VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

DATOVÁ SADA PRO KLASIFIKACI SÍŤOVÝCH ZAŘÍZENÍ POMOCÍ STROJOVÉHO UČENÍ

DATASET FOR CLASSIFICATION OF NETWORK DEVICES USING MACHINE LEARNING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. PAVEL EIS**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MARTIN ŽÁDNÍK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačových systémů (UPSY) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Eis Pavel, Bc.**
Program: Informační technologie a umělá inteligence Specializace: Kybernetická bezpečnost
Název: **Datová sada pro klasifikaci síťových zařízení pomocí strojového učení**
Dataset for Classification of Network Devices Using Machine Learning
Kategorie: Počítačové sítě
Zadání:

1. Nastudujte projekt ADiCT organizace CESNET zabývající se vytvářením profilů síťových zařízení.
2. Vyhledejte externí zdroje dat využitelné pro klasifikaci síťových zařízení.
3. S využitím výše zmíněných zdrojů dat navrhněte postup anotace datové sady pro klasifikaci síťových zařízení.
4. Implementujte nástroje podporující anotaci datové sady.
5. Vytvořte anotovanou datovou sadu obsahující relevantní štítky o síťových zařízeních a data o síťových tocích těchto zařízení ze stejného časového období.
6. Demonstrujte využitelnost datové sady aplikací vybrané metody strojového učení, která by pouze s využitím dat o síťových tocích odvozovala vybraný štítek či štítky popisující zařízení.
7. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího rozvoje a využití datové sady.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění bodů 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Žádník Martin, Ing., Ph.D.**
Vedoucí ústavu: Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 30. října 2020

Abstrakt

Automatická klasifikace zařízení v počítačové síti lze využít pro detekci anomálií v síti a také umožňuje aplikaci bezpečnostních politik dle typu zařízení. Pro vytvoření klasifikátoru zařízení je stěžejní kvalitní datová sada, jejichž veřejná dostupnost je nízká a tvorba nové datové sady je složitá. Cílem práce je vytvořit nástroj, který umožní automatizovanou anotaci datové sady síťových zařízení a vytvoření klasifikátoru síťových zařízení, který využívá pouze základní údaje o síťových tocích. Výsledkem této práce je modulární nástroj poskytující automatizovanou anotaci síťových zařízení využívající systém ADiCT sdružení Cesnet, vyhledávače Shodan a Censys, informace ze služeb PassiveDNS, TOR, WhoIs, geolokační databáze a informace z blacklistů. Na základě anotované datové sady je vytvořeno několik klasifikátorů klasifikujících síťová zařízení podle používaných služeb. Výsledky práce nejen výrazně zjednodušují proces vytváření nových datových sad síťových zařízení, ale zároveň ukazují neinvazivní přístup ke klasifikaci síťových zařízení.

Abstract

Automatic classification of devices in computer network can be used for detection of anomalies in a network and also it enables application of security policies per device type. The key to creating a device classifier is a quality data set, the public availability of which is low and the creation of a new data set is difficult. The aim of this work is to create a tool, that will enable automated annotation of the data set of network devices and to create a classifier of network devices that uses only basic data from network flows. The result of this work is a modular tool providing automated annotation of network devices using system ADiCT of Cesnet's association, search engines Shodan and Censys, information from PassiveDNS, TOR, WhoIs, geolocation database and information from blacklists. Based on the annotated data set are created several classifiers that classify network devices according to the services they use. The results of the work not only significantly simplify the process of creating new data sets of network devices, but also show a non-invasive approach to the classification of network devices.

Klíčová slova

síťová zařízení, monitorování sítě, anotace datové sady, strojové učení, klasifikace síťových zařízení, statistické chování zařízení, ADiCT, Shodan, Censys, PassiveDNS, WhoIs, GeoIP, TOR

Keywords

network devices, network monitoring, dataset anotation, machine learning, classification of network devices, statistical behavior of device, ADiCT, Shodan, Censys, PassiveDNS, WhoIs, GeoIP, TOR

Citace

EIS, Pavel. *Datová sada pro klasifikaci síťových zařízení pomocí strojového učení.* Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Martin Žádník, Ph.D.

Datová sada pro klasifikaci síťových zařízení pomocí strojového učení

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Martina Žádníka, Ph.D. Další informace mi poskytl konzultant specialista Ing. Václav Bartoš, Ph.D. ze sdružení Cesnet. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Pavel Eis
13. května 2021

Poděkování

Rád bych poděkoval Ing. Martinu Žádníkovi, Ph.D. za odborné vedení této diplomové práce a za všechny jeho rady a čas, který mi věnoval v průběhu konzultací. Poděkování také patří Ing. Václavu Bartošovi, Ph.D. ze sdružení Cesnet za pravidelné konzultace a rady v průběhu vývoje. Velké poděkování si zaslouží také rodina a všichni ostatní, kteří mě v průběhu práce podporovali.

Obsah

1 Úvod **3**

2 Projekt ADiCT - Asset Discovery, Classification and Tagging **5**
2.1 Architektura komponent projektu ADiCT 6
2.2 Způsob uložení informací o entitách v databázi 8
2.3 Sbíraná data o síťových zařízeních 9

3 Externí zdroje dat využitelné pro klasifikaci síťových zařízení **12**
3.1 Vyhledávač Shodan 13
3.2 Vyhledávač Censys 15
3.3 Služba WhoIs 17
3.4 Geolokační databáze IP adres 19
3.5 Seznam blokovaných identifikátorů - Blacklist 19
3.6 Služba PassiveDNS 20
3.7 TOR - The Onion Router 21

4 Existující přístupy ke klasifikaci síťového provozu a zařízení **25**
4.1 Klasifikace síťových zařízení pomocí statistického chování 26
4.2 Algoritmus strojového učení Náhodný les 28

5 Návrh anotace síťových zařízení v datové sadě **30**
5.1 Typ zařízení a podporované služby 30
5.2 Otevřené aplikace a název operačního systému 34
5.3 Geografická data a informace ze systému WhoIs 35
5.4 Anotace zařízení pomocí služby PassiveDNS 36
5.5 Anotace s využitím blacklistů a služby TOR 37

6 Implementace anotátoru datové sady síťových zařízení a vytvoření datové sady **39**
6.1 Zpracování vstupu anotátoru síťových zařízení 39
6.2 Implementace datových zdrojů pro anotaci 40
6.3 Implementace integrace datových zdrojů 43
6.4 Testování anotátoru a optimalizace výkonnosti 45
6.5 Analýza vytvořené datové sady sítových zařízení 46

7 Klasifikace síťových zařízení s využitím metody strojového učení **48**
7.1 Předzpracování datové sady a výpočet rysů pro strojové učení 48
7.2 Trénování klasifikačního modelu a provedené experimenty 51

7.3 Zhodnocení dosažených výsledků a možností dalšího rozvoje datové sady . . 55

8 Závěr **58**

Literatura **59**

A Obsah přiloženého paměťového média - CD **62**

B Seznam typů síťových zařízení **63**

C Dodatečné informace o datové sadě a k experimentům klasifikace **66**

Kapitola 1

Úvod

Při efektivní správě počítačové sítě je velmi důležité mít přehled o zařízeních, která jsou do monitorované počítačové sítě připojeny. Jedním ze způsobů, kterým lze získat ucelený pohled o zařízeních v síti, je sledování jejich krátkodobého i dlouhodobého chování. Tento přístup umožňuje vytvářet profily monitorovaných zařízení a zároveň klasifikovat typ zařízení dle určitých charakteristik (např. se jedná o webový server, tiskárnu, ...). Tyto informace slouží správci pro detekci běžného a naopak neobvyklého chování zařízení, která jsou připojena do sítě. Neobvyklé chování zařízení (často zvané *anomálie*) může indikovat nebezpečnou aktivitu (např. škodlivé chování nebo špatnou konfiguraci zařízení), která vyžaduje zvýšenou pozornost správce. Zároveň jde klasifikovaná zařízení dělit do skupin s podobnými vlastnostmi. To poskytuje lepší přehled o síti a na dané skupiny lze poté aplikovat společné bezpečnostní politiky.

Počty zařízení, která jsou zapojena do počítačové sítě, mnohdy přesahují stovky i tisíce. Je nutné zapojit automatizované nástroje, které umožňují bez větších zásahů správce automaticky klasifikovat zařízení v monitorované síti. Tyto nástroje velmi často využívají metody strojového učení. Aby byly metody strojového učení dostatečně přesné a efektivní, je nutné mít k dispozici dostatek trénovacích dat (jako celek zvané *datová sada*), které metody strojového učení potřebují k trénování, testování a zlepšování jejich výsledků, aby mohly být nasazeny do reálného provozu.

V případě volby datové sady je možné si vytvořit úplně novou datovou sadu. Proces vytváření datové sady je ale mnohdy velmi pracný, protože je nutné datovou sadu oštítkovat (*anotovat*). Proces štítkování obnáší přiřazení požadovaných výsledných kategorií klasifikace (štítků), které chceme dosáhnout, ke zkoumaným datům. Tímto přidáme datům důležité informace, které metody strojového učení využívají ke svému učení. Často je obtížné přiřadit správný štítek k zařízení a zároveň je štítek potřeba přiřadit ke každému záznamu. Proto je výhodné použít již existující datovou sadu. Veřejně dostupných datových sad je ale z mnoha důvodů (např. pracnost vytváření, příliš citlivá data na zveřejnění) v dané oblasti velmi málo. Tento fakt ztěžuje nový výzkum v dané oblasti a je složité navázat na výzkum předchozí, protože je obtížné objektivně porovnat výsledky navazujícího výzkumu, pokud není veřejně přístupná výchozí datová sada, na které byl proveden původní výzkum.

Cílem této práce je vytvořit nástroj, který zjednoduší tvorbu nových datových sad v oblasti klasifikace síťových zařízení a především pomůže s jejich anotací. Kapitola 2 představuje projekt ADiCT sdružení Cesnet, který slouží pro zpracování a vytváření profilů o síťových entitách. V kapitole 3 jsou představeny externí služby, které budou společně s projektem ADiCT využity ke správnému oštítkování datové sady. Kapitola 4 shrnuje existující přístupy ke klasifikaci síťových zařízení s důrazem na statistické přístupy, které

většinou využívají metody strojového učení. V následující kapitole (kapitola 5) je popsán návrh využití systému ADiCT a externích služeb pro anotaci síťových zařízení v datové sadě a celkový přehled štítků, které budou pomocí těchto zdrojů jednotlivým zařízením přiřazeny. Navazující kapitola 6 představuje všechny vlastnosti a funkce vytvořeného nástroje pro automatizovanou anotaci, včetně jeho funkčního a výkonnostního testování. Ve stejné kapitole se dále nachází analýza vytvořené datové sady, která je anotována s využitím vytvořeného automatizovaného anotátoru. V poslední fázi této práce (7) je popsán způsob využití vytvořené datové sady k natrénování klasifikačních modelů a různé experimenty a jejich výsledky, které byly provedeny nad datovou sadou.

Kapitola 2

Projekt ADiCT - Asset Discovery, Classification and Tagging

Návrh projektu ADiCT (Asset Discovery, Classification and Tagging)[1] byl zahájen sdružením Cesnet v druhé polovině roku 2019. Implementace projektu a drobné úpravy v návrhu pokračují v letech 2020 a 2021 a s velkou pravděpodobností budou pokračovat i dále. Hlavním cílem projektu je implementace systému, který umožňuje sběr informací o zařízeních v síti především pomocí pasivního monitorování provozu, ale i získáváním dat z externích zdrojů a následná analýza těchto dat k získání dodatečných užitečných vysokoúrovňových informací o zařízeních v síti. Implementovaný systém slouží pro podporu vývoje algoritmů a metod v oblasti vytváření síťových profilů zařízení.

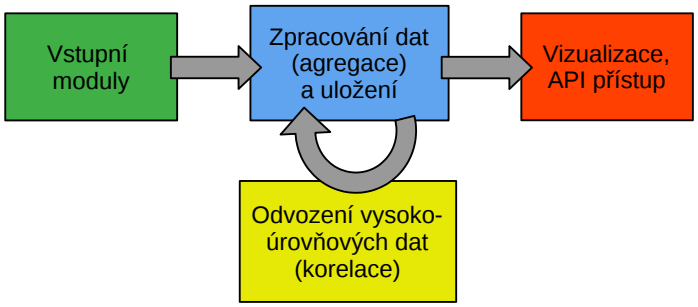
Systém je stále ve vývoji, ale již nyní poskytuje užitečné informace o síťových zařízeních. Mezi takové informace patří např. seznam otevřených portů daného zařízení. Systém poskytuje i další informace o síťových zařízeních a jejich popis je uveden v sekci 2.3. Všechna tato data lze určitým způsobem využít jako zdroj informací o síťovém zařízení, který lze využít pro anotaci datové sady, která bude vytvořena v rámci této diplomové práce. Zároveň bude systém ADiCT využívat i automatizovaný nástroj, který bude provádět anotaci datové sady. Postupným vývojem bude systém ADiCT poskytovat více a více relevantních informací, které mohou být (mimo jiné) použity pro anotaci datových sad.

V budoucnosti může být systém použit jako produkční služba a může tak pomoci administrátorům sítě při řešení různých incidentů. Nasazení systému jako produkční služby je prozatím pouze jeden z možných sekundárních cílů a hlavní důraz je kladen na podporu výzkumu v dané oblasti. Dalším sekundárním cílem projektu je detekce nežádoucích stavů či změn u síťových zařízení. Pokud by byla detekována určitá anomálie, mohlo by dojít k nahlášení této anomálie do interního systému Warden[2], který slouží pro sdílení informací o detekovaných bezpečnostních událostech. Pokud například určitý segment sítě slouží pro vědecké výpočty a najednou začne nějaké zařízení z tohoto segmentu komunikovat se serverem těžícím kryptoměnu Bitcoin, pravděpodobně se jedná o odchylku od normálního chování. Těžba kryptoměn není škodlivá aktivita, ale v některých sítích by se neměla vyskytovat.

[1] Informace o projektu ADiCT jsou čerpány z interních wiki stránek projektu: https://redmine.liberouter.org/projects/adict?jump=welcome

[2] https://warden.cesnet.cz/

Informace o jednotlivých zařízeních je možné využít k agregaci těchto dat do větších celků, např. získáním přehledu o celém prefixu sítě, popř. o organizaci. Některý prefix sítě může reprezentovat např. datacentrum. Dále je jedním z cílů uchovávat vztahy mezi entitami, aby bylo viditelné, které zařízení komunikuje s kterým. Z těchto informací lze vyvodit, které zařízení se připojuje k určitým serverům. Dále je možné vytvořit mapu závislostí, z které může být vidět, že některé zařízení je závislé na určitém serveru a vyvodit kritičnost služeb v síti.



Obrázek 2.1: Průběh zpracování dat projektem ADiCT. Obrázek je převzat z interních wiki stránek projektu ADiCT.

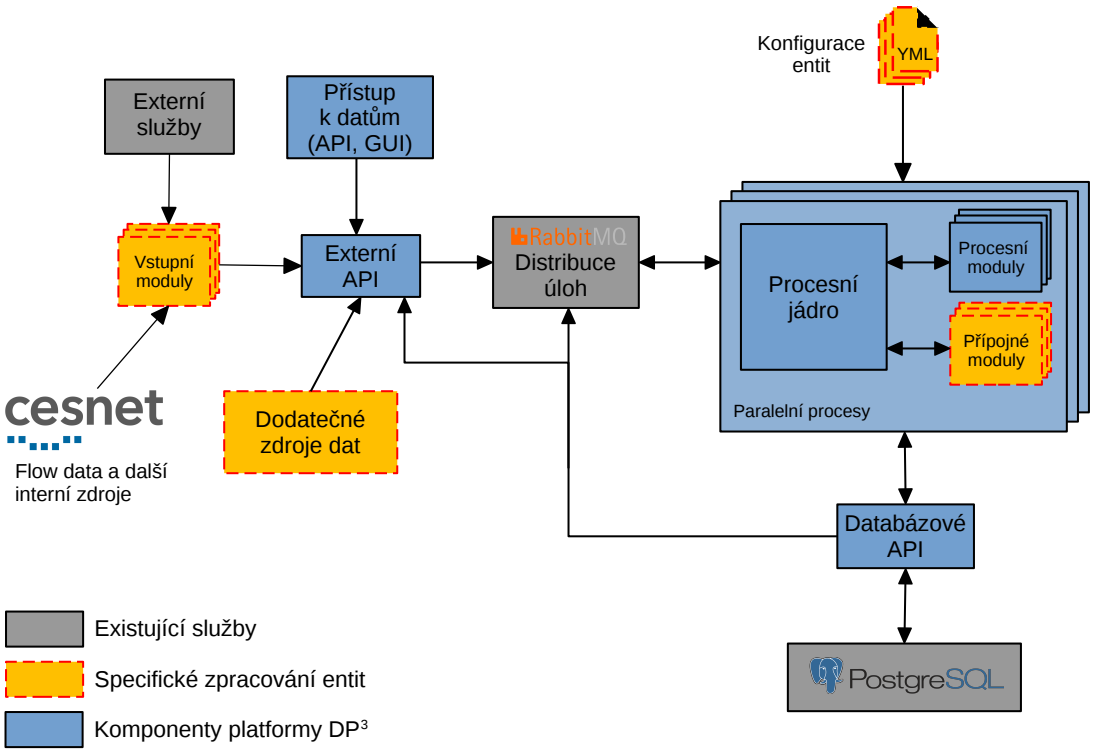
Na obrázku 2.1 je znázorněn zjednodušený proces zpracování dat projektem ADiCT. V projektu je vytvořeno několik vstupních modulů, které analyzují a zpracovávají interní data sdružení Cesnet nebo data z externích zdrojů. Tyto data jsou poté uložena a dohromady tvoří informace o zařízeních v síti. Mezi ukládané informace patří např. typ zařízení (server, uživatelská stanice, ...), název operačního systému, který běží na daném zařízení, seznam služeb, které zařízení poskytuje skrze otevřené porty a další. Zároveň jsou uložená data agregována do časových intervalů, protože v systému jsou uchována i historická data. Nad uloženými daty jsou prováděny různé analýzy, jejichž výstupem jsou další informace o uložených entitách. Ke všem uloženým datům je poté umožněn přístup skrze komunikační rozhraní API (Application Programing Interface) a v budoucnu je v plánu data i vizualizovat.

2.1 Architektura komponent projektu ADiCT

V rámci projektu ADiCT vznikla i samostatná platforma pro zpracování informací o různých entitách nazývaná DP^3 (Dynamic Profile Processing Platform, exponent 3 je součástí názvu, nejedná se o poznámku pod čarou). Jádro platformy vychází z implementace procesního jádra reputační databáze NERD[3] (Network Entity Reputation Database) organizace Cesnet. Reputační databáze NERD zpracovává podobná data o entitách jako projekt ADiCT. Tento fakt byl jedním z hlavních důvodů pro vývoj společné obecné platformy DP^3, která bude využita pro projekt ADiCT, ale bude na ní přenesen i systém NERD. Zároveň tato platforma může sloužit pro rychlý vývoj prototypu dalších podobných systému, které zpracovávají informace o síťových entitách.

Obsahem této platformy je komunikační rozhraní API, které zpracovává příchozí požadavky na zpracování nových dat o spravovaných entitách. Pokud dodržují vstupní požadavky správný formát, jsou rozhraním API transformovány na úkoly pro procesní jádro

[3] https://nerd.cesnet.cz/



Obrázek 2.2: Průběh zpracování dat projektem ADiCT. Architektura se skládá ze vstupních modulů, které zpracovávají vstupní data do formy, která se uchovává uvnitř platformy. Data jsou odeslána skrze externí API do platformy, která data zpracuje a spustí různé přípojné moduly, které přidají k datům dodatečné informace a výsledek je uložen v databázi PostgreSQL.

a vloženy do front distribučního systému zpráv RabbitMQ[4]. Procesní jádro platformy s přípojnými moduly běží v několika vláknech a je možné jej spouštět vícekrát paralelně. Tím je zaručeno rychlé a škálovatelné zpracování velkého objemu dat. Proto je nutné používat systém pro distribuci zpráv, aby mezi sebou mohly jednotlivá vlákna a procesy jádra efektivně komunikovat. Procesní jádro si vždy vyjme úkol z fronty požadavků systému RabbitMQ a zpracuje ho. Jednotlivé úkoly pro procesní jádro zpravidla obsahují pokyn k přidání nebo úpravě dat o určité entitě nebo vyvolání určité události. Na změnu uložených atributů mohou reagovat moduly vyvoláním dalších úkolů pro procesní jádro (např. pokud je procesním jádrem přidán záznam o nové entitě, mohou k novému záznamu některé moduly automaticky získat dodatečná data). Jeden úkol tak může vyvolat kaskádu dalších změn a událostí.

Samotná platforma DP^3 dokáže zpracovat různé entity, jejichž strukturu lze definovat v konfiguraci platformy. Touto konfigurací je definován používaný datový model a platforma dle konfigurace vytvoří databázové schéma při spuštění platformy. Databázové schéma se automaticky aktualizuje při aktualizaci konfigurace entit. Pro uložení dat je zvolen objektově relační databázový systém PostgreSQL. Platforma je zároveň přizpůsobena na vytvoření a připojení modulů, které slouží pro upravování entit ukládaných uvnitř platformy, takže je jednoduché definovat a upravovat vlastní správu uložených entit. Celá platforma

[4] https://www.rabbitmq.com/

je implementována v programovacím jazyce Python a proto je potřeba přípojné moduly platformy upravující entity implementovat také v programovacím jazyce Python. Pomocí platformy DP^3 je tedy možné ukládat data o různých entitách, charakter entit a způsob zpracování entit definuje konfigurace platformy a připojené moduly. Zároveň platforma poskytuje rozhraní API pro přístup k uloženým datům, které může být využito pro vizualizaci dat v grafickém uživatelském rozhraní.

Na obrázku 2.2 je vidět kompletní architektura projektu ADiCT. Projekt využívá pro zpracování informací o entitách platformu DP^3, která je sice ještě ve vývoji, ale hlavní komponenty jsou již funkční. V rámci projektu ADiCT vznikají především přípojné moduly analyzující a upravující entity uložené v databázi a vstupní moduly, které získávají základní data o entitách.

