo provést stejnou analýzu jako v případě hledání skenů IP adres, ale namísto atributu IP adresy byly použity AS. Opět se pracovalo se sto tisíci záznamy. Aby se daly výsledky lépe porovnat analýza byla provedena na stejných záznamech jako v prvním experimentu. Výsledkem bylo jen 6 shluků. S velkou pravděpodobností jde skutečně o skeny, znaky jsou stejné jako v předešlém případě. Někdy se stane, že se v databázi s AS nebo souřadnicemi nepovede najít informace pro danou IP adresu, proto jsou čísla zdrojových AS v Tabulce 6.3 nulová.

cluster_id	src_addr:port	dst_addr:port	prot	bytes	pkts	src_country	dst_country	src_as	dst_as	tcp_flags
5	252.68.10.81:46323	44.151.157.66:10000	6	40	1	none	US	0	7377	....S.
5	252.68.10.81:46323	44.151.129.112:10000	6	40	1	none	US	0	7377	....S.
5	252.68.10.81:46323	44.38.118.43:10000	6	40	1	none	US	0	7377	....S.
5	252.68.10.81:46323	44.40.74.53:10000	6	40	1	none	US	0	7377	....S.

Tabulka 6.3: Horizontální sken

Dalším pokusem bylo odhalit v datech DDOS útok. Konfigurace byla taková, že se zkoumaly jen cílové IP adresy a TCP příznaky. Tyto atributy měly záznamy ve shlucích stejné. Výsledkem bylo 1045 shluků. K DDOS útoku ale nejspíš patří jen 6. Část shluku s

DDOS útokem je vidět v Tabulce 6.4. Obětí je pravděpodobně webový server na portu 80. Je vidět, že na něj různí útočníci odesílají pakety s jediným TCP příznakem a sice SYN.

cluster_id	src_addr:port	dst_addr:port	prot	bytes	pkts	src_country	dst_country	src_as	dst_as	tcp_flags
455	143.1.222.133:14752	108.146.110.190:80	6	96	2	US	US	11003	20057	....S.
455	221.179.40.147:50505	108.146.110.190:80	6	152	3	CN	US	56040	20057	....S.
455	221.171.157.65:28639	108.146.110.190:80	6	96	2	JP	US	2518	20057	....S.
455	74.96.48.234:1736	108.146.110.190:80	6	192	3	US	US	701	20057	....S.

Tabulka 6.4: Horizontální sken

Kapitola 7

Závěr

Pro hledání podobností v síťových datech byla vyzkoušena jen shluková analýza pomocí DBSCANu. Tuto metodu skutečně lze použít pro hledání jistých síťových útoků. Konkrétně skenování portů, SYN Flood, DOS, DDOS a útoků podobného charakteru, které se nějak projeví v NetFlow datech. K detekci útoků nebo hledání závislostí, které nelze nalézt jednoduchým vyfiltrováním síťových dat se nepodařilo využít geografickou vzdálenost. Vytvořená aplikace nebyla vyzkoušena na datech z lokální sítě, kde by teoreticky bylo možné odhalovat další útoky, např. tzv. DHCP spoofing. Problémem vyvinuté metody je, že výsledkem nejsou jen záznamy patřící k útokům, ale i spoustu běžné komunikace. Do budoucna by bylo dobré provádět po analýze ještě odfiltrování běžných/nezajímavých dat. Velmi limitující je exponenciální složitost výpočtu. Na běžném počítači lze v únosném čase zpracovat kolem sto tisíc záznamů.

Literatura

[1] *GeoLite2 Free Downloadable Databases* [online]. [cit. 25. března 2018]. Dostupné na: https://dev.maxmind.com/geoip/geoip2/geolite2/.

[2] *MaxMind* [online]. [cit. 25. března 2018]. Dostupné na: https://www.maxmind.com/en/home.

[3] *NetFlow* [online]. [cit. 19. dubna 2018]. Dostupné na: https://cs.wikipedia.org/wiki/NetFlow.

[4] *NetFlow* [online]. [cit. 19. dubna 2018]. Dostupné na: https://www.ntop.org/products/traffic-analysis/ntop/.

[5] *NETFLOW EXPORT FORMAT* [online]. [cit. 15. dubna 2018]. Dostupné na: http://netflow.caligare.com/netflow_format.htm.

[6] *Nfdump* [online]. [cit. 27. dubna 2018]. Dostupné na: https://github.com/phaag/nfdump.

[7] *NfSen - Netflow Sensor* [online]. [cit. 21. dubna 2018]. Dostupné na: http://nfsen.sourceforge.net/#mozTocId642821.

[8] *Nprobe – An Extensible NetFlow v5/v9/IPFIX Probe for IPv4/v6* [online]. [cit. 19. dubna 2018]. Dostupné na: https://www.ntop.org/products/netflow/nprobe/.