2.2 Způsob uložení informací o entitách v databázi

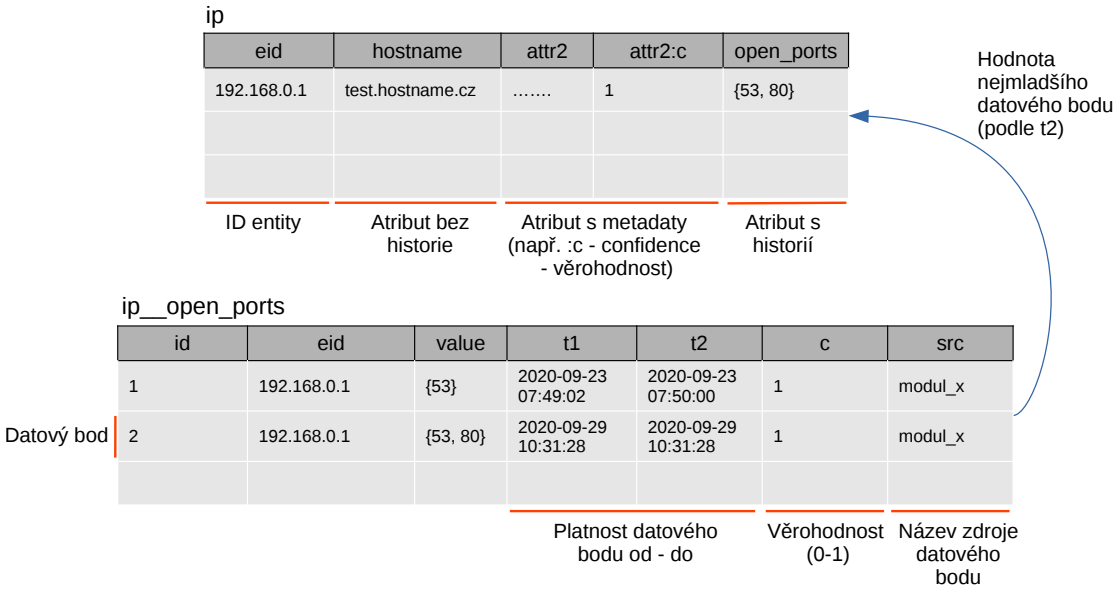
Pro běh celého projektu je nutné vytvořit konfiguraci entit, která definuje jejich strukturu v databázi. Pro každou entitu je nutné vytvořit zvláštní konfigurační soubor, v kterém jsou definovány vlastnosti entity (především její jméno a identifikátor) a seznam všech atributů včetně jejich konfigurace, které se budou o dané entitě uchovávat. Z celé konfigurace atributu je důležitý zvolený datový typ atributu (řetězec, celé číslo, seznam celých čísel, ...) a nastavení historie atributu, protože o všech atributech entity mohou být uložena i historická data v podobě tzv. **datových bodů** (data-point). Na výpisu 2.1 je vidět ukázka konfiguračního souboru entity, podle kterého je vytvořeno databázové schéma:

```
entity:
  id: ip
  name: ip address
  key_data_type: string
  auto_create_record: false
attribs:
  open_ports:
    name: Open ports
    description: List of open and actively used ports on the device
    data_type: array<int>
    history: true
    history_params:
      max_age: 7d
  activity_flows:
    name: Activity (flows)
    ...
```

Výpis 2.1: Ukázka konfigurace entity IP adresy s jejími atributy. U konfigurace atributu `open_ports` je vidět požadavek na uchování historie a konfigurace maximálního stáří datového bodu, který může být využit pro stanovení aktuální hodnoty, na 7 dní. Konfigurace je napsána ve formátu YAML[6].

Pro každý druh entity je následně automaticky procesním jádrem vytvořena vlastní tabulka se sloupci odpovídajícími jednotlivým atributům. Jeden záznam tabulky představuje

[6] https://yaml.org/



Obrázek 2.3: Ukázka struktury uložených dat o entitách v databázi. Na obrázku jsou vidět dvě databázové tabulky. Horní tabulka obsahuje ukázku atributů ukládaných o entitách IP adres a dolní tabulka obsahuje historické datové body atributu `open_ports` entit IP adres. Důležitým aspektem názorné ukázky uložení dat je volba aktuální hodnoty atributu `open_ports` k entitě IP adresy eid = 192.168.0.1. Jedná se o výběr nejmladšího datového bodu podle časové značky `t2` z tabulky `ip__open_ports`.

jednu entitu a její aktuální vlastnosti. Pro každý atribut entity, o kterém má být uchována historie (datové body), je také vytvořena vlastní tabulka s názvem skládajícím se z názvu entity a historického atributu složeného dvěmi podtržítky. Každý **datový bod** představuje historickou hodnotu atributu, která byla platná v časovém intervalu s hraničními hodnotami `t1` a `t2` (hodnota byla platná od `t1` do `t2`). Hraniční hodnota `t2` je volitelná a pokud není k dispozici, automaticky je do hodnoty `t2` dosazena hodnota `t1`, tedy `t2` = `t1` (hodnota platila pouze v jeden daný okamžik a ne v časovém intervalu). Jednotlivé datové body jsou uloženy v tabulce pro historická data atributu. Celý tento princip je lépe viditelný na obrázku 2.3. Na obrázku je vidět i zvolení aktuální hodnoty atributu `open_ports` entity `ip` v hlavní tabulce. Za aktuální hodnotu je zvolena nejmladší (nejaktuálnější) hodnota ze všech datových bodů vzhledem k hraniční hodnotě `t2` platnosti datového bodu. V konfiguraci atributu může být nastavena i maximální doba, po kterou může být historický datový bod nastaven jako aktuální hodnota. Např. pokud je datový bod starší než 7 dní, je jako aktuální hodnota nastavena hodnota NULL.

2.3 Sbíraná data o síťových zařízeních

Jedním z účelů využití nasbíraných dat je budoucí výzkum metod pro analýzu síťového provozu a profilování zařízení. Celkově se tento výzkum zaměřuje na zpracování vstupních dat z interních i externích zdrojů, jehož výsledkem jsou algoritmy a moduly přiřazující na základě těchto vstupních datech různé štítky charakterizující zařízení.

Projekt ADiCT aktuálně používá již několik vstupních modulů, které generují data o entitách IP adresy (aktuálně jsou podporovány IP adresy verze 4 i 6, ale IP verze 4 v uložených datech převažuje) a MAC adresy. Modul **IP Activity** sleduje aktivitu zahrnující data o tocích zařízení (IP adres) sdružení Cesnet. Modul v pevných časových intervalech (aktuálně 10 minut) počítá počet toků, paketů a bytů, které pochází z daných zařízení. Na konci každého intervalu odešle 3 datové body do vstupního rozhraní API obsahující jednotlivé počty pro každou aktivní IP adresu v daném časovém úseku. Atributy jsou poté uloženy v databázi pod názvem `activity_flows`, `activity_packets` a `activity_bytes`.

Modul **Shodan monitor** analyzuje emaily od služby Shodan Monitor[7]. Vyhledávač Shodan pravidelně skenuje globální rozsah IP prostoru a získává informace o dostupných službách na daných zařízeních skrze otevřené porty a banery služeb běžících na daných portech (vyhledávač Shodan je podrobněji popsán v kapitole 3.1). Služba Shodan Monitor umožňuje nastavit monitorování jednotlivých zařízení v síti nebo celé podsítě a generuje emailové hlášení o otevřených portech a dostupných službách na sledovaných zařízeních. Při volbě služby lze zvolit typ notifikací, o které má uživatel služby zájem. Mezi dostupné notifikace patří např. `uncommon`, zahrnující notifikace o neobvyklých službách veřejně dostupných na daném zařízení, které by obecně neměly být běžně dostupné, nebo notifikace `malware`, které upozorňují na zařízení, které jsou pravděpodobně kompromitovány nebo na nich běží škodlivý program.

Dalším modulem aktuálně nasazeným v projektu ADiCT je modul **SDP Analyzer**. Modul analyzuje data z rodiny protokolů objevujících dostupné služby v počítačové síti (SDP - Service Discovery Protocol). Tento protokol se ale pohybuje pouze v lokální síti a tyto informace tedy nejsou využitelné pro anotaci datové sady, protože nejsou dostupné pro globální IP adresy, pro které bude prováděn záchyt provozu, který bude součástí datové sady.

Modul **Service labels** sleduje provoz jednotlivých zařízení v síti a extrahuje čísla portů ze zachycené komunikace těchto zařízení. Podle aktivně využívaných čísel portů dohledá název používané služby podle registru názvů služeb a čísel portů transportních protokolů[8]. Pokud aktivně využívané číslo portu je registrované nějakou službou, je poté toto číslo uloženo u daného zařízení do atributu `open_ports`. Zároveň jsou čísla aktivních portů převedena na názvy korespondujících protokolů, z kterých je poté odvozen obecnější název role zařízení (štítek), která definuje charakter daného zařízení. Např. na portu 6600 běžně komunikuje virtualizační služba Hyper-V operačního systému Microsoft Windows. Na základě této znalosti je zařízení přiřazen štítek `Windows`. Stejný štítek může být přiřazen i službě Microsoft Windows Internet Name Server běžící na portu 1512. Mapování jednotlivých služeb na obecnější štítky je popsáno v bakalářské práci Josefa Koumara [13], včetně shrnutí všech používaných štítků.

Poslední plně funkční modul se nazývá **Dependency mapping**. Cílem tohoto modulu je sledovat závislosti v komunikaci mezi jednotlivými zařízeními. Předmětem zkoumaných závislostí jsou IP adresy dvou komunikujících zařízení, ale také čísla portů, přes které proudí komunikace. Modul nejprve ověřuje, zda je komunikace spojitá, aby jakýkoliv samostatný tok nevytvářel záznam o závislosti mezi zařízeními. Pro služby využívající transportní protokol UDP je nutné překročit určitý počet paketů a u služeb, využívajících transportní protokol TCP, modul kontroluje, zda byla podle příznaků protokolu TCP (TCP flags)

[7] https://monitor.shodan.io/

[8] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

ustanovena komunikace (3-way handshake). Pokud je komunikace spojitá, modul vytváří závislosti v podobě `IP1:port1-IP2`, kde `IP1` a `port1` jsou zdrojem komunikace.

V plánu roku 2021 je dokončit některé rozpracované vstupní moduly a začít pracovat na některých nových. Mezi rozpracované moduly patří např. modul přiřazující štítky dle klasifikace provozu, který využívá metody strojového učení a v plánu je modul pro získávání dat z aplikačních dat (konkrétně z HTTP komunikace se zaměřením na položky User-Agent, ale i Host a URL). Tyto moduly jsou a nebo teprve budou ve vývoji a jejich výstupy se proto mohou ještě měnit.

Kapitola 3

Externí zdroje dat využitelné pro klasifikaci síťových zařízení

Ke zjištění informací o síťových zařízeních lze použít kromě systému ADiCT i několik různých externích zdrojů, které shromažďují a získávají informace o velkém počtu zařízení z globálního prostoru IP adres. Několik těchto zdrojů lze přímo využít k anotaci datové sady, která je vytvořena v rámci této diplomové práce.

Jednou z velmi častých technik, pro zjištění informací o službách běžících na zařízení v počítačové síti, je **skenování otevřených portů** daného **zařízení** [1]. Skenovací nástroj se pokusí připojit na port zařízení a zkoumá jeho odpověď. Z odpovědi je jasné, zda je port otevřený a velmi často lze zjistit i další užitečné informace. Většina služeb má nastaven uvítací výpis pro uživatele, který se objeví po připojení uživatele ke službě (tzv. baner). Obsahem baneru jsou často informace o názvu konkrétní služby a název aplikace implementující danou službu (někdy i včetně verze aplikace). Baner může obsahovat i podrobnější informace, jako např. název operačního systému, který zařízení používá. Všechny tyto informace poskytne služba skenovacímu nástroji v podobě baneru a skenovacímu nástroji stačí pouze vyjmout užitečné informace. Celému procesu se často říká **banner grabbing** a je základem většiny skenovacích nástrojů [12].

Skenovací techniky se dělí do několika kategorií. Jedním způsobem dělení je počet zdrojových zařízení, která iniciují skenování. Některé skenovací techniky využívají pouze jedno zdrojové zařízení, další využívají zdrojů více (*distribuované skenování*). Dále jsou skenovací techniky děleny na *vertikální* a *horizontální*. Při *vertikálním skenování* dochází ke skenování jednoho nebo více portů na pouze jednom cílovém zařízení a při *horizontálním skenování* je skenován pouze jeden port na více cílových zařízeních.

Techniky skenování portů využívají vyhledávače Shodan a Censys a jsou použity v rámci této diplomové práce pro vytvoření datové sady, jejímž cílem je podpořit bezpečnostní výzkum v oblasti profilování síťových zařízení. Správci počítačových sítí mohou používat skenovací techniky k detekci otevřených portů, které by otevřeny být neměly a mohou to napravit správnou konfigurací zařízení, nebo nastavením určitých bezpečnostních metrik (např. firewall). Ne vždy jsou ale techniky skenování použity pro bezpečnostní výzkum. Velmi často jsou metody skenování portů zneužity hackery k nalezení zranitelných služeb a aplikací. Pokud se útočníkovi pomocí skenování podaří zjistit jméno a verzi konkrétní aplikace, může tuto informaci využít k nalezení existující zranitelnosti aplikace.

Proces skenování je často činnost, která předchází před provedením pokusu o kompromitaci sítě nebo zařízení právě zneužitím zranitelností. Různé práce se proto zabývají detekcí

skenovacích nástrojů. Např. v článku *Surveying Port Scans and Their Detection Methodologies* [1] je poskytnuto hierarchické dělení metod pro detekci metod skenování portů, které je rozděleno na metody detekující *skenování z jednoho zdroje* i *distribuované skenování*. Hierarchické dělení přístupů pokračuje podle přístupů využitých k detekci skenování (algoritmické, metody využívající hraniční hodnotu, metody založené na využití pravidel, vizuální, ...) a u všech přístupů je rozebráno několik relevantních prací v dané oblasti.

Tato kapitola pokračuje analýzou vyhledávačů zařízení Shodan a Censys, jejichž základem jsou právě techniky skenování portů na globální úrovni. Dále jsou rozebrány služby WhoIs, GeoIp, blacklisty, PassiveDNS a služba TOR. Informace z těchto datových zdrojů jsou přímo použity pro anotaci výsledné datové sady, která je v rámci této diplomové práce vytvořena.

3.1 Vyhledávač Shodan

Vyhledávač internetových zařízení Shodan [22, 14, 16] (dále v textu už pouze jako Shodan), vznikl v roce 2009 a za jeho vznikem stojí John Matherly. Vyhledávač Shodan detekuje všechna zařízení se směrovatelnou IP adresou (podporována je IPv4 a od roku 2015 i IPv6) a získává informace o těchto zařízeních metodou skenování portů. Mezi tato zařízení patří počítače, tiskárny, web kamery, ale i průmyslové řídící systémy. Dohromady Shodan sbírá informace o přibližně 500 miliónech zařízeních. Nasbírané informace ukládá do databáze a poskytuje je veřejně skrze rozhraní API i uživatelské rozhraní na webových stránkách.

Shodan skenuje veřejný prostor nonstop a nasbíraná data ukládá do databáze ihned. Servery Shodanu, které spouští skenování, jsou umístěny na různých místech světa (např. Čína, Francie, Česká republika), aby nebyl ovlivněn výsledek skenování různými restriktivními opatřeními některých vlád (např. někteří administrátoři v USA blokují veškeré prefixy z Číny). Shodan dříve skenoval pouze několik nejpoužívanějších portů (např. 22 – ssh, 80 – HTTP, ...), ale seznam podporovaných portů je neustále navyšován a v dnešní době dochází ke skenování několika stovek různých portů. Algoritmus každého serveru pro skenování je přímočarý. Nejprve dojde k náhodnému vygenerování IP adresy, kterou bude server skenovat a stejným způsobem je náhodně vygenerováno i číslo portu ze všech podporovaných portů, které Shodan skenuje. Následně dojde k pokusu o připojení k dané IP adrese a portu a získání baneru služby běžící na daném portu, který je poté uložen do databáze. Celý postup je ihned zopakován pro skenování dalšího zařízení. Tímto je zaručeno rovnoměrné rozložení skenování adresového prostoru. Z uložených banerů jsou poté vybrány užitečné informace, které jsou strukturovaně uloženy. Stejná struktura je poté poskytnuta uživatelům skrze webové rozhraní nebo rozhraní API.

Data o zařízení získaná z baneru jsou obohacena o data z dalších veřejně dostupných služeb, které umožňují získat informace o zařízení v síti. Mezi takové služby patří geolokační databáze, která shromažďuje informace o fyzické lokaci zařízení (vysvětleno v kapitole 3.4). Další volně dostupné informace jsou číslo autonomního systému, do kterého patří IP adresa zařízení, doménové jméno zařízení, poskytovatel internetu, který je zodpovědný za správu dané IP adresy a další.

Ve výchozím stavu webové rozhraní i rozhraní API prohledává při vyhledávání uložené banery. Pokud tedy uživatel zadá do vyhledávače např. řetězec „Apache“, vyhledávač mu najde všechna zařízení v databázi, u kterých baner služby běžící na nějakém otevřeném portu obsahuje řetězec „Apache“. Při vyhledávání zařízení vyhledávač uživateli zobrazí záznamy, které jsou staré maximálně 30 dní. Tím je zaručeno, že výsledky, které vyhledávač poskytuje, jsou aktuální. Vyhledávač umožňuje používat i pokročilejší filtrování. Základní

filtr zařízení je v podobě `filtr:hodnota`, kde `filtr` může být nahrazen např. za *port* (specifikace hledaného otevřeného portu), *country* (využití geolokačních dat k filtrování dle států), *asn* (specifikující číslo autonomního systému, v kterém se zařízení musí nacházet) a mnoho dalších[1]. U hodnot, které to podporují (např. čísla portů), lze specifikovat výčet hodnot oddělených čárkou – `port:25,80`. Tím dojde k filtraci zařízení, která mají otevřené oba porty. Zároveň lze jednotlivé filtry spojovat za použití logických spojek *AND* a *OR*, logické spojky *NOT* může být dosaženo prefixem znaku pomlčka před filtrem.

Velké množství služeb využívá kryptografický protokol SSL nebo TLS a vyhledávač Shodan se při skenování snaží zjistit používané verze těchto protokolů, včetně veřejných kryptografických certifikátů, dohodnuté metody pro šifrování komunikace a další. Tyto informace jsou dostupné pro každé zařízení včetně seznamu otevřených portů daného zařízení a vyextrahovaných informací z baneru služeb běžících na těchto otevřených portech. Číslo otevřeného portu nemusí být vždy přesným identifikátorem služby, protože někdy se snaží správci zařízení pro svoji ochranu spouštět známé služby na neobvyklých portech. Tomuto principu se často říká „security through obscurity“ (bezpečnost prostřednictvím neznáma). Tento přístup ale není příliš bezpečný, protože se spoléhá na to, že útočník nezná podrobnosti o vnitřnostech systému používajícím tuto techniku. Jenže to není dostatečné pro zabezpečení a jen v tomto případě lze důkazu dosáhnout aplikací jednoduchého filtru „`product:openssh -port:22`“ ve vyhledávači Shodan pro zobrazení všech zařízení, která používají službu SSH na nestandardním čísle portu.

Vyhledávač Shodan zobrazuje záznamy staré maximálně 1 měsíc. Zároveň vyhledávač limituje počet zobrazených záznamů při vyhledání na 10 zařízení nebo na 50 zařízení pro registrované uživatele. Zobrazení dodatečných záznamů je zpoplatněno ve formě tzv. dotazovacích kreditů. Další výhody ve formě např. neomezených dotazů skrze rozhraní API je umožněno ve formě předplatného[2].

Kromě webového rozhraní a rozhraní API poskytuje vyhledávač rozhraní příkazové řádky. Rozhraní lze nainstalovat samostatně[3] nebo je dostupné automaticky při instalaci oficiální knihovny `shodan` jazyka Python[4]. Přístup k API je sice veřejný, ale bezplatný účet obsahuje určitá omezení na počet použitých filtrů za měsíc. Dle dokumentace je také nastaven limit dotazů skrze API na 1 dotaz za 1 sekundu pro všechny typy účtů.

Shodan je robustní nástroj s obrovskou databází zařízení a informací o nich. Data jsou převážně používána pro bezpečnostní výzkum, ale mohou být i zneužita. Existuje několik prací [3, 2], které se zabývají analýzou Shodanu z pohledu průmyslových řídících systémů, které jsou v Shodanu k nalezení a často jsou součástí důležité infrastruktury (zahrnuje dopravní semafory, řídící systémy budov, ...). Obě práce zdůrazňují, že je velmi jednoduché vyhledat taková zařízení aplikací několika filtrů při vyhledávání. Často taková zařízení nejsou dostatečně zabezpečena a je pro útočníka jednoduché je ovládnout. Zároveň práce zkoumají rychlost vyhledávače z pohledu nalezení nových zařízení v globální síti (fáze skenování) a prodleva mezi indexací a zobrazením zařízení v rozhraní vyhledávače. Daného výzkumu dosahují nasazením vlastního zařízení na síť (honeypot[5] nebo jiné zařízení). Z obou článků vyplývá, že průměrná doba, kterou Shodan potřebuje pro objevení nového zařízení v globální síti, jsou přibližně 3 dny. Celková doba, kterou vyhledávači trvá, než zobrazí ske-

[1] Veškeré filtry jsou k nalezení v příloze B článku *Kompletní návod k Shodanu* [16].
[2] https://developer.shodan.io/pricing
[3] https://help.shodan.io/command-line-interface/0-installation
[4] https://pypi.org/project/shodan/
[5] https://cs.wikipedia.org/wiki/Honeypot

nované informace o novém zařízení v síti od jeho nasazení, je značně proměnná a pohybuje se přibližně od 3 do 15 dní.

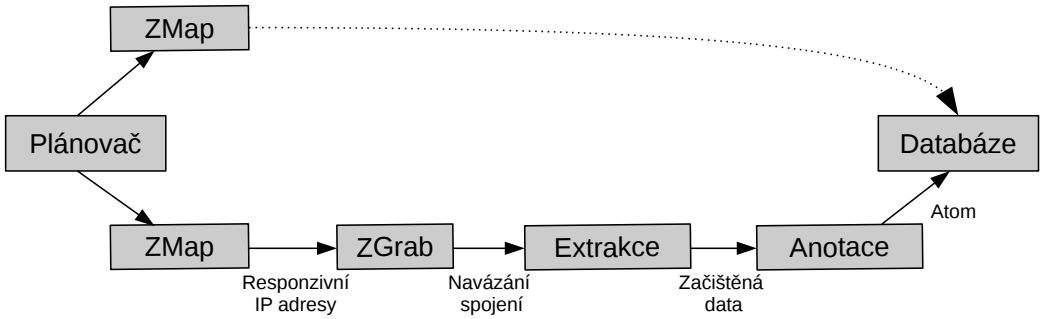
3.2 Vyhledávač Censys

Vyhledávač síťových zařízení a sítí připojených do internetu, zvaný Censys [4, 14], vznikl v roce 2015. Byl vyvinut stejnými vývojáři, kteří o dva roky dříve (2013) implementovali otevřený (open-source) nástroj ZMap [5], který umožňuje velmi rychle proskenovat globální prostor IPv4 adres. Hlavní iniciativou nástroje ZMap je zjednodušení bezpečnostního výzkumu v oblasti průzkumu internetu a analýzy nejen četnosti využívaných služeb (např. procentuální zastoupení protokolu HTTPS v internetu a metod pro šifrování, které jsou s protokolem HTTPS využity). Skenování globálního adresového prostoru bylo do té doby velmi složité a hlavně časově namáhavé. Např. při použití jednoho stroje a nástroje Nmap by skenování globálního adresového prostoru mohlo zabrat několik týdnů a i při použití více strojů by skenování trvalo příliš dlouho. Nástroj ZMap dokáže provést takto rozsáhlý sken do 45 minut z jednoho stroje, který nemusí být vybaven specializovanými komponentami, ani specializovanými kernelovými moduly.

Autoři nástroje ZMap dokazují, že pomocí jejich nástroje dokáží proskenovat globální prostor přibližně 1300 × rychleji, než nástroj Nmap při nejagresivnějším nastavení, a dosahují přitom ekvivalentní přesnosti. Tohoto velkého rozdílu v rychlosti dosahují optimalizovanou architekturou pro účel takto rozsáhlého skenování. Mezi několik optimalizací patří optimalizace sondování (nástroj Nmap adaptuje přenosovou rychlost, nástroj ZMap rychlost neomezuje a přímo sestavuje ethernetové rámce, čímž přeskakuje TCP/IP stack), odebrání nutnosti udržení stavu připojení (nástroj Nmap udržuje stav připojení, aby věděl, které zařízení již skenoval, ale nástroj ZMap stav neudržuje a spoléhá se na generátor adres pro skenování, který generuje náhodné permutace adresového prostoru) a několik dalších. Nástroj Nmap je vysoce flexibilní a je používán především pro skenování více portů na menším počtu cílových zařízení (*vertikální skenování*). Nástroj ZMap je naopak optimalizován pro skenování jednoho portu na velkém množství cílových zařízení (*horizontální skenování*). Autoři tyto dva nástroje porovnávají především z důvodu, že nástroj Nmap byl již dříve použit pro průzkum globálního adresového prostoru a snaží se tím dokázat, že pro tento účel lze vytvořit efektivnější nástroj.

Vyhledávač Censys pro vyhledávání a skenování zařízení připojených do internetu používá právě nástroj ZMap pro rychlé vyhledání nových hostů v IPv4 adresovém prostoru. Jednou z používaných technik pro detekci otevřených portů je TCP SYN sken, který je základem mnoha skenovacích nástrojů, a používá jej i ZMap. Tento typ skenu odešle jeden TCP paket s příznakem SYN a obdrží odpověď, podle které může určit, zda je port otevřen nebo ne. Port je otevřen, pokud přijde jako odpověď TCP paket s příznakem ACK a nebo je uzavřen, pokud přijde jako odpověď paket s příznakem RST. Pokud je port otevřen, je spuštěn nástroj ZGrab (součástí projektu ZMap), který se pokusí ustanovit plné spojení se službou běžící na daném portu pomocí „potřesení rukou“ (full handshake včetně kryptografického protokolu SSL/TLS). Nástroj ZGrab zpracuje získaná data z ustáleného spojení, zahrnující baner služby a veškerá data získaná při procesu ustanovení spojení (včetně podporovaného šifrování - verze protokolu SSL/TLS, veřejné kryptografické certifikáty, dohodnuté metody pro šifrování komunikace, ...) a uloží je do strukturované podoby. Data uložené ve struktuře jsou dále prozkoumána a struktura je rozšířená o nalezená a extrahovaná data, např. model nebo verze skenovaného zařízení a verze služby. Těmto datům jsou přiřazeny

reprezentující štítky a výsledná struktura je uložena do databáze. Celý proces je znázorněn na obrázku 3.1. Sken celého internetu je prováděn alespoň jednou týdně.



Obrázek 3.1: Celý průběh skenování internetu vyhledávačem Censys. Plánovač přiřadí množinu vygenerovaných IPv4 adres nástroji ZMap, který nalezne všechna responzivní zařízení. Nástroj ZGrab na těchto zařízeních provede navázání spojení (full handshake). Ze všech získaných informací při navazování spojení jsou extrahována zajímavá data, která jsou následně oštítkována. Po tomto procesu vznikne tzv. `Atom`, který reprezentuje finální datovou strukturu, která je uložena do databáze. Obrázek je převzat z článku *A Search Engine Backed by Internet-Wide Scanning* [4].

Data jsou ukládána v NoSQL databázi. Vzhledem k obrovskému množství dat, které je nutné zpracovat databází, si vývojáři museli vytvořit vlastní implementaci NoSQL databáze zvanou ZDb. Vytvořená databáze byla otestována nad stejným množstvím dat vůči nejpoužívanějším NoSQL databázím MongoDB a Cassandra. Obě databáze měly problém se zpracováním obrovského množství dat a v porovnání s databází ZDb zaostávají v rychlosti i výsledné velikosti uložených dat.

Všechna získaná data o zařízeních jsou veřejně dostupná skrze webové rozhraní, rozhraní API (např. skrze balíček programovacího jazyka Python[6]) a cloudová služba Google BigQuery[7] pro pokročilou analýzu dat. Vyhledávač umožňuje vyhledávat podle IPv4 adres, ale zároveň je možné zařízení podle webových stránek, protože Censys agreguje informace o 1 milionu zařízeních, které provozují nejpoužívanější webové stránky podle seznamu služby Alexa[8]. Poslední možností vyhledávání je podle veřejných certifikátů, které se povedly získat při skenování všech zařízení. Vyhledávač Censys je tedy velmi flexibilní v možnostech vyhledávání, které usnadňují vyhledání zkoumaného zařízení. Ve vyhledávači i v rozhraní API je možné vyhledávat použitím klíčových slov spojených znakem tečka. Tato notace je totožná s uloženou strukturou o zařízení (např. `location.country_code:US`). Filtry je možné spojovat pomocí logických spojek `AND` a `OR`, lze využít zástupné znaky, rozsahy i regulární výrazy. U každého nalezeného záznamu jsou zobrazeny všechny otevřené porty se všemi informacemi o nasazených kryptografických certifikátech a další metadata a štítky charakterizující zařízení. Informace zahrnují i geografická data a síťové směrovací informace (číslo autonomního systému, ...).

[6] https://pypi.org/project/censys/
[7] https://cloud.google.com/bigquery
[8] https://www.alexa.com/topsites

Porovnání s vyhledávačem Shodan

Autoři v článku představujícím vyhledávač Censys [4] porovnávají svůj vyhledávač s vyhledávačem Shodan, který je svou funkcionalitou nejblíže v porovnání s vyhledávačem Censys. Vyhledávač Shodan je bohužel mnohem méně transparentní ve svém fungování. Z dostupných informací není jasné, jak provádí proces získání baneru služeb a jak často dochází ke skenování celého internetu (v průvodci Shodanem [16] je zmínka o nonstop skenování, ale není zmíněno, kolik času zabere vyhledávači Shodan proskenování celého adresového prostoru). Proto se autoři pokusili porovnat výsledky skenování služeb podporovaných oběma vyhledávači (FTP, HTTP, HTTPS) nad sítěmi 3 akademických institucí. Vyhledávač Censys nalezl přibližně 2-8 × více zařízení, než vyhledávač Shodan. Výsledky byly ověřeny s univerzitními institucemi a bylo potvrzeno, že dodatečně nalezená zařízení opravdu chyběla nebo nebyla v blízké historii nalezena vyhledávačem Shodan (ve výchozím nastavení zobrazuje vyhledávač Shodan záznamy staré maximálně 1 měsíc, vyhledávač Censys zobrazuje záznamy staré maximálně 1 týden).

Zároveň došlo k porovnání rychlosti nalezení nového FTP serveru, který byl autory připojen do internetu. Vyhledávač Censys zobrazil své zařízení ve veřejném rozhraní do 48 hodin a vyhledávači Shodan to trvalo 25 dní (podobné experimenty popsané u představení Shodanu zabraly přiližně 3 - 15 dní, viz kapitola 3.1). Vyhledávač Censys zároveň poskytuje podrobnější filtraci celé struktury záznamu, vyhledávač Shodan umožňuje pouze textové vyhledávání v uložených banerech služeb zařízení s několika dalšími základními filtry, ale neumožňuje filtrovat podle struktury uložených záznamů. Zároveň Shodan limituje počet zobrazených hostů a zobrazení všech výsledků vyhledávání je tedy zpoplatněno. Zároveň je celý vyhledávač Censys veden formou otevřeného projektu (open-source). Omezená transparentnost celého systému Shodan a další zmíněné nedostatky mohou vést k přednostnímu využití vyhledávače Censys různými výzkumnými skupinami. Jedním z faktorů, v kterém je vyhledávač Shodan napřed, je podpora skenování zařízení používajících IP verze 6.

Skenování globálního adresového prostoru má veliký potenciál pro odhalení různých bezpečnostních problémů a umožňuje povrchovou analýzu zabezpečení na globálním měřítku. Autoři vyhledávače Censys jsou si vědomi i různých dopadů jejich systému a provedli různé kroky k jejich omezení. Mezi tyto kroky patří různé optimalizace skenovacích nástrojů, aby nepříznivě neovlivňovaly provoz síťových operátorů. Zároveň má každý server záznam v databázi WhoIs (systém popsán v kapitole 3.3), reverzní záznam v systému DNS a každý server provozuje webovou stránku popisující výzkumný úmysl skenování a obsahující kontaktní informace. I přes všechny kroky pro omezení nebezpečných dopadů je stále možné přístupná data zneužít k vyhledání různých zranitelností. To je jeden z důvodů výzkumu detekce skenovacích vyhledávačů Censys a Shodan [14], který popisuje způsob detekce všech fází skenování těchto vyhledávačů. Zároveň je možné kontaktovat podporu vyhledávače Censys pro odstranění určité sítě ze skenovaného rozsahu.

3.3 Služba WhoIs

Služba WhoIs [7, 15] poskytuje veřejně informace o majitelích zaregistrovaných internetových domén a IP adres. Informace uložené ve službě WhoIs o určité doméně pochází z doby její registrace, nebo je možné, aby byly v průběhu času aktualizovány registrátorem dané domény. Informace o doménách jsou uloženy u regionálních internetových registrátorů (RIR - Regional Internet Registry), které mimo jiné spravují domény nejvyšší úrovně (TLD - Top Level Domains). Pokud si chce organizace zaregistrovat vlastní doménu, musí přímo kon-

taktovat regionálního registrátora (pro Evropu je např. přidělen registrátor RIPE), popř. lze použít prostředníka pro registraci v podobě národního internetového registrátora (NIR - National Internet Registry).

Při registraci domény musí organizace, která registruje svoji novou doménu, specifikovat i kontaktní informace, které se dají použít v případě nutnosti kontaktování majitele domény např. v případě jejího zneužití. Záznamy služeb WhoIs poté většinou obsahují informace o regionálním internetovém registrátorovi, u kterého je doména registrována, a také informace o majiteli domény, který doménu registroval. Poskytovatelů služby WhoIs je více a záznamy služby WhoIs se mohou často lišit svojí strukturou, protože narozdíl od většiny internetových protokolů služba WhoIs nedefinuje pevný standard pro vytváření záznamů. Na výpisu 3.1 je ukázka záznamu služby WhoIs o IP adrese, pod kterou běží webové stránky naší fakulty www.fit.vut.cz.

```
NetRange: 147.228.0.0 - 147.237.255.255
CIDR: 147.236.0.0/15, 147.228.0.0/14, 147.232.0.0/14
Organization: RIPE Network Coordination Centre (RIPE)
OrgAbuseEmail: abuse@ripe.net

address: Brno University of Technology
address: Antoninska 1
address: 601 90 Brno
address: The Czech Republic

abuse-mailbox: abuse@vutbr.cz
route: 147.229.0.0/17
descr: VUTBR-NET1
origin: AS197451
```

Výpis 3.1: Ukázka základních informací nacházejících se v záznamu služby WhoIs o IP adrese, pod kterou běží webové stránky naší fakulty www.fit.vut.cz. Ze záznamu lze jednoduše vyčíst síťový rozsah, v kterém se adresa nachází, registrátor domény a kontakt na něj, fyzická adresa organizace vlastnící doménu, kontaktní údaje a i číslo autonomního systému. Celý záznam obsahuje podrobnější informace, zde je pouze uvedena ukázka důležitých údajů.

Uvedené informace v systému WhoIs ale nemusí být vždy správně. V technické zprávě zkoumající přesnost služby WhoIs [7] bylo při testování správnosti uvedených kontaktních údajů v záznamech WhoIs naprosto správně pouze 23 % záznamů. Ostatní záznamy obsahovaly chybu nebo byly nekompletní. I přes některé chyby se povedlo dohledat a kontaktovat vlastníka domény, ale přibližně u 50 % záznamů se nepovedlo navázat kontakt s vlastníkem domény. Příčin těchto chybných důvodů může být hned několik. Při registraci domény je vyžadováno uvést veškeré základní informace, které ale z důvodů anonymity může člověk registrující doménu úmyslně uvést chybně. Zároveň někteří lidé při registrování domény nedávají kontaktním údajům příliš velkou důležitost a některé kontaktní údaje jsou neaktuální, nebo se časem stanou neaktuálními. Mnoho těchto chyb je zapříčiněno tím, že není vyžadována kontrola identity při registraci domény.

3.4 Geolokační databáze IP adres

S přibývajícími zařízeními na internetu je stále větší poptávka po technikách geolokace [11, 21], které umožňují zjistit fyzickou polohu zařízení podle jeho aktuální IP adresy. Informace o fyzické poloze zařízení má mnoho využití, např. zobrazení lokálních událostí uživateli sociálních sítích, zobrazení lokální předpovědi počasí nebo automatický výběr jazyka při prvním zobrazení stránky. Zároveň lze tyto informace použít i pro bezpečnostní účely, mezi které patří detekce podezřelého přihlášení k určité online službě z neobvyklé lokace, platba kreditní kartou z neobvyklé lokace a mnohé další.

Techniky geolokace, které používají komerční geolokační databáze, nejsou důkladně známy. Obecně ale tyto databáze mohou používat informace z WhoIs služeb, respektive informace dostupné u regionálního internetového registrátora (RIR), jak bylo zmíněno v kapitole 3.3. Dále jsou využívány informace ze služby DNS, u kterých se spoléhá na geografické pojmenování uzlů podle konvence, která ale není žádným standardem. Dále je možné dohledat záznam typu LOC ve službě DNS, ale tento záznam není povinný (a může být i nepřesný) a proto se na něj nelze vždy plně spoléhat. Některé techniky využívají měření doby odezvy v síťové komunikaci k aproximaci vzdálenosti síťového uzlu, ale odezva je často velmi nestálá. Používají se i další zdroje jako služba GPS nebo skenování WiFi sítě nebo prohledávání dostupných informací na webových stránkách.

Existuje několik poskytovatelů geolokačních služeb, např. od organizace MaxMind, DP-IP, IP2Location a Skyhook. Podle důkladných testování [11] těchto geolokačních databází vychází, že přesnost geolokačních služeb je velmi dobrá na úrovni identifikace státu (dosahuje téměř 100 %), ale přesnost značně klesá na úrovni identifikace měst. Přesnost na úrovni měst se pohybuje okolo 30 %, ale pro testování bylo použito i hodně menších měst, celkový medián jejich populace je 17 000 obyvatel. Přesnost identifikace měst výrazně závisí právě na počtu obyvatel a především na jejich rozloze. Proto zařízení, která se nacházejí ve velkých městech, bývají mnohem přesněji lokalizována. Přesnost lokace zařízení (IP adresy) je měřena v počtu km od reálné lokace zařízení. Geolokační databáze MaxMind a SkyHook dosahují obě velmi dobrých výsledků na úrovni lokace státu a mají nejmenší medián ve vzdálenosti určení lokace od reálné polohy. Službu MaxMind také používá vyhledávač Censys pro doplnění geolokačních informací pro vyhledávané zařízení.

3.5 Seznam blokovaných identifikátorů - Blacklist

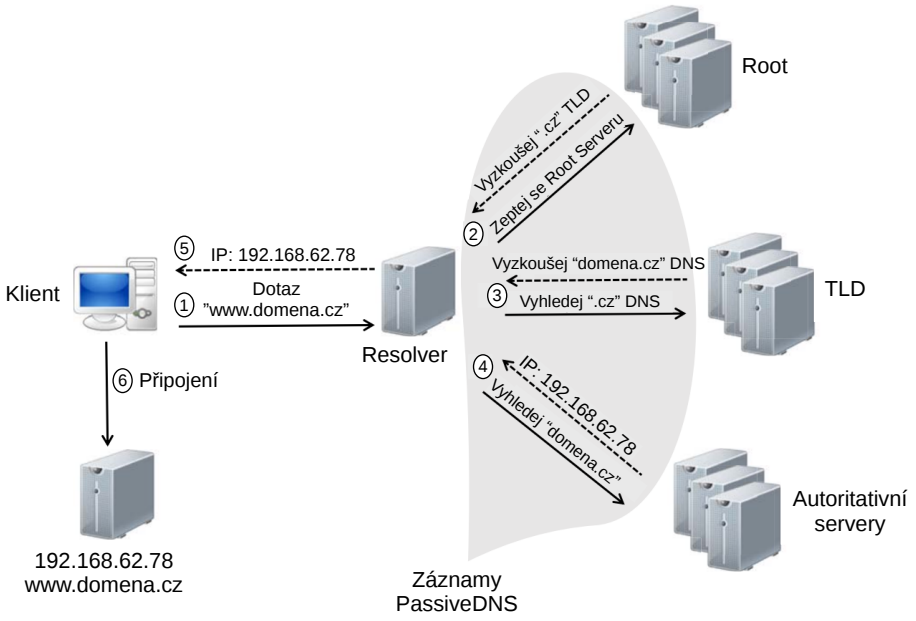
Blacklist je seznam identifikátorů, s kterými je zakázána komunikace. Těmito identifikátory jsou nejčastěji konkrétní IP adresy nebo doménové jména (případně konkrétní URL adresy) a emailové adresy. Konkrétní IP adresa se nejčastěji objeví v blacklistu, protože generuje škodlivý provoz a doména může obsahovat škodlivý obsah na svých webových stránkách. Emailové adresy jsou blokovány ze stejných škodlivých důvodů nebo rozesílají nevyžádané informace (tzv. spam). Pokud je identifikátor uveden v blacklistu a snaží se o navázání komunikace, je tato komunikace zamítnuta. Opakem blacklistu je tzv. whitelist, který funguje přesně naopak. V případě whitelistu je komunikace zamítnuta tehdy, kdy identifikátor není uveden na seznamu.

Blacklist může být veřejný i privátní. Veřejný blacklist může pomoci více správcům, kteří ho poté mohou využít pro ochranu své počítačové sítě, ale každý blacklist je nutné v průběhu času aktualizovat. Zařízení, které generovalo škodlivý provoz, nemusí být škodlivé pořád. Mohlo dojít např. k nakažení škodlivým programem, který může být po jeho detekování odstraněn. Jakmile je škodlivý program odstraněn, zařízení (identifikátor v po-

době IP adresy) by se už v blacklistu nemělo dále vyskytovat. Ke správci blacklistu se ale informace o vyčištění zařízení od škodlivého programu nedostane, pokud není přímo podán požadavek na odstranění ze seznamu z důvodu odstranění příčiny, kvůli které bylo zařízení přidáno na seznam. To je ale málo časté a proto mají záznamy v blacklistu v mnoha případech expirační dobu. Pokud po uběhnutí určitého časového úseku nevznikne další podnět na uchování identifikátoru v seznamu, dojde k jeho odstranění.

3.6 Služba PassiveDNS

Systém DNS je jednou z nejdůležitějších služeb pro provoz internetu. Hlavním cílem služby DNS je překlad doménových jmen identifikujících různé služby na konkrétní IP adresy, která identifikuje zařízení, které provozuje službu dané domény. Typické využití systému DNS je znázorněno na obrázku 3.2.



Obrázek 3.2: Průběh překladu doménového jména, které klient vyhledává, na IP adresu zařízení provozující konkrétní službu. Dotaz je předán tzv. resolveru, který se nejprve zeptá root serveru, který ho odkáže na korespondující TLD servery (Top Level Domain), které spravují a reprezentují nejvyšší vrstvu hierarchie doménových jmen (domény „.com“, „.org“, „.cz“, ...). Servery TLD už resolver odkáží přímo na autoritativní servery, které spravují konkrétní zóny systému DNS, v kterých se nacházejí záznamy o jednotlivých zařízeních provozující služby. Na obrázku je také vidět, která část provozu systému DNS bývá zachycena z důvodu replikace zón pro systém PassiveDNS. Obrázek je převzat z článku *Detecting Internet Abuse by Analyzing Passive DNS Traffic: A Survey of Implemented Systems* [23].

Systém DNS bývá často zneužíván útočníky, protože využívají otevřené struktury systému DNS, díky které mohou udržovat a ovládat škodlivé zařízení a domény skrze internet. Útočníci využívají množinu doménových jmen a IP adres pro spouštění různých útoků v podobě spamových kampaní, útoků typu phishing, DDoS (Distributed Denial of Service) a také mohou zneužívat službu DNS pro provoz Command & Control serverů. Cílem ochrany proti zneužívání služby DNS je zablokování kompromitovaných domén a hostů od

zbytku sítě a zabránit tak jejich interakci s ostatními uživateli internetu (např. pomocí vytvoření blacklistu, který bude obsahovat škodlivé domény a IP adresy). Škodlivé domény a IP adresy je možné detekovat pozorováním a analýzou aktuálního provozu služby DNS, ale tento přístup je z několika důvodů problematický. Hlavním problémem je množství provozu, který proudí skrze službu DNS a bylo by obrovskou výzvou dělat analýzu aktuálního provozu v reálném čase. Navíc by byla potřeba spolupráce mnoha různých entit po celém světě, jelikož systém DNS je globální služba. Z těchto důvodů byla vytvořena služba PassiveDNS [25, 23].

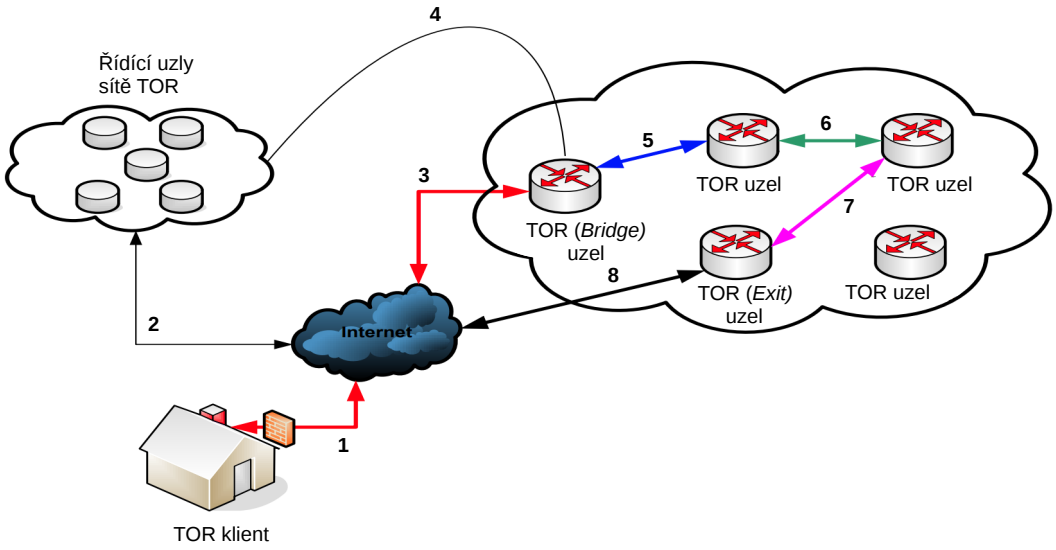
Služba Passive DNS slouží pro vytvoření co nejvíce částečných replikací zón systému DNS pomocí záchytu živého provozu systému DNS a to především záchytem odpovědí systému DNS na dotazy uživatelů. Právě odpovědi systému DNS obsahují všechny informace potřebné k vytvoření replikací zón. Konkrétně IP adresu a název dotazované služby a názvy autoritativních serverů pro danou doménu (záznamy typu NS - Name Server record). Pomocí tohoto přístupu jsou shromažďovány pouze záznamy, které jsou aktivně využívány a zároveň není nutná spolupráce s administrátory DNS zón. Na těchto uložených informacích lze poté dělat analýzu provozu služby DNS, která může odhalit různé zneužívání škodlivými entitami.