[9] *Pandora FMS Monitoring– NetFlow* [online]. [cit. 19. dubna 2018]. Dostupné na: https://wiki.pandorafms.com/index.php?title=Pandora:Documentation_en:Netflow.

[10] *TCP Transport* [online]. [cit. 19. dubna 2018]. Dostupné na: https://intronetworks.cs.luc.edu/current/html/tcp.html.

[11] *Wireshark* [online]. [cit. 21. dubna 2018]. Dostupné na: https://www.wireshark.org/.

[12] *Transmission Control Protocol* [online]. 1981 [cit. 25. dubna 2018]. RFC, 793. Dostupné na: https://tools.ietf.org/html/rfc793.

[13] *Sampled NetFlow* [online]. 2003 [cit. 10. dubna 2018]. Dostupné na: https://www.cisco.com/c/en/us/td/docs/ios/12_0s/feature/guide/12s_sanf.html.

[14] *Cisco Systems NetFlow Services Export Version 9* [online]. říjen 2004 [cit. 13. dubna 2018]. 2 s. RFC, 3954. Dostupné na: https://www.ietf.org/rfc/rfc3954.txt.

[15] *NetFlow Services Solutions Guide* [online]. 2007 [cit. 12. dubna 2018]. Dostupné na: https://www.cisco.com/en/US/products/sw/netmgtsw/ps1964/products_implementation_design_guide09186a00800d6a11.html#wp1030045.

[16] *Application Monitoring Using NetFlow* [online]. 2013 [cit. 21. dubna 2018]. Dostupné na: https://www.cisco.com/c/dam/en/us/td/docs/solutions/CVD/Dec2013/CVD-ApplicationMonitoringUsingNetFlowDesignGuide-DEC13.pdf.

[17] *Special-Purpose IP Address Registries* [online]. Duben 2013 [cit. 12. dubna 2018]. 6-20 s. RFC, 6890. Dostupné na: https://tools.ietf.org/html/rfc6890.

[18] *Libnf Documentation* [online]. 2015 [cit. 8. března 2018]. Dostupné na: http://libnf.net/doc/api/.

[19] *Guidelines for Writing an IANA Considerations Section in RFCs* [online]. červen 2017 [cit. !2018-05-1]. RFC, 8126. Dostupné na: https://tools.ietf.org/html/rfc8126.

[20] *Service Name and Transport Protocol Port Number Registry* [online]. Duben 2018 [cit. 12. dubna 2018]. RFC, 6335. Dostupné na: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[21] Chappell, L. a Tittel, E. *Guide to TCP/IP*. [b.m.]: Course Technology, 2002. ISBN 9780619035303.

[22] Erickson, J. *Hacking: The Art of Exploitation*. [b.m.]: William Pollock, No Starch Press, Inc., 2008. 251–258 s. ISBN 978-1-59327-144-2.

[23] Ester, M., Kriegel, H.-P., Sander, J. et al. A Density-Based Algorithm for Discovering Clusters. In *KDD-96 Proceedings*. 1996. Dostupné na: http://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf.

[24] (gyaikhom), G. Y. *The DBSCAN Clustering Algorithm* [online]. 2015 [cit. 8. března 2018]. Dostupné na: https://github.com/gyaikhom/dbscan.

[25] Han, J., Kamber, M. a Pei, J. *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012. 72-76 s. ISBN 978-0-12-381479-1.

[26] Ing. Petr Grygárek, P. Směrovací protokol BGP. [online]. 2005 [cit. 22. dubna 2018]. Dostupné na: http://www.cs.vsb.cz/grygarek/SPS/lect/BGP/BGP.html.

[27] Kretchmar, J. a Dostálek, L. *Administrace a diagnostika sítí pomocí OpenSource utilit a nástrojů*. [b.m.]: Computer Press, 2004. ISBN 80-251-0345-5.

[28] Kriz, I. a Pultr, A. *Introduction to mathematical analysis*. [b.m.]: Springer Basel, 2013. 33 s. ISBN 978-3-0348-0635-0.

[29] Kumar, S., Dharmapurikar, S., Yu, F. et al. Algorithms to Accelerate Multiple Regular Expressions Matching for Deep Packet Inspection. *SIGCOMM Comput. Commun. Rev.* Srpen 2006, roč. 36, č. 4. S. 340. Dostupné na: http://doi.acm.org/10.1145/1151659.1159952. ISSN 0146-4833.

[30] Matoušek, P. *Síťové aplikace a jejich architektura.* [b.m.]: Nakladatelství Vysokého učení technického v Brně VUTIUM, 2014. 31–33 s. ISBN 978-80-214-3766-1.

Příloha A

Obsah CD

- sources.zip - zdrojové kódy aplikace
- zaverecna_prace.zip - zdrojové soubory k tomuto textu v LaTeXu