V Souhrnném článku o analýze provozu systému PassiveDNS [23] je shrnuto mnoho způsobů zneužití systému DNS. Mezi jedny z nejčastějších typů zneužívání služby DNS patří algoritmy pro generování doménových jmen (tzv. `DGA` - Domain Generation Algorithms), které umožní útočníkům vytvářet velké množství doménových jmen, pod které mohou skrývat své škodlivé služby. Některé algoritmy jsou známé a existují různé klasifikátory, které se snaží detekovat tyto náhodně generovaná doménová jména, protože legitimní služby tyto generátory ve většině případů nepoužívají. Pokud tedy nějaké zařízení používá pro své služby náhodně generované doménové jméno, může být toto zařízení podezřelé ze škodlivých aktivit. Dalším typem zneužití jsou tzv. `fast flux domains`. Útočníci využívají vlastnosti systému DNS, který umožňuje překlad jednoho doménového jména na více IP adres (původně zavedeno pro dobré účely - rozložení provozu mezi více zařízení). Útočníci tento systém používají pro zahlazení stop jejich škodlivých aktivit velmi rychlou a častou aktualizací DNS záznamů, které poté dovedou uživatele vždy na jiné zařízení. Oba tyto přístupy mají společnou vlastnost. Často aktualizují doménové záznamy již po jednom nebo několika překladech, aby zabránily omezení jejich služeb v případě odhalení. Odhalená doménová jména provozující škodlivé aktivity jsou často umístěna na blacklistech. Častá obměna doménových záznamů tento bezpečnostní přístup značně omezuje.

3.7 TOR - The Onion Router

Projekt TOR (The Onion Router) [9] je druhou verzí původní snahy o tzv. onion routing („cibulové směrování“ - smysl cibule je vysvětlen v následujícím textu a je vidět na obrázku 3.4), jehož účelem je docílit anonymní internetové komunikace a zamezení možnosti analýzy síťového provozu. Tento projekt byl v minulosti sponzorován Úřadem námořního výzkumu Spojených států amerických (ONR - Office of Naval Research) a Agenturou ministerstva obrany pro pokročilé výzkumné projekty (DARPA - Defense Advanced Research Projects Agency).

Projekt TOR se snaží docílit anonymity uživatele internetu tunelováním komunikace mezi uživatelem a cílovou stanicí skrze několik síťových uzlů, které se chovají jako proxy



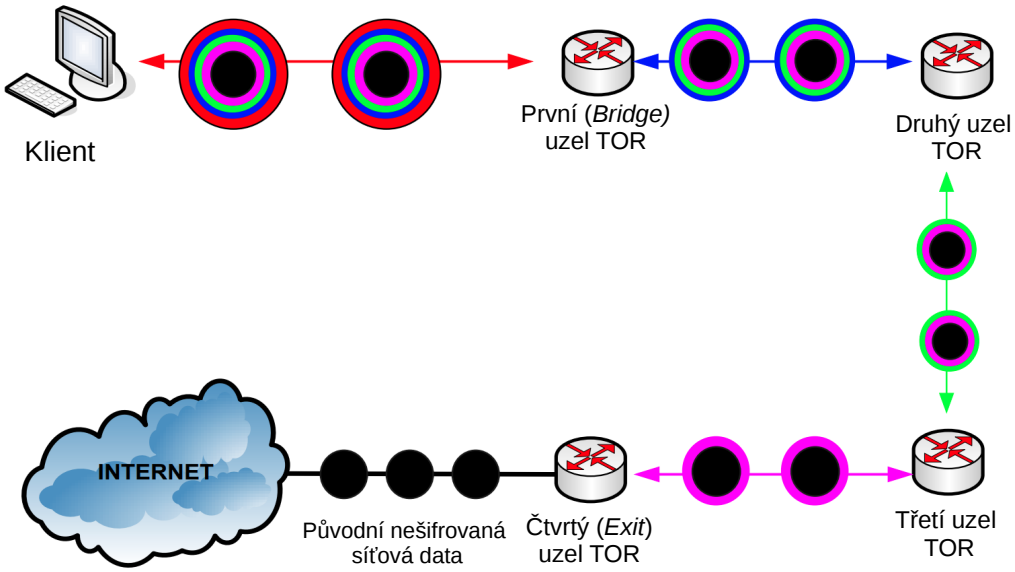
Obrázek 3.3: Postup ustanovení komunikačního okruhu klientem služby TOR. V kroku 1 a 2 se klient skrze internet připojí k řídícím uzlům, aby zjistil adresu vstupního *bridge* uzlu. V kroku 3 je navázáno spojení se vstupním *bridge* uzlem, skrze který klient v kroku 4 opět komunikuje s řídícími uzly, aby získal adresy dalších tří uzlů pro uzavření anonymizačního *okruhu*. Poté v kroku 5 naváže klient skrze *bridge* uzel komunikaci s dalším uzlem. Tento postup pokračuje v krocích 6 a 7, až nakonec poslední výstupní *exit* uzel komunikuje přímo s cílovým hostem v internetu. Obrázek je převzat z článku *The TOR Data Communication System: A Survey* [9].

server[9] (slouží jako prostředník v komunikaci a přeposílá požadavky ze zdrojového uzlu cílovému uzlu ve jménu zdrojového uzlu). Anonymizace je docíleno tím, že jednotlivé anonymizační proxy servery (uzly TOR) vzájemně neznají všechny proxy servery, přes které proudí uživatelská komunikace. Každý uzel TOR zná pouze identitu (IP adresu) předchozího a následujícího uzlu. Čím více je použito uzlů TOR po cestě komunikace mezi uživatelem a cílovou stanicí, tím větší může být potenciálně dosaženo anonymity. Každým dalším uzlem TOR v cestě se ale zvyšuje odezva v komunikaci a po mnoha testech byl stanoven počet uzlů TOR v cestě při každé komunikaci na 4 uzly.

Celá komunikace je šifrována a zároveň je použita fixní velikost paketu (498 bajtů) pro jakýkoliv typ provozu. Použitím fixní velikosti paketu se skrývá obsah paketu a je poté nemožné rozeznat typ komunikace, která proudí od uživatele a také není možné rozeznat odpověď od anonymizačních uzlů TOR.

Pro zahájení komunikace skrze anonymizační síť TOR je nutné použít klienta služby TOR, který řídí komunikaci s anonymizační sítí. Klient zařídí inicializaci spojení s anonymizační sítí nalezením vstupního uzlu TOR (zvaný ***bridge***). Seznam těchto vstupních uzlů TOR je udržován řídícími uzly a administrátory anonymizační sítě TOR. Po připojení k *bridge* uzlu klient zjistí dostupné uzly TOR dotazem na řídící uzly, které udržují seznam dostupných uzlů TOR. Z dostupných uzlů jsou náhodně zvoleny další tři uzly, které budou použity pro vytvoření tzv. *okruhu*. Seznam dostupných a zvolených uzlů TOR je přenesen ke klientovi šifrovaný, aby ani *bridge* uzel nevěděl, které uzly byly zvoleny a byl stále za-

[9] https://en.wikipedia.org/wiki/Proxy_server



Obrázek 3.4: Klient služby TOR postupně zašifruje data veřejnými klíči jednotlivých uzlů a tato data postupně jednotlivé uzly rozšifrovávají svými privátními klíči, až poslední *exit* uzel pošle do cílové destinace původní nešifrovaná data. Obdobně funguje přenos zpět, při kterém jednotlivé uzly šifrují svými privátními klíči. Obrázek je převzat z článku *The TOR Data Communication System: A Survey* [9].

chován princip, že každý uzel TOR zná jen svého předchůdce a následníka v komunikaci. Poté je navázána komunikace se zvolenými uzly TOR sítě a dojde k uzavření ***okruhu***, který je reprezentován přenosem dat skrze anonymizační síť. Celý proces je znázorněn na obrázku 3.3.

Každý uzel TOR používá svůj vlastní pár klíčů pro asymetrickou kryptografii[10], kterou využívá pro šifrování provozu a dešifrování provozu. Každý uzel tedy nad původní data přidává další vrstvu šifrování, která slouží k dodatečnému zabezpečení dat. Každá další vrstva nad původními daty může připomínat slupku a všechny tyto vrstvy tak dohromady připomínají cibuli, která má také několik vrstev. Proto bylo použito pro název projektu označení *The Onion Routing* (TOR - *„cibulové směrování“*). Celý průběh komunikace i s šifrováním jednotlivých uzlů je znázorněn na obrázku 3.4. Klient služby TOR má k dispozici veřejné klíče všech uzlů TOR v ustanoveném anonymizačním *okruhu*. Postupně klient tedy původní data (černé jádro) začne šifrovat veřejnými klíči uzlů od konce okruhu. Prvně použije veřejný klíč výstupního ***exit*** uzlu, poté třetího uzlu a pokračuje až aplikuje všechny vrstvy šifrování. Takto zašifrovaná data pošle vstupnímu *bridge* uzlu, který rozšifruje data svým privátním klíčem (odstraní vrchní vrstvu, která byla vytvořena jeho veřejným klíčem) a takto data postupují až k poslednímu *exit* uzlu, který odstraní poslední vrstvu šifrování a pošle původní data do cílové destinace. Stejným způsobem probíhá šifrování odpovědi zpět k uživateli, kdy každý server přibalí šifrování vlastním privátním klíčem a uživatel si poté data rozšifruje aplikací všech veřejných klíčů.

Vstupní *bridge* uzel je jediný uzel, který ví o identitě uživatele, protože s ním přímo komunikuje. Ostatní uzly už vůbec nevědí, odkud daná komunikace pochází. Původ komu-

[10] https://cs.wikipedia.org/wiki/Asymetrick%C3%A1_kryptografie

nikace nezná ani *exit* uzel, který už vidí původní uživatelská data, protože data byla při původním šifrování klienta osekána o všechny síťové informace vztahující se k uživatelově identitě. Všechny uzly TOR tedy pouze přeposílají provoz dále a nezajímá je, odkud daný provoz pochází (ani to nemůžou zjistit). Tím je zajištěna kompletní anonymita uživatele.

Služba TOR bývá často využívána pro škodlivé aktivity. Aktéři škodlivých aktivit se snaží skrze síť TOR získat anonymitu, aby bylo nemožné je poté zpětně vystopovat při vyšetřování. Na základě této znalosti lze usoudit, že zařízení v síti, které se účastní provozu služby TOR (je jedním z TOR uzlů), je podezřelé ze škodlivých aktivit.

Kapitola 4

Existující přístupy ke klasifikaci síťového provozu a zařízení

Tato kapitola poskytuje souhrn existujících přístupů ke klasifikaci síťového provozu a zařízení. Z existujících přístupů jsou důkladněji popsány přístupy využívající ke klasifikaci statistické chování zařízení, protože jsou vhodné pro aplikaci metod strojového učení a také se jedná o oblast s vhodným potenciálem pro další experimenty. Z těchto a dalších důvodů popsaných při jejich analýze je tento přístup zvolen v rámci této práce pro vytvoření klasifikátoru využívající vytvořenou datovou sadu síťových zařízení.

Mezi základní metody klasifikace síťového provozu patří **analýza čísel portů** a **inspekce přenášených dat uvnitř paketu**. *Článek shrnující techniky pro klasifikaci síťového provozu pomocí strojového učení* [18] uvádí několik nevýhod těchto přístupů. Analýza čísel portů použitých v provozu se spoléhá na fakt, že služba vytvářející síťovou komunikaci a používající daná čísla portů používá porty podle registrovaných čísel portů databáze společnosti IANA[1]. Hlavním nedostatkem tohoto přístupu je situace, kdy daná služba nemá oficiálně registrované číslo portu. Poté není možné správně klasifikovat službu nebo dojde ke klasifikaci chybné, pokud používá číslo portu registrované jinou aplikací. Některé služby navíc používají dynamickou alokaci portů, která také znesnadňuje identifikaci služby. Z těchto důvodů se později začaly více využívat přístupy inspekce přenášených dat uvnitř paketu, které dosahují vyšší přesnosti klasifikace, můžou ale porušovat zásady ochrany osobních údajů. Síťový provoz může být také šifrovaný a v takovém případě nelze inspekci paketů a jejich obsahu použít vůbec. Navíc postup inspekce paketů vyžaduje neustálé aktualizace databáze signatur (příznaky v podobě řetězců znaků reprezentující určitý protokol), jelikož obsah dat uvnitř paketů se může často měnit s novou verzí protokolu a také je důležité důkladně znát sémantiku klasifikovaných protokolů. Výzvou pro tento přístup ke klasifikaci je také zvládnutí souběžné inspekce dostatečného počtu toků, aby bylo možné postup využít v reálném provozu.

Souhrnný článek o profilování síťových zařízení na základě chování jejich provozu [24] definuje několik základních přístupů ke klasifikaci síťových zařízení. Prvním z nich je **přístup zkoumající jednotlivé toky**, u kterého se většinou vytváří grafy závislostí v komunikaci mezi hosty. Každé ustanovené spojení mezi síťovými hosty poté reprezentuje jednu hranu grafu. Tato technika dále zkoumá vlastnosti grafů, především hledá různé vzory a závislosti. Mezi zkoumané vlastnosti provozu patří strukturální informace odvozené z grafové

[1] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

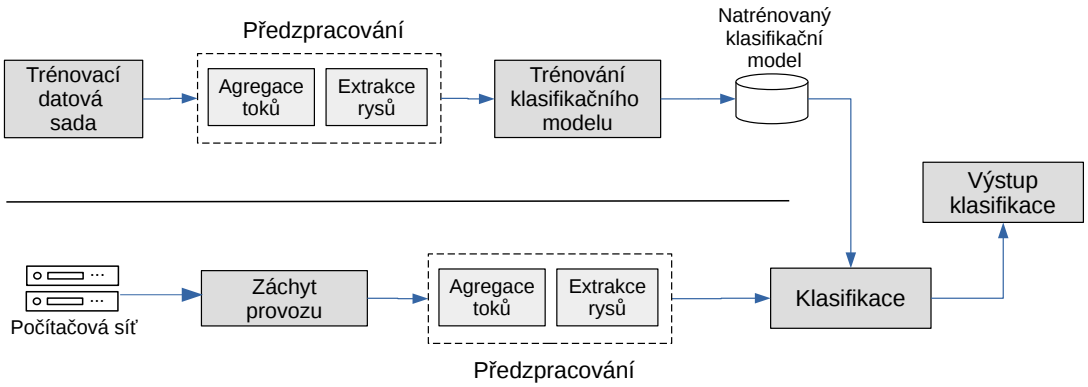
struktury (např. počet uzlů, počet hran a jejich orientace) a různé statistiky (např. rozložení stupně uzlu grafu). Další technikou profilování síťových hostů je podle autorů článku **hluboká inspekce obsahu paketů**, která se shoduje s přístupem inspekcí paketů představené dříve. Metody hluboké inspekce paketů extrahují užitečné informace přímo z přenášených dat, které jsou uvnitř paketu a často dosahují největší přesnost klasifikace z doposud představených přístupů v tomto textu. Autoři článku ale zvýrazňují stejné nevýhody, tedy náchylnost na šifrovaný provoz a přístup k soukromí přenášených dat. Výhodou předchozích technik je zachování soukromí (nepotřebují zkoumat obsahy paketů a uživatelských dat), které vyměňují za celkově nižší přesnost klasifikace oproti metodám hluboké inspekce paketů.

4.1 Klasifikace síťových zařízení pomocí statistického chování

V článku *Klasifikace provozu: Problémy a výzvy* [26] autoři představují přístupy, které mají řešit hlavní problém hluboké inspekce paketů, kterým je šifrovaný provoz. Kvůli šifrovanému provozu se začaly vyvíjet tzv. **flow-based** metody (nebo také statistické profilování), které neprovádějí inspekci jednotlivých paketů, ale zkoumají statistiky vyplývající z analýzy síťových toků. Mezi zkoumané vlastnosti patří informace o objemu provozu, velikost paketů, časová prodleva mezi pakety, příznaky specifických protokolů (např. počet RST paketů v případě použití protokolu TCP) a mnohé další. V souhrnném seznamu *Diskriminátoru použitelných pro flow-based klasifikaci* [17] si lze všimnout, že množství a variabilita zkoumaných vlastností síťových toků je veliká. Předpokladem těchto metod je, že toky vygenerované různými protokoly mají unikátní statistické vlastnosti, díky kterým lze vždy jednoznačně klasifikovat protokol. Vzhledem k stále se zvyšující složitosti aplikací se ale postupně objevují sofistikované aplikace, které ve své komunikaci používají několik protokolů najednou. Inspekcí toků je tedy možné identifikovat jednotlivé protokoly, ale je obtížné klasifikovat sofistikované aplikace.

Vzhledem k nedostatkům předchozích přístupů byl vědci představen další přístup ke klasifikaci síťového provozu, kterým jsou tzv. **host-based** techniky [26], které agregují souhrnné statistiky o provozu jednoho konkrétního hosta (zařízení). Jedná se také o statistický přístup, který ale využívá statistiky s menší granularitou. Sbírané statistiky jsou agregovány z informací o jednotlivých tocích za určitý časový úsek (např. za 1 hodinu). Mezi sbírané statistiky patří obecné informace o provozu daného zařízení, mezi které patří počet přenesených bytů, počet přenesených paketů (celkový počet, průměrný počet v síťovém toku, ...), průměrná doba trvání síťového toku a mnoho dalších. Obecný pohled na proces host-based klasifikace je vidět na obrázku 4.1.

V bakalářské práci *Behavior-based Identification of Similar Hosts in Computer Network* jsou jako souhrnné statistiky kromě základních agregací použity počty unikátních komunikujících hostů se zkoumaným zařízením a dále také celkové počty unikátních hodnot určité proměnné, např. komunikujících portů a nebo IP adres. Všechny tyto statistiky jsou vypočteny zvlášť pro příchozí a odchozí provoz, protože rozdíly mezi příchozím a odchozím provozem mohou poskytnout zajímavý pohled na profil síťového zařízení. Cílem práce bylo pomocí těchto statistik odhalit podobná zařízení v počítačové síti. Podobná zařízení často používají podobné nebo stejné služby a mělo by tedy být možné použít stejné rysy k přímé klasifikaci typů zařízení.



Obrázek 4.1: Přehled postupu klasifikace tzv. host-based statistických metod. Horní část obrázku znázorňuje fázi trénování klasifikačního modelu a dolní část obrázku znázorňuje postup klasifikace v reálném čase.

V další práci zabývající se profilováním síťových zařízení [10] jsou představeny, kromě základních rysů, různé poměry údajů získaných z agregací toků. Jedním z důležitých poměrů je poměr počtu zdrojových portů ku počtu komunikujících IP adres s daným hostem. Pokud zkoumané zařízení slouží jako server, přijímá komunikaci opakovaně na jednom portu a zároveň komunikuje se spoustou klientů. Naopak klientské zařízení používá obvykle pro navázání komunikace pokaždé odlišný port, takže počet použitých unikátních portů bude vysoký a počet cílových IP adres může být nižší. Stejně tak významnou informací je poměr počtu cílových portů ku počtu komunikujících IP adres. Velké výsledné číslo může reprezentovat skenování více portů hostem na jedné nebo pouze několika IP adresách a naopak malé číslo může reprezentovat skenování jednoho portu na mnoha IP adresách. Důležitým rysem je také distribuce mezi používanými porty. Pokud bychom uvažovali pouze top N nejpoužívanějších portů, ztratíme informaci o tom, jaký byl rozdíl v používání jednotlivých portů. Např. pokud zařízení používá port 80 a další porty, může být z pohledu klasifikace zařízení velmi důležitá informace, že v komunikaci převažuje právě port 80. Pokud bychom neměli distribuci používání jednotlivých portů, mohli bychom přijít o informaci, že je dané zařízení velmi pravděpodobně webovým serverem.

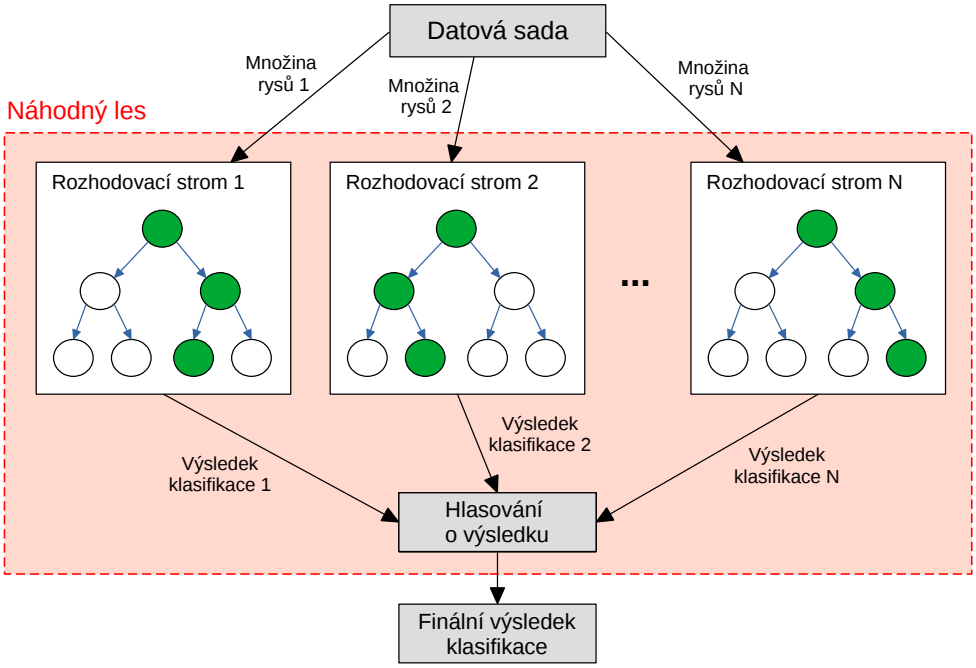
Výhodou statistického přístupu zkoumající jednotlivé hosty (host-based) je jeho vysoká škálovatelnost a adaptovatelnost na více typů provozu. Většinou tyto metody využívají strojového učení a tento přístup je dobře udržovatelný, protože stačí pouze jednou za čas přetrénovat model z aktuálnějších dat, pokud se statistické vlastnosti některých typů zařízení postupně mění. Proměnlivost statistických vlastností je ale obecně výrazně menší než např. proměnlivost signatur, které používají metody hluboké inspekce paketů.

Existují i určité nevýhody tohoto přístupu [26], mezi které patří především šum více běžících služeb (zařízení) na daném zařízení. Pokud na zkoumaném zařízení běží pouze jedna služba (aplikace), její statistické chování je dobře identifikovatelné. Čím více služeb ale na zkoumaném zařízení běží, tím více se statistické vlastnosti jedné služby ztrácí ve společné agregaci informací o provozu, který generují i ostatní služby. Další nevýhodou je problematická klasifikace jednotlivých aplikací. Ze statistických údajů lze dobře zjistit typ aplikace, ale je obtížné rozlišovat aplikace stejného typu mezi sebou. Poslední obtíže způsobuje systém pro překlad síťových adres NAT, protože statistiky jsou závislé na znalosti příchozího a odchozího provozu daného zařízení. Tento problém by s vyšší adaptací IPv6 měl být spíše redukován.

4.2 Algoritmus strojového učení Náhodný les

Klasifikace zařízení pomocí statistického chování využívá ve většině případů metod strojového učení. Před začátkem trénování klasifikačního modelu strojového učení je ale nutné zvolit algoritmus strojového učení, který bude použit pro vytvoření modelu. V souhrnném porovnání [6] sedmnácti rodin klasifikátorů (celkově 179 implementací klasifikátorů) na 121 datových sadách se klasifikátory náhodných lesů (tzv. Random forest klasifikátory) řadí mezi klasifikátory s největší přesností. Kromě vysoké přesnosti klasifikace mají ale i další výhody, mezi které patří efektivní běh na velkých datových sadách. Jsou méně náchylné na přetrénování (tzv. over-fitting) oproti klasickým rozhodovacím stromům a pokud není počet stromů uvnitř lesa příliš vysoký, lze tento algoritmus použít i pro predikce v reálném čase. Hlavním cílem této práce je automatizovaná anotace datové sady a strojové učení slouží jen pro evaluaci vytvořené datové sady. Proto nejsou zkoumány další přístupy ke klasifikaci a to i vzhledem k dobrým předpokladům algoritmu Náhodné lesy.

Algoritmus Náhodný les [27] je založen na použití mnoha odlišných obyčejných rozhodovacích stromů. Obyčejný rozhodovací strom se na základě vstupních dat snaží vytvořit strukturu stromu, kde každý nelistový uzel představuje jedno rozhodovací pravidlo, které na základně vstupních hodnot posune rozhodovací proces stromem níže k dalšímu pravidlu, až se rozhodovací proces dostane do listového uzlu, který představuje výsledek klasifikace. Velmi důležitou součástí procesu sestavení stromu je nalezení atributů, které mají největší rozlišovací schopnosti vzhledem ke klasifikačnímu problému. Atributy s největší rozhodovací schopností musejí být umístěny co nejblíže kořenu, čímž se může dosáhnout výrazného zmenšení velikosti stromu.



Obrázek 4.2: Postup klasifikace algoritmem strojového učení Náhodný les. Jednotlivé rozhodovací stromy vytvoří samostatné výsledky klasifikace, které jsou poté použity pro hlasování o finálním výsledku.

Vytváření obyčejných rozhodovacích stromů je v náhodném lese trochu odlišné od běžného postupu. První technika využívaná pro vytváření stromu se nazývá *bagging*, která ve vstupních datech z datové sady provede nahrazení některých hodnot takovými hodnotami, které se již nacházejí ve vstupní posloupnosti hodnot, některé hodnoty jsou tak náhodně duplikovány. Díky této metodě má na vstupu každý rozhodovací strom náhodného lesa odlišnou množinu dat, čímž je předcházeno přetrénování klasifikátoru (overfittingu). Druhou technikou je *náhodná volba vstupních rysů* z celé množiny rysů klasifikátoru, které jsou předány rozhodovacímu stromu. Každý strom tak pro rozhodování používá jinou podmnožinu rysů. Díky odlišným množinám rysů, které používají jednotlivé rozhodovací stromy, je zaručeno, že stromy mezi sebou nejsou příliš korelované a je tak dosaženo lepších výsledků.

Náhodný les je tedy složen z mnoha rozhodovacích stromů, které používají odlišné rysy pro rozhodování a zároveň jsou trénovány na odlišných vstupních datech. Jakmile je model natrénován a jsou mu předložena vstupní data, jednotlivé stromy vytvoří vlastní výsledek klasifikace, který poté předají hlasovací strategii. Ta obvykle zvolí takový finální výsledek, který se objevil v jednotlivých hlasech rozhodovacích stromů nejvícekrát. Celý proces je znázorněn na obrázku 4.2.

Kapitola 5

Návrh anotace síťových zařízení v datové sadě

Pro vytvoření datové sady je jednou z velmi důležitých činností anotace (oštítkování) všech záznamů, které se nachází v datové sadě. Proces anotace není vždy přímočarý a je důležité, aby byla data správně anotována. Chybná anotace by mohla způsobit vytvoření špatně natrénovaných modelů, které by poté nebyly použitelné v praxi nebo by mohlo dojít k chybným klasifikacím, pokud by došlo k jejich nasazení v praxi (v případě neodhalení chybné anotace).

Tato kapitola shrnuje postup přiřazovaní štítků ke všem síťovým zařízením, které uživatel zadá na vstupu automatizovaného nástroje, který má za úkol provést anotaci. Běžným výstupem anotace datové sady je seznam štítků charakterizujících zkoumanou entitu (např. formát CSV). V tomto případě je ale použito několik různých datových zdrojů a každý z nich přispívá jiným způsobem, který je vhodné od sebe oddělit. Proto je jako výsledný formát výstupu anotace zvolen formát JSON, který je dobře strojově zpracovatelný a má také dobrou lidskou čitelnost. V následujícím textu je popsán způsob využití jednotlivých datových zdrojů pro účely anotace a ukázka možného výsledku anotace využitím datového zdroje je vždy zobrazena ve formátu JSON. Spojením jednotlivých ukázek výstupů anotace vznikne finální struktura štítků, které jsou přiřazeny každému zařízení.

5.1 Typ zařízení a podporované služby

Typ zařízení je definován službami, které běží na daném zařízení. Jednotlivé služby jsou na zařízení přístupné přes síťové porty, které jsou využity při komunikaci se zařízením a určují službu, s kterou má být komunikace navázána. Pokud chce uživatel zpřístupnit určitou službu na zařízení, musí otevřít síťový port, na kterém bude služba naslouchat. K tomuto procesu většinou dojde automaticky při spuštění služby a většina služeb využívá vlastní registrované číslo portu, které je registrováno organizací IANA.

Podle všech portů, na kterých zařízení naslouchá a čeká na příchozí komunikaci, lze určit i seznam služeb, které jsou velmi pravděpodobně na daném zařízení spuštěny. Nemusí to platit úplně vždy, protože již dříve bylo zmíněno, že některá jiná služba se může „schovávat“ za číslo portu, které používá jiná registrovaná služba. Tento přístup je ale poměrně málo častý a navíc bývá v mnoha případech odhalen při procesu skenování portů a následné analýze baneru služby.

Zdroje dat ADiCT, Shodan i Censys poskytují pro každé dostupné zařízení výčet otevřených portů, který lze dále převést na názvy služeb, které jsou registrované pod danými čísly portů pomocí veřejného registru. Konkrétní název služby je užitečný, ale při trénování klasifikátoru je často kladen důraz na určitou množinu služeb, např. je úkolem natrénovat klasifikátor klasifikující webové služby. Proto lze služby rozdělit do skupin podle jejich účelu a zařízení tak přiřadit obecnější štítek, který reprezentuje typ zařízení. Např. služby běžící na portech 80, 443, 8080, 8443 (protokoly HTTP a HTTPS) lze celkově označit štítkem `web server`. Více příkladů typů zařízení získatelných z běžících služeb na zařízení je uvedeno v tabulce 5.1. Všechny tyto obecné štítky jsou aktivně využívány v systému ADiCT modulem `Service labels` a převodní tabulka byla navržena Josefem Koumarem v rámci jeho bakalářské práce [13].

Typ zařízení (štítek)	**Zahrnuté služby (čísla portů)**	**Popis**
`web server`	HTTP (80, 8080), HTTPS (443, 8443)	Webové služby
`file server`	FTP (20, 21), TFTP (69), NFS (111), ...	Služby poskytující síťový přístup k souborům
`authentication server`	Kerberos (88, 752, 753, ...), Radius (1645, 1646, ...), ...	Služby poskytující autentizaci
`database server`	MySQL (3306), PostgreSQL (5432), ...	Databázové servery
`router`	BGP (179), RIP (520), ...	Služby, které používají routery (např. výměna směrovacích informací)
`printer`	NPP (92), IPP (631), ...	Služby, které používají tiskárny

Tabulka 5.1: Část seznamu z převodové tabulky převádějící služby na typ zařízení. Typ zařízení je štítek přiřazený síťovému zařízení a zahrnuté služby ukazují čísla portů služeb, které spadají pod daný štítek.

Převodní soubor již obsahuje velké množství služeb, které lze takto agregovat do skupin definujících typ zařízení a je celkově dobře zpracovaný. Při zkoumání převodního souboru bylo odhaleno několik malých nedostatků, které byly upraveny. Při analýze nástrojů ADiCT, Censys a Shodan jsem vytvořil seznam všech otevřených portů u zařízení z malé podsítě (rozsah o velikosti CIDR notace /23) z dostupných dat, které tyto nástroje poskytují. Tento seznam byl porovnán se seznamem portů, které jsou převáděny pomocí převodní tabulky. Takto došlo k získání výčtu portů, které nejsou převeditelné převodní tabulkou a byly otevřeny na zkoumaných zařízeních. Ve výčtu portů se nacházely porty, pod kterými jsou

registrované služby, které nelze zařadit do obecnější kategorie. Zároveň se zde ale nacházely takové porty, které šlo zařadit do určitých kategorií (typů zařízení), ale nebyly v převodním souboru doposud uvažovány.

Do převodní tabulky bylo takto přidáno několik služeb patřících do kategorie `IoT` (např. služba VSCP běžící na portu 9598, která se zabývá automatizací úkolů v budovách a chytrých domácnostech), `Windows` a `Mac OS`. Další úpravy vyžadují některým službám přiřazení více štítků najednou. Původně měla každá služba přiřazen vždy pouze jeden štítek, ale k některým službám patří štítků více. Např. službě Kazaa, sloužící ke sdílení souborů operačním systémem Windows (port 1214), byl přiřazen pouze štítek `file server`, ale vzhledem k využívání služby operačním systémem Windows by měl být přiřazen i štítek `Windows`.

Do převodní tabulky byly také přidány některé služby využívané zařízeními od výrobce značky Apple, např. služba Digital Audio Access Protocol (iTunes) běžící na portu 3689. Všem těmto službám je přiřazen štítek `Apple`, pokud jsou služby využívány výhradně operačním systémem Mac OS, je přiřazen i štítek `Mac OS`. Seznam využívaných služeb zařízeními od výrobce Apple je uveden přímo na stránkách podpory výrobce[1] (je ale potřeba vyfiltrovat služby, které jsou používány pouze zařízenimi Apple). Podobný seznam lze nalézt i pro operační systém Windows[2].

Mezi již existující štítky byly dále přidány štítky `ICS` (Industrial Control System) a `message_broker`. Pod štítek `ICS` spadají např. služby Modbus a BACnet, pod štítek `message_broker`[3] patří služby využívané pro distribuci zpráv ve škálovatelných systémech. Mezi tyto služby patří např. Apache Kafka a RabbitMQ. Celkově je možné na základě překladové tabulky přiřadit zařízení 40 různých štítků, jejichž celý seznam je uveden v příloze B.

Sjednocení informací z nástrojů ADiCT, Shodan a Censys

Informace o otevřených portech získané pomocí nástrojů ADiCT, Shodan a Censys jsou často velmi podobné, někdy se ale liší. Je tedy nutné navrhnout postup, podle kterého bude docházet ke sjednocení informací v případě, kdy si nástroje rozporují svými informacemi, aby byly konkrétnímu zařízení přiřazeny finální štítky a informace týkající se typu zařízení. Celý proces sjednocení informací a určení finálních štítků je popsán v následujícím textu a zároveň je znázorněn na obrázku 5.1.

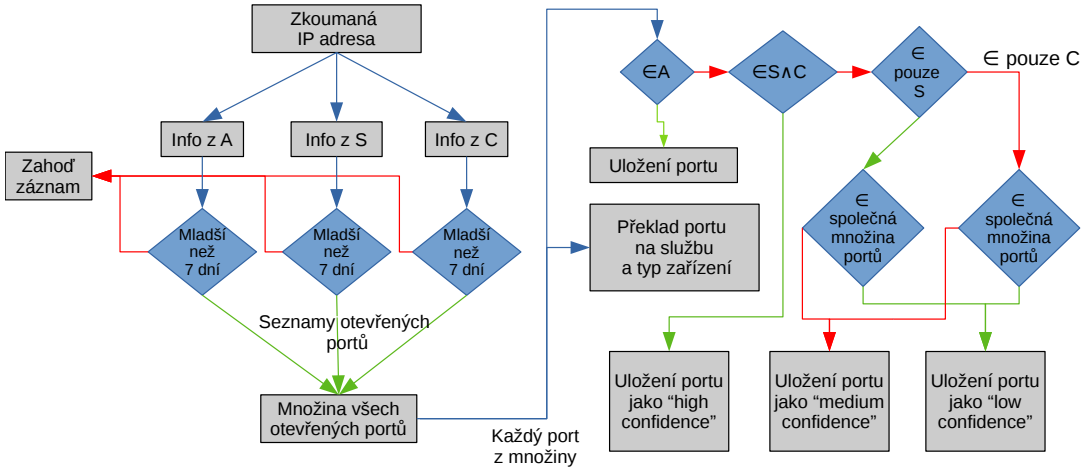
Při získávání informací z uvedených služeb se nejprve zkontroluje, zda nejsou informace starší než 7 dní (7 dní je výchozí nastavení pro atribut `open_ports` systému ADiCT, než se stane neaktuálním a i vyhledávač Censys minimálně jednou za 7 dní skenuje celý adresový prostor). Pokud jsou informace ze služby starší než 7 dní, neúčastní se následujícího postupu. Dále dojde ke sjednocení seznamů všech otevřených portů, které jsou přeloženy na službu a typ zařízení, pokud je takový port nalezen v překladové tabulce B.1 (příloha B). Poté je zkoumán každý port zvlášť, podle několika následujících pravidel.

Ze všech nástrojů má největší váhu z pohledu výsledné datové sady nástroj ADiCT. Seznam otevřených portů v atributu `open_ports` je tvořen pouze z takových portů, které byly v určitém časovém okamžiku použity v síťovém provozu. To znamená, že uvedený port musel být určitě na zařízení otevřený a byl použit i pro komunikaci. Výsledná datová sada bude kromě přiřazených štítků obsahovat zachycený síťový provoz pro zkoumaná zařízení.

[1] https://support.apple.com/cs-cz/HT202944

[2] https://docs.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements

[3] https://en.wikipedia.org/wiki/Message_broker



Obrázek 5.1: Způsob přidělení štítků o dostupných službách na zařízení a typu zařízení. Znak **A** reprezentuje službu **A**DiCT, znak **S** reprezentuje službu **S**hodan a znak **C** reprezentuje službu **C**ensys. Pokud je určitá podmínka (modré kosočtverce) splněna, proces pokračuje po zelené šipce. Pokud podmínka splněna není, pokračuje se po šipce červené. Nejprve dojde ke kontrole, zda jsou informace ze systémů ADiCT, Censys a Shodan aktuální. Pokud ano, seznamy portů se sjednotí a zkoumá se, v jakých službách se daný port vyskytuje. Společná množina portů je rovna sjednocení portů, které skenují na zařízeních zároveň obě služby Shodan i Censys.

Pokud je otevřený port uveden v systému ADiCT, je tedy vysoká pravděpodobnost, že se komunikace skrze daný port bude vyskytovat i v zachyceném provozu (je-li provoz daného zařízení sledován systémem ADiCT). Pokud se daný port, uvedený ve sjednoceném seznamu portů, vyskytuje v systému ADiCT pro dané zařízení, je dané číslo portu uloženo do finální struktury anotace do seznamu portů pod klíč `open_ports`.

V případě, kdy se port nevyskytuje ve službě ADiCT, jsou kontrolovány informace z nástrojů skenující globální prostor IP adres. Všechny takové porty jsou uloženy pod klíčem `open_ports_scanners`. Pokud se port vyskytuje v obou službách Shodan i Censys, je vysoká pravděpodobnost, že je daný port i v daném okamžiku otevřený a je uložen se štítkem `high_confidence`. Vyskytuje-li se daný port pouze v informacích poskytnutých vyhledávačem Shodan, je potřeba zkontrolovat, zda je dané číslo portu skenováno oboumi vyhledávači. Pokud ne, je port uložen se štítkem `medium_confidence`, ale pokud ano, dochází k rozporu získaných informací, protože každý vyhledávač o daném portu tvrdí jinou informaci. Z tohoto důvodu je daný port uložen jako `low_confidence`. Stejný postup platí, pokud se číslo portu nachází pouze v informacích z vyhledávače Censys. Výsledná podoba anotace otevřených portů je zobrazena ve výpisu 5.1.

V případě rozporu obou vyhledávačů by bylo ideální porovnat čas skenování daného portu. V případě, že port patří do společné množiny portů, rozhodující slovo by měla časová známka posledního provedeného skenu. Pokud by byl daný port otevřen pouze podle služby Shodan a mladší záznam o provedení skenu by byl u vyhledávače Shodan, velmi pravděpodobně to znamená, že ve chvíli, kdy prováděla sken služba Censys, byl daný port zavřený. Někdy později došlo ale k otevření daného portu a to detekovala služba Shodan, která provedla sken později než služba Censys. Služba Shodan obsahuje časovou informaci o provedení skenu u každého čísla portu (pokud je daný port otevřen), u informací ze služby Censys je bohužel tato informace nestálá a je uvedena pouze u několika portů. Podle

odpovědi z uživatelské podpory služby Censys jsem se dozvěděl, že to je způsobeno tím, že každý modul pro skenování určitého portu implementoval jiný vývojář a bohužel se tak mohou mezi datovými strukturami jednotlivých portů vyskytovat nekonzistence. Proto je mnohem složitější extrahovat informace z odpovědi aplikačního rozhraní služby Censys, než z odpovědi aplikačního rozhraní služby Shodan, která používá jednotnou datovou strukturu pro každý port.

```
'device_type': ["web server", "file server"],
'open_ports': [80, 443, 21],
'open_ports_scanners': {'22': "medium_confidence"}
'services': ["HTTP", "HTTPS", "SSH", "FTP"]
```

Výpis 5.1: Ukázka struktury výsledné anotace získané ze seznamu otevřených portů služeb ADiCT, Shodan a Censys.

5.2 Otevřené aplikace a název operačního systému

Služby Shodan i Censys jsou schopné získat z baneru běžících služeb na otevřených portech **název aplikace** a její **verzi**, pokud se taková informace v baneru nachází. Název aplikace je možné často zjistit už podle názvu poskytované služby zařízením, protože některé služby jsou implementované pouze jednou aplikací. Některé služby jsou ale implementovány více aplikacemi a je tedy důležité umět mezi nimi rozlišovat. Příkladem takové služby mohou být aplikace Apache a Nginx, které obě slouží pro provoz webového serveru a obě běží na portech 80 a 443. Služba Shodan poskytuje informace o aplikaci pod klíči `name` a `version` zanořené ve slovníkové struktuře u každého portu. Pokud se podaří extrahovat informace o aplikaci z baneru, jsou tyto informace dostupné pod těmito klíči. Služba Censys uchovává stejné informace pod klíči `product` a `version`, někdy je část názvu uvedena i pod klíčem `manufacturer`. Tyto informace jsou ale u služby Censys různě zanořeny ve slovníkové struktuře dat o otevřeném portu, která se liší dle služby běžící na daném portu[4]. I přesto lze celkem jednoduše tyto informace dohledat, pokud je služba Censys dokázala extrahovat z baneru.

Obě služby byly testovány na síti o rozsahu CIDR notace /23. Pokud na zařízení běží známé služby na nejčastěji používaných portech, např. služby webových serverů a SSH na portech 80, 443 a 22, jsou obě služby úspěšné ve správném extrahování názvu aplikace i její verze. Pro obě služby (především pro službu Censys) je ale problematické správně extrahovat název aplikace a její verzi v případě o něco méně obvyklých portů. Takovými službami jsou např. služby `rsyncd` běžící na portu 873 a `Qpopper` bežící na portu 995. Obě tyto aplikace úspěšně extrahoval vyhledávač Shodan, ale vyhledávač Censys nikoliv. Oba vyhledávače mají v extrahování názvu aplikace a její verze určité limity. V celé zkoumané podsíti se ale vždy alespoň jednomu vyhledávači povedlo úspěšně extrahovat název aplikace a její verzi, pokud jsou tyto informace uvedeny v baneru služby. Jediný problém nastal, pokud se některé známé aplikace vyskytují na neobvyklých portech (např. aplikace OpenSSH na portu 4200). V takovém případě bývají neúspěšní oba vyhledávači. U každého z nich lze ale uložit získaný baner služby běžící na daném portu a provést dodatečný pokus o extrakci známých aplikací a jejich verzí pomocí sady regulárních výrazů.

[4] https://censys.io/ipv4/help/definitions

Z pohledu **extrakce názvu operačního systému** je dobrý vyhledávač Censys, kterému se daří vždy extrahovat název operačního systému, pokud je uveden v baneru poskytované služby. Vyhledávač Shodan je v této disciplíně v celku neúspěšný, protože informace o operačním systému není nikdy uvedena přímo v atributu pojmenovaném `os`. Někdy je tato informace uvedena u konkrétního portu, ale často je ve špatném formátu (např. „(Ubuntu)“ - závorky navíc). Z pohledu extrahování názvu operačního systému je tedy vhodnější používat data poskytovaná vyhledávačem Censys, informace z vyhledávače Shodan jsou vhodné spíše pro sekundární doplnění. Vyhledávač Shodan navíc poskytuje seznam otevřených zranitelností daného zařízení, který je také přidán do finální struktury anotace. Ukázka dat je ve výpisu 5.2.

```
'applications': [
    {'name': "OpenSSH", 'version': "7.9"},
    {'name': "Apache httpd", 'version': None}
],
'os': "Ubuntu",
'vulnerabilities': ["CVE-2017-9798"]
```

Výpis 5.2: Ukázka struktury výsledné anotace aplikací a operačního systému získané ze služeb Shodan a Censys.

5.3 Geografická data a informace ze systému WhoIs

Geografická data nám umožňují klasifikovat síťová zařízení podle jejich lokace. Tato informace může být zajímavá pro správce velké sítě, jejíž zařízení nejsou soustředěna na jednom místě, ale po větší geografické lokaci. Z geografické databáze lze získat název země a města, v kterém se dané zařízení nachází. Tyto informace jsou většinou uvedeny i se souřadnicemi zeměpisné délky a šířky. Geografické informace nelze přímo využít k přiřazování konkrétních štítků (pokud nejsou použity kódy země, např. „CZ“), ale jsou z hlediska datové sady o síťových zařízeních určitě zajímavé a mohou být využity pro dodatečný výzkum (např. článek [11]).

Pro získání dat z geolokační databáze bude využita volně dostupná služba **GeoLite2**[5], kterou poskytuje zdarma provozovatel geolokačních služeb, organizace MaxMind[6]. Z této služby je možné získat potřebné informace o zařízení, tedy název země a města, v kterém se zařízení nachází, a korespondující souřadnice zeměpisné šířky a délky. Službu GeoLite2 také využívá vyhledávač Censys pro obohacení informací o zařízeních. Vyhledávač Shodan také poskytuje geolokační informace, ale není přímo jasné, jestli daná data získává vlastními metodami, nebo používá externí službu, jako vyhledávač Censys.

Data získaná ze systému **WhoIs** také nejsou přímo využitelná pro přiřazování konkrétních štítků, ale opět mohou v datové sadě najít určité využití. Pro zisk dat ze systému WhoIs jsou využity oficiální záznamy od registrátora doménových jmen a adresového prostoru RIPE NCC (RIPE Network Coordination Centre), který je správcem Evropského regionu. Z dostupných informací je do datové sady o každé IP adrese uloženo číslo autonomního systému (`asn`), do kterého konkrétní IP adresa patří, rozsah sítě autonomního systému v CIDR notaci (`asn_cidr`), dostupný popis autonomního systému (`asn_description`) a seznam en-

[5] https://dev.maxmind.com/geoip/geoip2/geolite2/
[6] https://www.maxmind.com/en/home

tit, které jsou zodpovědné za správu tohoto autonomního systému (`entities`). Dále je také uchována adresa organizace, která danou IP adresu (resp. doménu) spravuje, a kontaktní informace pro případ jejího zneužití. Celková struktura je zobrazena ve výpisu 5.3.

```
geoip: {'country': "CZ", 'city': "Vsetin", 'timezone': "Europe/Prague",
       'longitude': 2, 'latitude': 42},
whois: {
    'asn': 197451,
    'asn_cidr': "147.229.0.0/17",
    'asn_description': "VUTBR-AS, CZ",
    'entities': ["CA6319-RIPE"', "VUTBR-MNT", "AR21405-RIPE"],
    'address': "Brno University of Technology ...",
    'abuse_email': "abuse@vutbr.cz"
}
```

Výpis 5.3: Ukázka struktury výsledné anotace získané ze služeb GeoIP a WhoIs.

5.4 Anotace zařízení pomocí služby PassiveDNS

Pro získání informací ze systému PassiveDNS využitelných pro přidělení štítků zařízením je využita služba PassiveDNS sdružení Cesnet[7]. Tato služba umožňuje pro konkrétní IP adresu (zařízení) získat název domény, která byla detekována při dotazu DNS na dané zařízení a také počet, kolikrát byla určitým zdrojem dotazována. U každého záznamu je také uveden přesný čas, kdy byla rezoluce službou DNS provedena a jaká byla hodnota TTL u DNS záznamu, která určuje dobu uchování záznamu v cache paměti lokálního DNS serveru. Existujících implementací systémů PassiveDNS je více (např. implementace organizace CIRCL[8]), pro účely anotace datové sady je ale služba PassiveDNS sdružení Cesnet plně vyhovující a navíc obsahuje záznamy o zařízeních v síti sdružení Cesnet, v které bude zachycen provoz síťových zařízení ve vytvořené datové sadě.

Lze tedy zjistit seznam všech domén, které byly aktivně dotázány a nalezeny na daném zařízení. U každé domény lze provést test na její legitimitu. Pokud je název domény vygenerován náhodně, daný název nese často určité znaky, které pomohou odhalení jejího náhodného generování. Existuje několik různých implementací klasifikátorů, které se snaží klasifikovat doménová jména na legitimní a na vygenerovaná. Pro účely této práce je použit volně dostupný klasifikátor implementovaný Maciejem Andinskim[9]. Implementace tohoto klasifikátoru obsahuje skript na natrénování vlastního modelu. Pro trénování tohoto klasifikátoru jsem použil již existující datovou sadu doménových jmen[10], která obsahuje celkově 157 913 doménových jmen (z toho 80 735 legitimních a 77 178 vygenerovaných pomocí DGA algoritmů (Domain Generation Algorithm). Klasifikátor byl otestován vůči testovací datové sadě, která vznikla rozdělením datové sady a jeho výsledná přesnost je 94 %.

Vzhledem k vysoké úspěšnosti zvoleného klasifikátoru bude použit pro analýzu získaných domén u konkrétní IP adresy. Prvně dojde k filtraci starých záznamů. Pokud je záznam starší než 7 dní, není pro následující rozhodování uvažován. Pro každou IP adresu je uložen

[7] https://passivedns.cesnet.cz
[8] https://www.circl.lu/services/passive-dns/
[9] https://gitlab.nic.cz/adam/dga-classifier-public/-/tree/master
[10] https://raw.githubusercontent.com/andrew-jeremy/Domain-Generation-Algorithm-Detector/master/dga-dataset.txt

přesný počet domén, které se v záznamech u daného zařízení vyskytují. Mnoho domén v záznamech může indikovat zařízení, které je využito pro provoz škodlivých aktivit. Zároveň je každá doména zkontrolována klasifikátorem DGA domén a celkový počet domén klasifikovaných jako DGA je také uložen. Pro každou doménu jsou zkontrolovány její historické záznamy překladů a kontroluje se, kolik těchto záznamů obsahovalo nízké TTL. Nízké TTL je označen takový záznam, jehož TTL bylo nižší nebo rovno 300 sekundám (5 minut je poměrné nízké TTL DNS záznamu). Celková struktura štítků je vidět ve výpisu 5.4.

```
'PassiveDNS': {
    'domain_count': 5,
    'dga_domains_count': 1,
    'low_ttl_count': 0
}
```

Výpis 5.4: Ukázka struktury výsledné anotace podle informací získaných ze služby PassiveDns.

5.5 Anotace s využitím blacklistů a služby TOR

Některé štítky charakterizující zařízení lze získat i kontrolou různých **blacklistů**, protože každý blacklist je zaměřen na určitou škodlivou aktivitu. Pokud se např. IP adresa zkoumaného zařízení vyskytuje na blacklistu uchovávající seznam IP adres, které se pokouší připojit na službu SSH hrubou silou, může být zařízení přidělen štítek `SSH_brute_force`. Podobnými štítky podle charakteristiky blacklistu mohou být štítky `phishing`, `botnet`, `scanner` a další. Tento přístup by ale mohl být problematický, pokud by daný štítek byl přiřazen k více blacklistům (např. dva blacklisty obsahující seznam phishingových domén). Poté by nebylo jasné, na kolika blacklistech daného štítku se zařízení vyskytuje, navíc některý blacklist může být více důvěryhodný než druhý. Z těchto důvodů je pro každý blacklist vytvořen samostatný štítek, který koresponduje s typem blacklistu a název daného blacklistu je přidán za znak „/“ (např. `phishing/openphis`). Ukázka některých použitých blacklistů a korespondujících štítků je vidět v tabulce 5.2.

Posledním zdrojem štítků charakterizujících zařízení jsou informace o službě **TOR**. V kapitole 3.7 bylo zmíněno, že zařízení, které je uzlem v síti TOR, je podezřelé ze škodlivých aktivit. Samotná služba TOR poskytuje seznam všech výstupních uzlů sítě (*exit nodes*). Seznamů, které zveřejňují všechny výstupní uzly sítě TOR je více. Např. na stránkách www.dan.me.uk lze nalézt seznam všech výstupních uzlů služby TOR, ale zároveň zde lze nalézt seznam všech známých uzlů sítě TOR (neobsahuje seznam úplně všech zařízení sítě TOR, některé uzly jsou neznámé). Pokud je určité zařízení výstupním uzlem sítě TOR, je mu přiřazen štítek `TOR_exit` a zařízením, které nejsou výstupními uzly, ale jsou součástí sítě TOR (obyčejné uzly), je přiřazen štítek `TOR`. Štítky získané pomocí služby TOR jsou přiřazeny k ostatním štítkům reprezentující typ zařízení, tedy pod klíč `device_type` výsledné datové struktury reprezentující všechny štítky. Ukázka je ve výpisu 5.5.

[11] https://feodotracker.abuse.ch/downloads/ipblocklist.txt
[12] http://danger.rulez.sk/projects/bruteforceblocker/blist.php
[13] https://www.dshield.org/block.txt
[14] https://openphish.com/feed.txt
[15] https://raw.githubusercontent.com/jjsantanna/booters_ecosystem_analysis/master/booterblacklist.csv

Název blacklistu	**Kontext blacklistu**	**Identifikátor**	**Štítek**
abuse.ch Feodo Tracker[11]	Seznam botnetů	IP adresa	`botnet/feodo_tracker`
Daniel Gerzo's BruteForceBlocker[12]	Pokusy o připojení ke službě SSH hrubou silou	IP adresa	`bruteforce/brute-forceblocker`
DShield.org Recommended Block List[13]	Zařízení skenující otevřené porty	Prefix sítě	`scanner/dshield`
openphis.com feed[14]	Phishingové domény	URL adresa	`phishing/openphis`
Booters[15]	Domény pro koupi DDoS útoku	Doména	`DDoS/booters`

Tabulka 5.2: Ukázka použitých blacklistů a korespondujících štítků, které jsou přiřazeny zařízení, pokud se na daném blacklistu nachází. Identifikátor určuje typ identifikátoru, který je používán blacklistem.

```
'blacklists': ["scanner/dshield", "bruteforce/bruteforceblocker"],
'device_type': [..., "TOR_exit"]
```

Výpis 5.5: Ukázka struktury výsledné anotace získané ze služby TOR a různých blacklistů.

Kapitola 6

Implementace anotátoru datové sady síťových zařízení a vytvoření datové sady

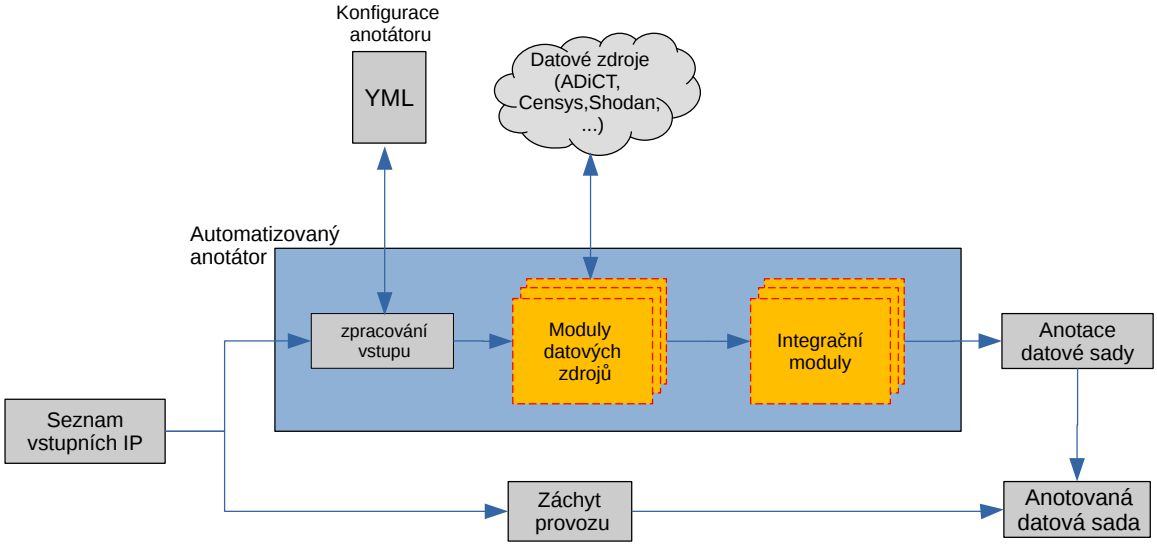
Anotace jakékoliv datové sady je problematická. Některé datové sady lze sice anotovat přímočařeji, např. v případě anotace obrázků detekujících určitý typ objektu stačí správně označit objekt v obrázku, ale i v takovém případě je velmi pracné anotovat tisíce obrázků. V případě síťových zařízení je anotace složitější, protože pouhým pohledem na síťový provoz zařízení není hned jasné, o jaký typ zařízení se jedná. Postup určení typu zařízení byl popsán v předchozí kapitole, stále ještě ale zbývá vyřešit stěžejní část anotace velkého počtu objektů (zařízení), který je automatizace celého postupu anotace. Tato kapitola popisuje návrh a implementaci automatizovaného anotátoru implementovaného v programovacím jazyce Python verze 3, který využívá postup anotace definovaný v předchozí kapitole a umožňuje jeho aplikaci na anotaci stovky i tisíců síťových zařízení. Základní pohled na celý postup anotace datové sady s využitím automatizovaného anotátoru je zobrazen na obrázku 6.1 a jednotlivé části postupu jsou vysvětleny v následujícím textu.

6.1 Zpracování vstupu anotátoru síťových zařízení

Před spuštěním anotátoru je nutné nejprve specifikovat seznam zařízení, které mají být anotovány. Uživatel může zvolit ze dvou základních možností definice tohoto seznamu. První možností je seznam identifikátorů obsahující seznam názvů zařízení (tzv. hostname), seznam IP adres zařízení nebo seznam síťových prefixů v notaci CIDR. Druhou možností uživatele je na vstupu zadat soubor se zachyceným provozem síťových zařízení ve formátu PCAP nebo CSV (aktuálně dva podporované vstupní formáty). Anotátor z tohoto souboru extrahuje seznam IP adres, které byly zachyceny v provozu. Seznam extrahovaných IP adres je možné omezit vstupním filtrem IP adres (síťových prefixů) pro extrahování pouze adres, které spadají do daného filtru.

Svoji volbu možnosti vstupu musí uživatel upřesnit v konfiguraci anotátoru. Konfigurace anotátoru se skládá z jednoho konfiguračního souboru ve formátu YAML[1]. Tento formát byl zvolen pro jeho jednoduchou zpracovatelnost programovacím jazykem Python a dobrou lidskou čitelnost. Lidská čitelnost je v tomto případě důležitá, protože v průběhu této kapitoly bude vysvětleno více konfiguračních položek, které může uživatel nastavo-

[1] https://en.wikipedia.org/wiki/YAML



Obrázek 6.1: Přehled celého procesu anotace datové sady s využitím automatizovaného anotátoru. Uživatel musí nejprve specifikovat seznam IP adres, které chce anotovat. Tento seznam slouží jako vstup k automatizovanému anotátoru, kterému uživatel musí ještě doplnit konfiguraci. Poté anotátor pro každou IP adresu ze seznamu sesbírá informace z datových zdrojů, které poté integruje do finální anotace datové sady. Stejný vstupní seznam IP adres byl použit pro omezení (filtraci) zachytávaného síťového provozu, ke kterému byla vytvořena anotace. Uživatel kromě vstupního seznamu IP adres může použít i jiné možnosti vstupu, viz kapitola 6.1.

vat a je tedy vhodné, aby to pro uživatele bylo co nejjednodušší. V případě konfigurace vstupu zajímá uživatele především klíčová hodnota `input` a její podklíče. Na úrovni podklíče `type` je nutné zvolit typ vstupu. Uživatel může volit mezi dvěmi předem specifikovanými možnostmi (`identifier_list` nebo `captured_traffic`). Poté stačí zadat správnou cestu ke vstupnímu souboru (popř. složce obsahující vstup) a v případě použití možnosti `captured_traffic` jako vstup je ještě možné zadat cestu k filtračnímu souboru IP adres.

V případě první varianty vstupu je každý identifikátor uveden ve vstupním souboru používající jako oddělovač nový řádek. Tento seznam identifikátorů může uživatel definovat v jednom souboru nebo může použít logické členění vstupních zařízení a vstupní seznam identifikátorů tak rozdělit do více souborů. Logické členění na vstupu je poté zachováno na výstupu anotátoru. Po načtení vstupních identifikátorů je seznam identifikátorů převeden na seznam IP adres (jsou načteny přímo nebo v případě seznamu názvů zařízení je proveden převod). Tento seznam IP adres se získanými názvy hosta (pokud má dané zařízení přiřazený nějaký název) slouží jako vstup pro další zpracování datovými zdroji.

6.2 Implementace datových zdrojů pro anotaci

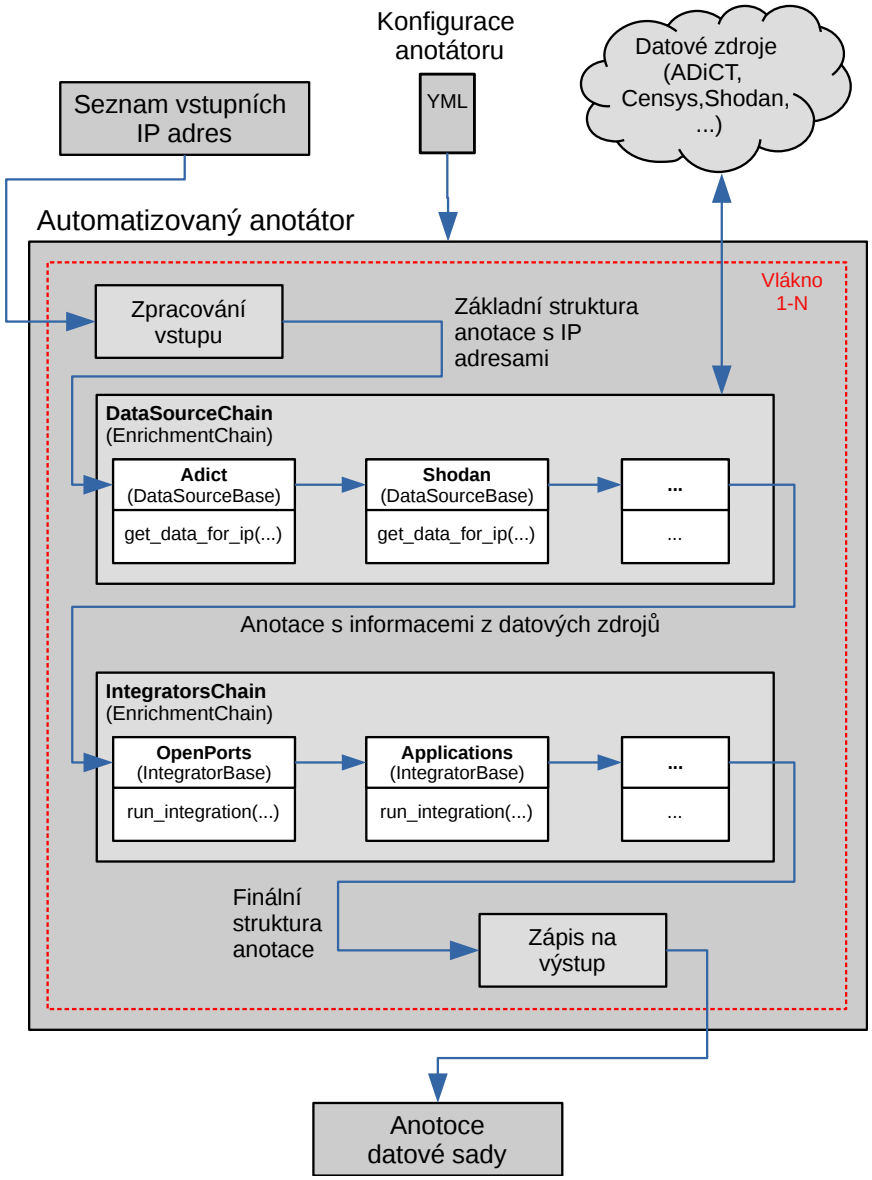
Po úspěšném načtení seznamu zařízení jsou o každém zařízení zjištěny všechny informace potřebné k anotaci. O každém zařízení jsou sbírány informace z více datových zdrojů. Výčet datových zdrojů se může v budoucnu měnit, především mohou přibývat nové datové zdroje ke zpracování a proto bylo nutné navrhnout zpracování datových zdrojů tak, aby bylo co nejjednodušeji rozšiřitelné.

Pro všechny datové zdroje je definována jedna bázová abstraktní třída `DataSourceBase` s abstraktní metodou `get_data_for_ip()`. Tuto funkcionalitu musí implementovat každý datový zdroj a slouží jako vstupní bod pro spuštění procesu získání informací o IP adrese daným datovým zdrojem.

Pro uživatele nemusí být užitečné informace z každého podporovaného datového zdroje a je tedy žádoucí, aby mohl uživatel anotátoru definovat seznam datových zdrojů, které chce použít pro anotaci datové sady. Z tohoto důvodu může uživatel nalézt v konfiguraci anotátoru nastavení seznamu datových zdrojů, které má při svém běhu anotátor použít. Toto nastavení lze najít pod klíči `enrichment` a `enabled_data_sources` a ukázka této konfigurace je zobrazena ve výpisu 6.1. Jelikož se takto může v konfiguraci měnit seznam datových zdrojů, které anotátor využívá, nelze jednotlivé datové zdroje volat ručně každý zvlášť. Proto byla vytvořena abstraktní třída `EnrichmentChain`, jejíž úkolem je načíst z konfigurace seznam povolených datových zdrojů a inicializovat všechny tyto moduly. Definice abstraktní třídy byla vyžadována, protože kromě řetězce datových zdrojů používá anotátor řetězec integračních modulů (popsán v kapitole 6.3). Základní zpracování obou řetězců je podobné, ale drobné detaily se liší a ty už musí definovat každý řetězec zvlášť. Řetězec datových zdrojů si načte seznam povolených datových zdrojů z konfigurace a v balíku datových zdrojů detekuje všechny třídy těchto datových zdrojů, které implementují vstupní metodu zpracování. Podmínkou každého datového zdroje v řetězci tedy je implementovat funkcionalitu abstraktní třídy `DataSourceBase`. Daný řetězec poté už může jednoduše spouštět pro každou IP adresu jeden datový zdroj za druhým a uživatel tak může velmi jednoduše měnit seznam datových zdrojů určených k anotaci pouhou změnou konfigurace. O načtení a spuštění správných datových zdrojů se anotátor postará plně automaticky. Názorněji je vidět popisovaná architektura anotátoru na obrázku 6.2.

```
enrichment:
  enabled_data_sources:
    - "adict"
    - "shodan"
    - "censys"
    - "pdns"
    ...
  data_sources:
    adict:
      url: "http://adict..."
    shodan:
      api_key: "kfuyahsdl124"
    ...
```

Výpis 6.1: Ukázka části konfigurace anotátoru, která slouží pro konfiguraci datových zdrojů a jejich následné integrace. Seznam datových zdrojů, které budou použity anotátorem, je uveden pod klíčem `enabled_data_sources`. Jednotlivé datové zdroje jsou spouštěny ve stejném pořadí, ve kterém jsou uvedeny v konfiguraci. Názvy datových zdrojů se musí shodovat s názvem souboru zdrojového kódu, který implementuje zpracování daným datovým zdrojem (je tedy vyžadováno, aby implementace každého datového zdroje byla oddělena kvůli lepší udržovatelnosti). Zároveň je možné v konfiguraci předat datovým zdrojům dodatečné parametry, jako např. URL datového zdroje nebo klíč sloužící k autentizaci uživatele.



Obrázek 6.2: Podrobnější pohled na architekturu automatizovaného anotátoru. Anotátor na vstupu přijímá seznam vstupních IP adres (existují i jiné možnosti, viz kapitola 6.1), z kterých vytvoří základní strukturu anotace, kterou předá ke zpracování řetězci datových zdrojů. Ten postupně předává základní strukturu anotace modulům implementující datové zdroje. Po zpracování všemi datovými zdroji je struktura se shromážděnými informacemi předána řetězci integrátorů, který také předává strukturu jednotlivým integrátorům. Vzhledem k dynamické konfiguraci datových zdrojů a integrátorů jsou názvy konkrétních datových zdrojů a integrátorů na obrázku pouze jednou z více možností konfigurace anotátoru. Po integraci datovými zdroji je finální anotace zapsána na výstup anotátoru. Celý anotátor umožňuje z optimalizačních důvodů běh ve více vláknech (podrobněji vysvětleno v kapitole 6.4).

Vlastnosti implementace jednotlivých datových zdrojů

U datových zdrojů **Shodan** a **Censys** je nutné řešit omezení propustnosti aplikačního rozhraní, protože při vývoji a testování byly použity uživatelské účty, které používaly bez-

platnou licenci. Obě služby nabízejí základní počet zařízení v bezplatné licenci, o kterých může uživatel získat skrze aplikační rozhraní informace. Určují ale rozdílné omezení propustnosti aplikačního rozhraní. Služba Shodan omezuje dotazování na 1 dotaz za sekundu a služba Censys umožňuje 120 dotazů v časovém okně pěti minut. Omezení ze strany služby Censys je tedy výraznější, ale při produkčním nasazení je očekávané použití komerční licence, která by měla mít větší propustnost přes aplikační rozhraní (na stránkách služby Censys údaj o propustnosti aplikačního rozhraní komerčních licencí bohužel není dohledatelný). Konektor připojující se ke službě Censys implementuje jednoduchý omezovač, který pracuje s pěti minutovým oknem a jakmile se v pěti minutovém okně provede 115 dotazů (rezerva pěti dotazů je z důvodu zaručení nepřekročení limitu, protože server služby Censys penalizuje překročení limitu dalším časovým omezením), konektor se uspí po dobu uplynutí pěti minutového okna.

Při implementaci **konektoru ke službě ADiCT** byly odhalené drobné nekonzistence mezi seznamy `open_ports` a průnikem historických datových bodů za poslední týden. V implementaci aktualizace políčka `open_ports` se nachází drobná chyba, kterou vývojáři služby ADiCT v průběhu implementace anotátoru nestihli opravit a implementace konektoru ke službě ADiCT tedy používá průnik hodnot všech historických datových bodů, které jsou mladší než 7 dní od aktuálního data. Historické datové body jsou už nastaveny správně, protože jsou do databáze vkládány přímo bez pokročilých úprav.

U zpracování **blacklistů** je problematická extrakce identifikátorů (IP adres, síťových prefixů, ...), které jsou na blacklistu uvedeny, protože formáty jednotlivých blacklistů jsou nejednotné. Aby uživatel nemusel implementovat zpracování každého blacklistu zvlášť, byla přidána implementace umožňující automatické zpracování blacklistu na základě regulárních výrazů, které může uživatel nastavit v konfiguraci anotátoru. Uživateli tak stačí vytvořit správný regulární výraz, který v každém řádku blacklistu vyhledá požadovaný identifikátor a anotátor takto extrahuje všechny identifikátory splňující daný regulární výraz. Konfigurace umožňuje zkrácený zápis regulárního výrazu pro vyhledání IP adresy a síťového prefixu, protože se jedná o nejčastější typ identifikátorů uvedených v blacklistech. Je tedy umožněno zpracování i prefixových blacklistů. Všechny blacklisty jsou staženy a zpracovány lokálně při inicializaci řetězce datových zdrojů. Při zpracování konkrétní IP adresy anotátor už jen kontroluje její výskyt v určitém lokálně staženém a zpracovaném blacklistu.

Při inicializaci řetězce datových zdrojů jsou staženy i seznamy uzlů sítě **TOR**, ty ale nevyžadují složitější zpracování jako blacklisty. Inicializovat také potřebuje **geolokační databáze** od společnosti MaxMind[2]. Tato databáze využívá lokálního souboru ve vlastním formátu *mmdb*[3] využívající binární vyhledávací strom pro vyhledání IP adresy. Při inicializaci dojde k stažení archivu, v kterém se databázový soubor nachází a dojde k jeho automatické extrakci a umístění do správné lokace, odkud jej dokáže anotátor používat. Modul PassiveDNS kromě získání základních údajů o IP adrese ze systému PassiveDNS sdružení Cesnet dále předá všechny domény ke kontrole klasifikátorem DGA domén, který byl představen v kapitole 5.4.

6.3 Implementace integrace datových zdrojů

V návrhu anotace síťových zařízení (kapitola 5) byla představena integrace některých datových zdrojů, protože informace z některých zdrojů se do značné míry mohou překrývat.

[2] https://dev.maxmind.com/geoip/geoip2/geolite2/
[3] https://maxmind.github.io/MaxMind-DB/

Jedná se především o vyhledávače Shodan a Censys, ale také o službu ADiCT. Z pohledu udržovatelného návrhu architektury anotátoru je vhodné, aby každý typ integrace zpracovával oddělený modul. Uživatel tedy opět dostává možnost definovat si vlastní seznam povolených integračních modulů, stejně jako to může dělat u datových zdrojů (postup konfigurace datových zdrojů je stejný jako ve výpisu 6.1, jen jsou tyto informace k naleznutí pod klíči `enabled_integrators` a `integrators`). Tyto integrační moduly by teoreticky pro jednoduchost mohly být přidány do řetězce datových zdrojů. Tato myšlenka má ale několik problémů, kvůli kterým došlo v návrhu k vytvoření dvou oddělených řetězců datových zdrojů a integrátorů.

První problém by mohl nastat ve chvíli, kdy by uživatel nastavil špatné pořadí modulů v konfiguraci, která umožňuje měnit pořadí zpracovávajících modulů. Pokud by uživatel umístil nějaký integrační modul, který používá informace z datového zdroje nacházejícího se až na nějaké z dalších pozic, způsobila by se chyba ve zpracování integračním modulem. Pokud by uživatel znal všechny moduly, které nakonfiguroval k anotaci, měl by si sice být této vlastnosti vědom, ale je to zbytečná obtíž, která by mohla způsobit nechtěné komplikace. Další nežádoucí vlastností tohoto návrhu je možná změna struktury anotace v průběhu zpracování řetězcem. Integrátory potřebují měnit strukturu anotace, jelikož přidávají nové informace a mohou měnit i staré informace. Datové zdroje by ale měly pouze informace přidávat a v žádném případě by neměly měnit informace získané z jiných datových zdrojů.

Řešením obou problémů je rozdělení datových zdrojů a integrátorů do oddělených řetězců. Nejprve jsou spuštěny postupně v nakonfigurovaném pořadí datové zdroje, které mohou do anotační struktury pouze přidávat nové informace, ale nemohou nijak měnit již existující informace, aby bylo zaručeno, že při spuštění integračního procesu se nacházejí informace ze všech datových zdrojů v anotační struktuře. Poté jsou spuštěny v nakonfigurovaném pořadí integrátory. Tento přístup předchází zbytečným problémům, protože nejprve jsou spuštěny všechny moduly, které pouze získávají informace a až po dokončení sběru všech informací přichází na řadu jejich integrace. Jedná se i o lépe přestavitelný průběh celé anotace.

Vlastnosti implementace jednotlivých integrátorů

Společnou vlastností integrátorů je kontrola dostupnosti získaných dat datovými zdroji. Některé zdroje nemusí mít informace o dané IP adrese, proto je nutné si z používaných datových zdrojů integrátorem vytřídit pro danou IP adresu ty, kterým se povedlo o IP adrese zjistit nějaké informace. Následně lze na dostupných informacích spustit proces integrace.

První z implementovaných integrátorů implementuje **integraci otevřených portů** daného zařízení podle návrhu uvedeného v kapitole 5 a na obrázku 5.1. Integrátor otevřených portů musí před spuštěním samotné integrace zkontrolovat stáří získaných informací z vyhledávačů Shodan a Censys. Pokud jsou informace dostatečně aktuální, jsou použity k integraci. Podle postupu uvedeném v návrhu integrace dojde ke sjednocení otevřených portů a k jejich překladu na název služby je použit balíček `whatportis`[4] programovacího jazyka Python, který stahuje a následně používá informace o registrovaných službách z oficiální databáze registrovaných služeb spravované společností IANA.

Další integrátory slouží k **integraci informací o operačních systémech a bežících aplikacích** získaných z vyhledávačů Shodan a Censys. Zde opět dochází ke kontrole aktuálnosti informací a následně je spuštěna integrace. Posledním velmi jednoduchým integračním modulem je modul zvaný `cleaner`, který by měl být spuštěn vždy úplně na konci integrač-

[4] https://pypi.org/project/whatportis/

ního řetězce. Jediným úkolem tohoto modulu je odstranění nežádoucích informací z finální struktury anotace, které vznikly v průběhu anotace dané IP adresy. Např. moduly datových zdrojů Shodan a Censys shromažďují o malinko více informací, než které potřebují k finální anotaci. Po ukončení integrace uživatele vůbec nemusí zajímat tyto dodatečné informace a proto může v konfiguraci modulu `cleaner` definovat klíče, které chce na konci procesu anotace automaticky odstranit z finální struktury anotace. Toto mazání informací z finální struktury anotace je ale naprosto volitelné a uživatel nemusí mazat informace žádné. Může si například informace z použitých datových zdrojů ponechat pro nějakou další budoucí analýzu nebo úpravu anotace, na kterou přijde v pozdější fázi výzkumu anotované datové sady.

6.4 Testování anotátoru a optimalizace výkonnosti

Implementace anotátoru obsahuje i několik jednotkových testů, které se zaměřují na implementaci integrátorů. Každý integrátor (kromě integrátoru `cleaner` pro jeho jednoduchost) je testován sadou testů, které v průběhu vývoje pomohly odhalit několik chyb v implementaci. Testy definují několik ukázkových očekávaných vstupů, které mohou integrátory očekávat a testují správnost jejich výstupu.

Mezi jednotkovými testy anotátorů nejsou testy datových zdrojů, kvůli proměnlivosti informací o zařízeních, které získávají z aktuálních dat o zařízeních. Vstupní data by šla použít nějaká testovací, ale značná část implementace řeší právě část komunikace s datovými zdroji a zbytek implementace již neobsahuje příliš náročné zpracování. Kritickou částí zpracování jsou integrátory, které jsou řádně otestovány.

Při anotaci datové sady s využitím automatizovaného anotátoru se objevil problém výkonnosti anotátoru. Délka anotace 100 IP adres trvá anotátoru přibližně 5 minut a 30 sekund. To je přibližně 3.3 sekundy na jednu IP adresu. Celková rychlost procesu anotace není kritickou částí aplikace, ale při anotaci většího počtu zařízení by se celkový čas mohl dostat až na jednotky hodin a to už může být uživatelsky nepříjemné. Z tohoto důvodu byla k implementaci přidána podpora využití více výpočetních vláken anotátorem. V programovacím jazyce Python musí být při optimalizaci výkonu aplikace brán ohled na globální zámek interpretu (GIL - Global Interpreter Lock), protože ten umožňuje v jednom okamžiku běh pouze jednoho vlákna. Z tohoto důvodu není vhodné použití vláken ve výpočetně náročných aplikacích, které výrazně vytěžují procesor. Implementace anotátoru ale neobsahuje žádné příliš výpočetně náročné operace, problém výkonnosti spočívá především v síťové komunikaci, kdy hlavní vlákno pošle síťový požadavek na aplikační rozhraní některé z používaných služeb a poté musí čekat na odpověď a další požadavek může poslat až poté, kdy obdrží odpověď na první požadavek. To se ale může protáhnout zpracováním na straně serveru provozující aplikační rozhraní a také přenosem dat. V tomto případě, kdy je program brzděn vstupně výstupními operacemi, lze použít vlákna pro optimalizaci programu, protože jakmile některé z vláken programu odešle např. síťový požadavek, vlákno uvolní globální zámek interpretu dalšímu vláknu, které v mezičase může odeslat další síťový požadavek. Jakmile dorazí odpověď prvnímu vláknu, opět si požádá o globální zámek interpretu a začne zpracovávat odpověď. Takto ale může být odesláno více síťových požadavků souběžně a celková doba čekání na odpovědi se výrazně zkrátí.

Uživatel má tedy možnost v konfiguraci anotátoru nastavit počet vláken, na kterých má anotátor běžet. Stále tak může použít pouze 1 vlákno, ale také může použít vláken třeba 8. Anotace 100 IP adres s využitím 8 vláken se tak zkrátila z 5 minut a 30 sekund na 2 minuty. Na první pohled to s využitím 8 vláken není takové zrychlení, které by člověk

mohl očekávat. V tomto případě ale anotátor nenaráží na svoje limity, ale naráží na limity rychlostního omezení aplikačních rozhraní některých z používaných služeb. V tomto případě se jedná o omezení rychlosti na 1 požadavek za 1 sekundu vyhledávačem Shodan. Při anotaci 100 IP adres lze tedy očekávat nejrychlejší možný výstup za 1 minutu a 40 sekund. Dalších 12 sekund zabralo předzpracování v podobě sběru názvů jednotlivých zařízení (hostname), které probíhá před spuštěním celé anotace (běží ale také ve více vláknech). Ze zbylých 8 vteřin sebrala část režie přepínání vláken a část také samotné zpracování dat.

6.5 Analýza vytvořené datové sady sítových zařízení

Záchyt síťového provozu zařízení v datové sadě probíhal na serverových strojích sdružení Cesnet, které mají pro účely záchytu provozu nainstalovaný vysoko výkonnostní kolektor NetFlow dat zvaný IPFIXcol2[5]. Data proudící z kolektoru síťového provozu poté zpracovávají různé moduly systému NEMEA[6], skrze který je za použití modulu `unirecfilter` jednoduché spustit záchyt provozu a jeho ukládání do souborů na disk. Záchyt provozu byl aplikací filtru omezen pouze na IP adresy, které patří do datové sady síťových zařízení. Výstupní formát zachyceného provozu je serializovaný formát `trapcap`[7], který lze pomocí modulu `logger` překonvertovat do formátu CSV, který už je lehce zpracovatelný programovacím jazykem Python.

Celkově se v datové sadě nachází 2300 zařízení, které se nacházejí v univerzitním prostředí a jedná se o serverové stroje, ale i o uživatelské stanice. Záchyt provozu probíhal celý jeden týden od pondělní půlnoci do nedělní půlnoci a záchyt komunikace probíhal na třech datových zdrojích, mezi které patřily základní flow data (Netflow verze 5 pokrácen o nedůležité atributy, přehled položek je zobrazen v příloze C.1), HTTP a DNS data. Záchyt každého datového zdroje je rozdělen do 168 souborů (po 1 hodině) a ve čtvrtek v průběhu dne byl spuštěn automatizovaný anotátor nad všemi IP adresami, jejichž síťový provoz byl určen k záchytu. Každý datový soubor obsahuje přibližně 2 - 3 miliony záznamů (toků).

Na vytvořené anotaci byla provedena **jednoduchá korelační analýza štítků** určujících typ zařízení (položka anotace `device_type`), jejíž cílem bylo prozkoumat spojitosti mezi jednotlivými typy zařízení. Mezi typy zařízení bylo nalezeno pouze několik spíše slabších pozitivních korelací. Největší pozitivní korelace (0.7), je mezi štítky `Apple` a `MacOS` a mezi štítky `ICS` a `IoT` (0.59). Třetí největší korelace (0.43) je mezi štítky `MacOS` a `password server`, která už je ale slabší, stejně tak jako zbytek korelací. Mezi zajímavé korelace lze už zařadit pouze korelaci mezi štítky `mail server` a `proxy server` (0.36) a korelaci mezi štítky `mail server` a `web server` (0.3).

Následně byla provedena analýza informací získaných z jednotlivých datových zdrojů. Nejprve byly zanalyzovány, vzhledem k jejich podobnosti, informace z nástrojů ADiCT, Shodan a Censys. Na všech zařízeních bylo dohromady detekováno 45 644 otevřených portů, z nichž 45 409 bylo získáno pomocí služby ADiCT. Ze všech portů bylo detekováno 815 portů vyhledávačem Shodan a 1 044 portů vyhledávačem Censys, přičemž na 546 otevřených portech se shodly všechny 3 služby a na 562 otevřených portech se spolu shodly vyhledávače Shodan a Censys, aniž by je detekovala služba ADiCT. Z těchto informací lze usoudit, že největší vliv na množství detekovaných otevřených portů má služba ADiCT a vyhledávače Shodan a Censys detekují pouze malé množství otevřených portů. To může být způsobeno malou množinou portů, kterou vyhledávače skenují, ale také množstvím zařízení, o kterých

[5] https://github.com/CESNET/ipfixcol2
[6] https://github.com/CESNET/Nemea
[7] https://nemea.liberouter.org/trap-ifcspec/

mají tyto vyhledávače dostupné informace. Z 2 300 zařízení nebyly nalezeny žádné informace o 1 800 zařízeních vyhledávačem Shodan a o 1 896 zařízeních vyhledávačem Censys. Vyhledávači Shodan se tak povedlo získat informace pouze o 22 % zařízeních a vyhledávači Censys pouze o 18 % zařízeních. Vliv na nízké množství oskenovaných zařízení může mít dobré zabezpečení univerzitní sítě a vystavení síťových zařízení do internetu pouze v nutných případech. Z pohledu důvěrnosti informací jsou nejvíce důvěryhodné takové porty, které jsou detekovány všemi třemi službami, to je ale pouze 1,2 % zařízení. Z pohledu anotace má tak vzhledem k množství detekovaných portů nejvyšší váhu systém ADiCT. Tato informace byla ale zmíněna již v návrhu anotace v kapitole 5.1.

I přes malý počet zařízení bylo pomocí vyhledávačů Shodan a Censys detekováno 37 různých aplikací a 10 různých operačních systémů (seznamy detekovaných aplikací a operačních systémů jsou uvedeny v příloze C, ve výpisech C.2 a C.1). Z toho u 150 zařízení detekoval operační systém vyhledávač Censys a pouze u 38 zařízení byl operační systém detekován vyhledávačem Shodan, přičemž pouze u 2 zařízení se vyhledávače v informaci o operačním systému lišily. Zároveň vyhledávač Censys detekoval 321 aplikací a vyhledávač Shodan detekoval aplikací 284. Z pohledu detekce operačního systému se tedy potvrdil původní předpoklad, že v detekci je úspěšnější vyhledávač Censys. Z pohledu počtu detekovaných aplikací není rozdíl závažný. Vyhledávač Shodan navíc dále odhalil 255 různých zranitelností.

Žádné ze zařízení se nenacházelo na ani jednom z nakonfigurovaných blacklistů a také u žádného zařízení nebyla detekována tzv. DGA doména ani DNS záznam s nízkým TTL. Pravděpodobným vysvětlením těchto faktů je dobré zabezpečení univerzitní sítě, z které pocházejí zařízení v datové sadě.

Pro zveřejnění datové sady je nutné na závěr provést anonymizaci IP adres. Všechny IP adresy výsledné anotované datové sady byly anonymizovány pomocí anonymizačního nástroje `cryptopan`, který je součástí systému NEMEA vyvíjeného sdružením Cesnet, který byl představen na začátku této kapitoly a byl používán pro záchyt síťového provozu.

Kapitola 7

Klasifikace síťových zařízení s využitím metody strojového učení

V této kapitole je popsán způsob experimentálního využití vytvořené anotované datové sady, která byla vytvořena pomocí automatizovaného nástroje. Po analýze jednotlivých přístupů ke klasifikaci síťových zařízení v kapitole 4 byl pro účely této práce zvolen tzv. *host-based* přístup využívající statistického chování zařízení. Mezi hlavní důvody patří odolnost vůči šifrovanému síťovému provozu a také jeho adaptovatelnost na více různých typů provozu. Oproti *flow-based* přístupu je také vhodný pro klasifikaci typu zařízení, protože zkoumá generovaný síťový provoz z pohledu statistické chování celého zařízení a nesnaží se o klasifikaci jednotlivých protokolů, které jsou použity v komunikaci.

7.1 Předzpracování datové sady a výpočet rysů pro strojové učení

Pro účely klasifikace byla ze tří typů zachyceného provozu (základní flow data, HTTP a DNS data) využita pouze část obsahující základní flow data, v kterých se nacházejí pouze základní informace o provedené síťové komunikaci jednotlivých zařízení (přehled atributů základních flow dat se nachází v tabulce C.1). V rámci optimalizace prostorové a časové náročnosti vytvořené datové sady byly soubory zachyceného provozu uloženy místo původního formátu `CSV` ve formátu `Parquet`. Formát Parquet je vhodným formátem pro dlouhodobé uložení datové sady, jehož hlavní výhodou je značná redukce prostorové náročnosti dat, které si lze jasně všimnout v článku porovnávajícím více různých formátů pro uložení datových sad [28]. Po převodu základních flow dat datové sady z formátu CSV do formátu Parquet byla snížena velikost z 36.5 GB na 7.5 GB.

Po úpravě datové sady byl proveden návrh rysů, které budou následně použity algoritmem strojového učení. Zvolený přístup ke klasifikaci se řídí statistickými vlastnosti provozu sledovaných zařízení a každá statistika (rys) je vypočtena za jednu hodinu síťového provozu zařízení. Celkově je tedy každý datový soubor základních flow dat převeden na soubor obsahující vypočtené statistiky pro každou IP adresu. Tyto předvypočtené souhrnné statistiky lze nazývat krátkodobým profilem daného zařízení. Důležitou vlastností těchto statistik je jejich výpočet zvlášť pro všechen příchozí síťový provoz a zvlášť pro všechen odchozí síťový provoz. Toto dělení statistik poskytuje dvě výhody. První výhodou, která byla představena

Popis statistiky	**Důvod výpočtu**
Celkový počet toků	Reflektuje celkovou aktivitu uzlu
Celkový počet paketů	Chování hosta, mnoho paketů indikuje mnoho krátkých spojení
Celkový počet bytů	Mnoho přenesených dat indikuje velké datové přenosy
Suma doby trvání všech toků	Zobrazuje celkové vytížení zařízení
Průměrná doba toku	Odlišuje připojené a nepřipojené toky
Průměrná velikost paketu	Malé pakety často slouží jako signalizace určitého protokolu, velké pakety indikují datové přenosy
Průměrný počet paketů na hosta	Rozlišuje velké toky od malých
Průměrný počet bytů na hosta	Může indikovat datové přenosy

Tabulka 7.1: Souhrnné agregační statistiky provozu zařízení za 1 hodinu. Statistiky jsou zde zobrazeny z pohledu příchozího provozu, stejné statistiky jsou vypočteny i pro odchozí provoz. Např. průměrný počet paketů na hosta tedy znamená průměrný počet paketů za všechny zdrojové IP adresy, které jsou v příchozím provozu daného zařízení.

v bakalářské práci Pavla Nováka [19], je odlišení poměru příchozího a odchozího provozu, které může pomoci klasifikátoru jednodušeji určit rozdíl mezi klientskou stanicí a serverem. Druhou výhodou je možnost přímého výpočtů poměrů mezi statistikami příchozího a odchozího provozu. Pro téměř každou statistiku lze vypočítat tento poměr, který jako samostatný rys pomůže klasifikátoru ještě více.

Použité statistiky lze rozdělit do několika kategorií podle podobnosti jejich výpočtu. První kategorií **jsou souhrnné agregace provozu**, jejichž přehled lze vidět v tabulce 7.1. Jedná se o aplikaci základních agregačních funkcí jako je suma a průměr, které poskytnou pohled na množství provozu, s kterým zařízení přijde v rámci jedné hodiny do styku. Dalšími používanými statistikami jsou **počty unikátních hodnot atributů** v komunikaci jako je např. počet hostů, s kterými dané zařízení komunikovalo. Tento typ statistik zobrazuje různorodost komunikace daného zařízení. Celý výčet unikátních počtů hodnot je zobrazen v tabulce 7.2.

Mezi další statistiky je vhodné zařadit **přehled nejvíce používaných hodnot**, které zobrazují nejvíce používané hodnoty atributů komunikace, jako např. čísla portů, ale i nejvíce se objevující komunikující hosté a jejich geografická lokace. Tato statistika je uložena formou výčtu hodnot, které mohou nabývat i textové podoby. Zvolený klasifikační algoritmus Náhodné lesy ale umí používat pro rozhodování při klasifikaci pouze číselné údaje. V případě malého výčtu možných hodnot se dá v podobných případech využít techniky zvané `One-hot-encoding`[1], ale celkový počet možných portů, IP adres komunikujících hostů a jejich zemí je příliš veliký.

[1] https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/

Popis statistiky	Důvod výpočtu
Počet hostů	Zobrazuje celkovou aktivitu hosta
Počet cílových portů	Množství aplikací, s kterými host komunikuje
Počet zdrojových portů	Množství aplikací, s kterými host komunikuje
Počet protokolů	Zobrazuje odlišnost použitých protokolů
Počet autonomních systémů	Geolokační rozložení komunikace
Počet států	Geolokační rozložení komunikace

Tabulka 7.2: Přehled unikátních hodnot jednotlivých atributů komunikace, které se objevily v síťové komunikaci hosta. Statistiky jsou zobrazeny z pohledu příchozího provozu, stejné statistiky jsou vypočteny i pro odchozí provoz. Počet unikátních hostů takto indikuje počet unikátních zdrojových IP adres v komunikaci, protože cílová IP adresa je vždy stejná (zkoumaný host).

Přehled nejvíce používaných portů v komunikaci zařízení a především i množství komunikace provedené skrze určitý port je ale velmi důležitá informace. Proto byla provedena jednoduchá analýza nejpoužívanějších portů v zachyceném provozu. Z každého zachyceného dne provozu (celkem 7) bylo extrahováno 30 nejpoužívanějších zdrojových portů a 30 nejpoužívanějších cílových portů. Nad těmito množinami byl provedl průnik, jehož výsledek je celkem 17 nejpoužívanějších čísel portů v zachyceném provozu. Pro každý směr komunikace je do souhrnných statistik přidáno všech 17 čísel portů (seznam portů je uveden v příloze C ve výpisu C.3), jejichž hodnoty nesou počet provedených toků skrze daný port. Počet provedených toků je vypočten pro všechny zdrojové i cílové porty v příchozí i odchozí komunikaci.

Posledními použitými statistikami jsou dva **poměrové rysy**. Těmito rysy jsou poměr zdrojových portů vůči počtu hostů a stejný poměr je vypočten i pro cílové porty. Tyto poměry mohou pomoci rozlišit serverový stroj od klientské stanice, protože server většinou naslouchá na jednom portu, ale klient navazuje spojení z více různých (často náhodných) portů. Pro souhrnné agregace provozu a unikátní počty je ve finální fázi vypočten poměr mezi těmito statistikami příchozího a odchozího provozu.

Průběh výpočtu statistik spočívá v načtení základních flow dat do datové struktury `DataFrame` knihovny `Pandas`[2], která slouží pro datovou analýzu a nabízí spoustu vestavěných a především optimalizovaných funkcí, které usnadňují výpočet těchto statistik.

Vzhledem k velkému počtu toků (přibližně 2 - 3 miliony za každou hodinu zachyceného provozu), z kterých jsou statistiky vypočteny, bylo nutné urychlit některé pomalé části výpočtu. Jedná se především o získávání informací o geografické poloze zařízení (stát) a číslo autonomního systému. Informace jsou získávány z lokálních databázových souborů používaných služeb, které jsou optimalizovány pro větší výkon. Databázové souboru jsou inicializovány před započetím výpočtu statistik. Používanými službami jsou geolokační databáze GeoLite2, která byla využita již při anotaci datové sady, a knihovna `pyasn`[3], která umožňuje jednoduché dotazování na autonomní systém IP adresy. Dotazování z lokálního

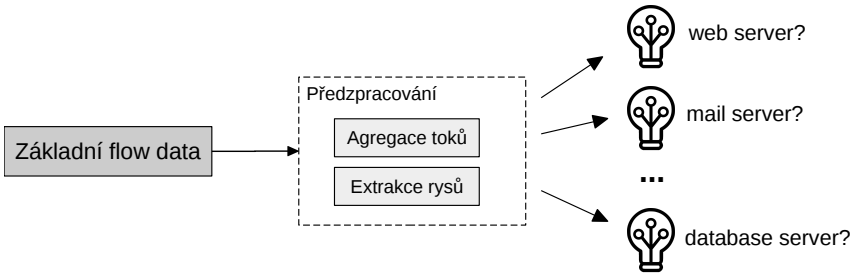
[2] https://pandas.pydata.org/
[3] https://pypi.org/project/pyasn/

databázového souboru výrazně snižuje zátěž síťové komunikace, která by mohla být potřeba při dotazování se externích zdrojů. Získání informací o geografické lokaci zařízení je i tak velmi pomalé pro vysoký počet toků v datové sadě. Mnoho IP adres se ale v jednotlivých tocích často opakuje, proto byla do programu přidána cache, která uchovává informace o již vyhledaných IP adresách. Přidání cache výpočet výrazně urychlil a výpočet statistik pro všechna zařízení v datové sadě (cca 2300 zařízení) z jedné hodiny jejich provozu tak zabere přibližně 20 sekund. Statistiky by měly být vypočteny každou hodinu a vzhledem k aktuální rychlosti výpočtu by měl výpočet statistik pro 100 000 zařízení zabrat přibližně 15 minut. Celkový výpočet by se dal na některých místech stále asi urychlit, ale již v této verzi je výpočet použitelný pro velký počet zařízení. Bylo by ale nutné rozdělit výpočet statistik na menší části, které by se poté sloučily, protože načtení všech toků do paměti RAM by vyžadovalo velkou kapacitu pamětí RAM.

7.2 Trénování klasifikačního modelu a provedené experimenty

Před začátkem trénování klasifikačního modelu je potřeba zvolit štítky z anotace, které bude klasifikátor predikovat. Jednou z nejzajímavějších informací z anotační struktury je typ zařízení (štítek `device_type`). Proto byla provedena analýza těchto štítků a pro testování klasifikace byly vybrány štítky s největším zastoupením v datové sadě. Mezi tyto štítky patří štítky `Windows`, `web server`, `file server`, `time server`, `database server`, `mail server`, `DNS server` a `IoT`.

Každé zařízení může být anotováno více štítky (může provozovat více služeb). Proto byl pro každý ze zvolených štítků natrénován samostatný binární klasifikátor, jehož úkolem je z rysů vstupního profilu zařízení predikovat, zda se jedná o profil zkoumaného typu zařízení nebo nikoliv a ve výsledku je tak možné zkoumanému zařízení přiřadit více štítků najednou (přehled je vidět na obrázku 7.1). Pro trénování je tedy nutné zvolit jeden finální štítek, který bude klasifikátor predikovat. Pro každý profil, které byly vytvořeny z datové sady, byla provedena automatizovaná anotace. Jejím úkolem je podle volby daného štítku (např. `web server`) přiřadit zvolený štítek danému profilu, pokud se takový štítek nachází v množině štítků zařízení (reprezentované anotovaným profilem) v anotační struktuře pod klíčem `device_type`. Pokud štítek přítomen není, je profilu přiřazen štítek `other`, který klasifikátoru sděluje, že se jedná o profil, který nereprezentuje typ zkoumaného zařízení.



Obrázek 7.1: Architektura způsobu predikce typu zařízení. Každé zařízení může provozovat více služeb a je tedy použito několik binárních klasifikátorů, které dohromady mohou zařízení přiřadit několik štítků.

Pro trénování klasifikačního modelu byla použita knihovna `scikit-learn`[4] a vzhledem k pozitivním vlastnostem algoritmu Náhodné lesy, které byly popsány v kapitole 4.2, byl použit klasifikátor `RandomForest`. Jedinou úpravou výchozího nastavení klasifikátoru bylo zvýšení počtu vytvořených stromů z výchozích 100 stromů na 300 stromů. Větší počet stromů by neměl mít zásadní vliv na přesnost klasifikačního modelu a spíše by zpomaloval klasifikační proces [20]. Datová sada (celkem 116 080 profilů) byla pro každý klasifikátor rozdělena v poměru 7,5 : 2,5 na trénovací a testovací část.

Pro hodnocení přesnosti klasifikátoru byla použita matice záměn (tzv. confusion matrix), která definuje 4 základní parametry:

- **True Positive** (TP) a **True Negative** (TN) - Počet správně predikovaných štítků dané třídy, kde Positive reprezentuje první třídu (např. `web server`) a Negative reprezentuje druhou třídu (`other`).
- **False Positive** (FP) a **False Negative** (FN) - Počet nesprávně predikovaných štítků dané třídy, tedy např. počet predikcí štítku `web server`, přičemž měl být predikován štítek `other`.

Celková přesnost klasifikace (*Accuracy*) se poté získá jako podíl správně klasifikovaných záznamů a všech záznamů:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Dalšími metrikami přesnosti klasifikace jsou tzv. **Recall** a **Precision**. Obě metriky se počítají zvlášť pro každou klasifikační třídu a jsou uváděny také v procentech. Vyšší hodnota metriky Recall reflektuje vysokou míru důvěry ve výsledek klasifikace v případě označení profilu daným štítkem. Vyšší hodnota metriky Precision reflektuje nízký počet vzorků klasifikovaných jako false positive, tedy chybné označení profilu daným štítkem (např. `web server`), přičemž správné označení je štítek druhý (štítek `other`). Vzhledem k odlišnosti těchto metrik je často používána metrika **F1-score**, která rovnoměrně kombinuje obě metriky do jedné hodnoty. Výpočet těchto metrik je následující:

$$Recall = \frac{TP}{TP + FN} \qquad Precision = \frac{TP}{TP + FP} \qquad F1\text{-}score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Výsledky přesnosti klasifikace po natrénování a otestování klasifikátorů jsou zobrazeny v tabulce 7.3 ve sloupci *Přesnost bez ADASYN*. Přesnost klasifikace některých štítků a metrika F1-score jsou nízké a z analýzy klasifikovaných dat bylo zjištěno, že nižší přesnost klasifikace může způsobovat nevyváženost tříd v datové sadě. Pro vyvážení dat v datové sadě se často používá metoda převzorkování datové sady nebo generování nových syntetických vzorků, které reprezentují minoritní třídu. V případě vzorkování datové sady dochází ke snížení celkového počtu dat pro trénování a to má často negativní vliv na výslednou přesnost klasifikátoru, který potřebuje co nejvíce trénovacích dat. Z těchto důvodů může vykazovat lepší výsledky generování nových syntetických dat. Mezi často používané metody patří metoda SMOTE a metoda ADASYN [8]. Metoda SMOTE generuje pro každý vzorek minoritní třídy stejný počet nových vzorků. Metoda ADASYN se naopak snaží jednotlivým vzorkům minoritní třídy přiřadit určitou váhu, která je vyšší u menšinových vzorků, tedy u vzorků, v jejichž okolí se nachází méně ostatních vzorků. Nové vzorky jsou tak generovány

[4] https://scikit-learn.org/stable/

Predikovaný štítek	**Přesnost, F1-score** bez ADASYN	**Přesnost, F1-score** s ADASYN
`database server`	92,69 %, 69,36 %	95,24 %, 95,28 %
`DNS server`	99,12 %, 82,92 %	99,53 %, 99,53 %
`file server`	89,90 %, 88,39 %	89,93 %, 89,65 %
`IoT`	97,62 %, 81,67 %	98,57 %, 98,60 %
`mail server`	91,24 %, 65,50 %	93,53 %, 93,38 %
`time server`	90,64 %, 69,75 %	92,96 %, 92,47 %
`web server`	84,43 %, 86,27 %	84,60 %, 85,85 %
`Windows`	85,37 %, 91,34 %	89,65 % 90,04 %

Tabulka 7.3: Přehled přesností a metriky *F1-score* jednotlivých klasifikátorů před použitím metody ADASYN pro generování syntetických vzorků dat a po použití metody ADASYN.

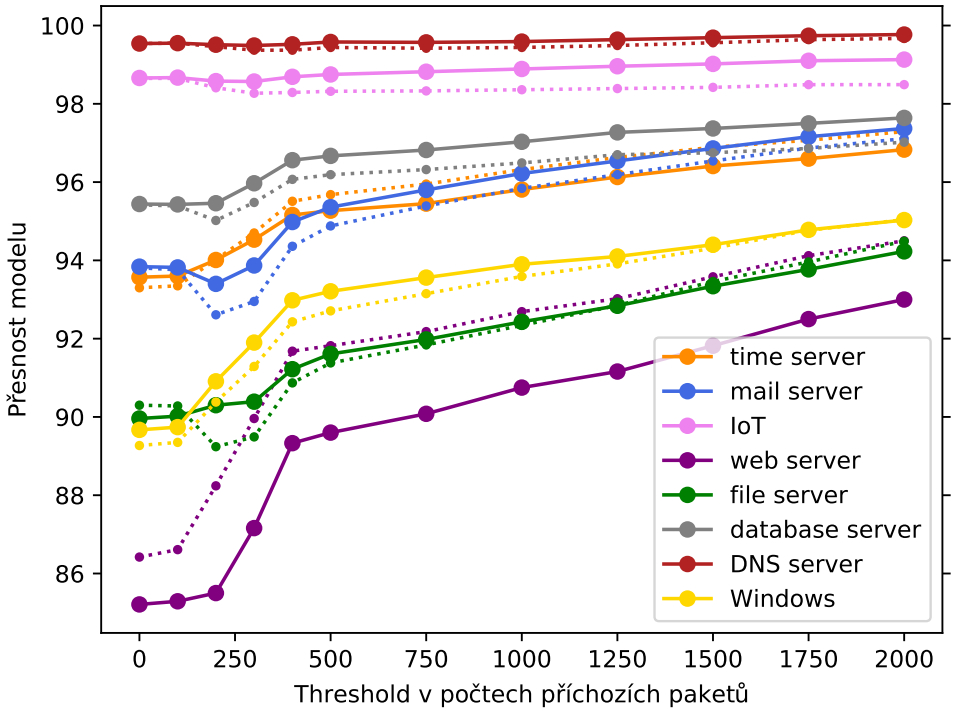
ve větším počtu v blízkosti vzorků s vyšší vahou, čímž je kladen větší důraz na vzorky, které jsou za normálních podmínek hůře klasifikovatelné vzhledem k jejich menšině a tím může být dosaženo lepší výsledné rozlišovací schopnosti u těžko predikovatelných vzorků.

Vliv aplikace metody ADASYN na datovou sadu před spuštěním trénovacího procesu na úspěšnost klasifikace je vidět v tabulce 7.3. Každý klasifikátor určitého štítku zvýšil svoji klasifikační přesnost přibližně o 3 %. Při podrobnější analýze je ale vidět, že zásadní zlepšení nastalo u metriky F1-score, která je velmi náchylná na nevyváženost tříd v datové sadě a aplikací metody ADASYN tak došlo k výraznému zlepšení obou metrik Precision a Recall a tedy i ke zlepšení souhrnné metriky F1-score.

V průběhu další analýzy byl vypočten průměrný rozdíl hodnot rysů profilů, které byly klasifikovány správně a profilů, které byly klasifikovány chybně. Ze získaného rozdílu bylo viditelné, že klasifikátory většinou špatně predikují profily, které obsahují málo síťového provozu, především malý celkový počet přenesených paketů a provedených toků. Pokud zařízení generuje málo síťového provozu, nemusí být vytvořen dostatečný statistický profil, podle kterého by šlo určit typ zařízení. Z těchto důvodů byl proveden experiment, který testuje vliv profilů s nízkým síťovým provozem na celkovou přesnost klasifikátoru. Pro každý klasifikátor (predikovaný štítek) bylo vyzkoušeno několik thresholdů, které představují minimální nutný počet přenesených paketů v příchozím provozu klasifikovaného profilu, aby mohl být profil zařízení předán klasifikátoru. Tento experiment simuluje nasazení v reálném provozu, kde je očekávatelné, že pokud zařízení neprodukuje dostatečný síťový provoz, je obtížné správně klasifikovat správný typ zařízení.

Přehled přesností klasifikátorů po aplikaci vstupního thresholdu je vidět na obrázku 7.2. Z výsledku experimentu lze usoudit, že čím striktnější je vstupní filtr, tím je dosaženo vyšší přesnosti i metriky F1-score. Při aplikaci vstupní podmínky v podobě thresholdu je ale nutné zkoumat i počet profilů, které nesplní vstupní podmínku klasifikace a nebudou tak klasifikovány vůbec. Přehled počtu profilů, které jsou klasifikovány po aplikaci určitého thresholdu je zobrazen na obrázku 7.3. Vzhledem k výrazně klesajícímu počtu klasifikovatelných profilů po aplikaci vstupního thresholdu není možné použít jakoukoliv hodnotu vstupního

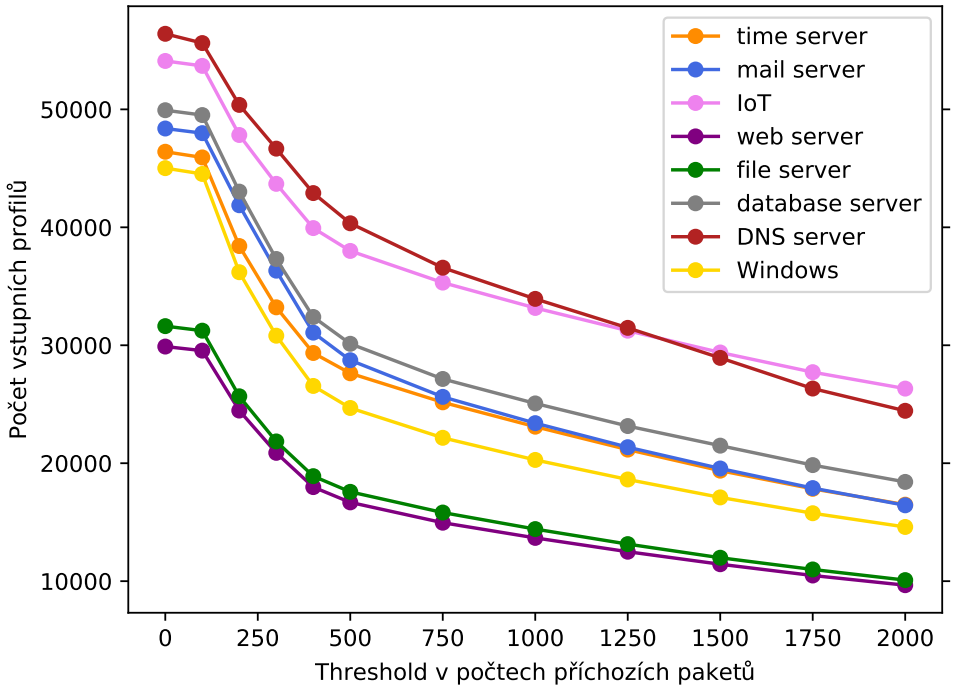
thresholdu. Při volbě hodnoty thresholdu je nutné zvolit vhodný kompromis mezi přesností klasifikátoru a množstvím neklasifikovatelných profilů zařízení. Nejvyšší míry zlepšení přesnosti klasifikátorů je dosaženo při aplikaci vstupního thresholdu o velkosti 400-500 příchozích paketů. Při tomto nastavení dochází k odmítnutí přibližně jedné třetiny vstupních profilů. To je poměrně vysoký počet odmítnutých profilů, při reálném nasazení ale nemusí být použita pouze jedna hodnota vstupního thresholdu. U každého vstupního profilu se může zkontrolovat počet vstupních paketů a podle této hodnoty lze výsledku klasifikace nastavit určitou míru důvěry. Vyšší míra důvěry v klasifikovanou informaci bude přiřazena profilu s vyšším počtem vstupních paketů a naopak menší míra důvěry bude přiřazena v případě nízkého počtu vstupních paketů, protože v takovém případě je obtížné takový profil klasifikovat.



Obrázek 7.2: Přehled přesnosti jednotlivých klasifikátorů při aplikaci určitého thresholdu v počtu **příchozích paketů**. Plná čára klasifikátoru představuje celkovou přesnost modelu, přerušovaná čára zobrazuje metriku *F1-score* predikovaného štítku (typu zařízení).

Kromě počtu vstupních paketů byly jako vstupní threshold testovány i počet vstupních toků, počet výstupních paketů a počet výstupních toků. Vliv na přesnost a počet klasifikovaných profilů je přibližně stejný a lze jej vidět v příloze C na obrázcích C.3, C.1 a C.5. Při tomto experimentu bylo předvypočítáno několik dalších rysů, které klasifikátor používá pro rozhodování. Jedná se o různé směrodatné odchylky, které mají poskytnout klasifikátoru lepší přehled o variabilitě provozu. Tyto dodatečné rysy jsou uvedeny v tabulce C.2 a jejich vliv na přesnost lze vidět na všech grafech s testováním vstupního thresholdu s hodnotou thresholdu 0. Přibližně bylo dosaženo zlepšení celkové přesnosti v rámci desetin procent.

Při reálném nasazení je důležitá vysoká míra metriky Precision, protože v klasifikaci síťových zařízení je nežádoucí mít klasifikovaná zařízení jako false positive, tedy označení profilu štítkem, přičemž by mu štítek být přiřazen neměl. Z tohoto důvodu bylo experimen-



Obrázek 7.3: Přehled množství síťových profilů, které jsou puštěny ke klasifikaci po aplikaci určité velikosti thresholdu v počtu **příchozích paketů**.

továno i s váhováním důležitosti klasifikačních tříd (tzv. penalizované klasifikační modely). Různé nastavení vyšší důležitosti predikovaného štítku ale nijak výrazněji metriku Precision nezlepšily, někdy došlo dokonce i ke zhoršení. Proto ve finální verzi klasifikátorů není váhování aplikováno. Jednou z výhod klasifikačního algoritmu Náhodný les je možnost získat přehled vah, které jsou přiřazeny jednotlivým klasifikačním rysům. Před nasazením v reálném provozu je tedy možné dodatečně u každého klasifikátoru odebrat nepoužívané rysy. Odstranění nepoužívaných rysů má především pozitivní vliv na rychlost klasifikace.

Jako poslední experiment byla vyzkoušena ternární klasifikace, při které klasifikátor rozlišoval 3 klasifikační třídy. První třída obsahovala profily síťových zařízení, které byly v anotaci zařízení anotovány pouze daným štítkem. Druhá třída reprezentovala také zkoumaný typ zařízení, ale zahrnovala síťové profily zařízení, která měly ve své anotací kromě zkoumaného typu zařízení i jiný typ zařízení. Cílem tedy bylo otestovat schopnost klasifikátoru správně rozeznat, že na daném zařízení běží pouze daná služba a žádná jiná služba. Pro většinu štítků bylo ale v datové sadě nedostatek takových zařízení, na kterých by běžela izolovaně pouze jedna zkoumaná služba a z výsledků nelze vyvodit přímé závěry.

7.3 Zhodnocení dosažených výsledků a možností dalšího rozvoje datové sady

V kapitole 4.1 byla jako jedna z hlavních možných překážek *host-based* statistické klasifikace zmíněn vliv počtu běžících služeb na daném zařízení, protože více služeb od sebe bývá složité oddělit v souhrnných statistikách. Z analýzy výsledků klasifikace na testovacích sadách jednotlivých klasifikátorů vyplývá, že při klasifikaci některých typů zařízení se

Štítek zařízení	Průměrný počet služeb při správné klasifikaci	Průměrný počet služeb při chybné klasifikaci
`database server`	4,1	5,2
`DNS server`	4,1	7,9
`file server`	4,2	3,9
`IoT`	4,2	5,2
`mail server`	4,1	4,9
`time server`	4,1	5,2
`web server`	4,3	3,6
`Windows`	4,3	3,1

Tabulka 7.4: Přehled vlivu počtu běžících služeb na zařízení na vliv úspěšnosti klasifikace.

klasifikaci daří lépe, pokud dané zařízení používá méně služeb, ale zároveň to není podmínkou úspěšnější klasifikace. Např. klasifikátor štítku `database server` správně klasifikoval zařízení, na kterých v průměru běží 4,1 služeb a nesprávně klasifikoval zařízení, na kterých v průměru běží 5,2 služeb. Naopak klasifikátor štítku `Windows` úspěšně klasifikoval zařízení, na kterých v průměru běží 4,3 služeb a chybně klasifikoval taková zařízení, na kterých běží průměrně 3,1 služeb. Přehled těchto statistik u všech klasifikovaných typů zařízení je zobrazen v tabulce 7.4.

Celkově bylo dosaženo přesností klasifikace jednotlivých typů zařízení větší než 90 % (kromě zařízení `web server`), přičemž průměrný počet běžících služeb na zařízení je 4,5 (při správné klasifikaci je to průměrně 4,2 služeb) a pomocí experimentů bylo předvedeno, že celková přesnost klasifikace se dá dále zlepšit aplikací vstupního thresholdu. Lze tedy usoudit, že i pro zařízení, které aktivně používá více služeb, je tento přístup použitelný. Stále lze ale očekávat, že při příliš vysokém počtu aktivně využívaných služeb bude přesnost klasifikace spíše klesat.

Při nasazení v reálném provozu lze menší přesnost klasifikace dále kompenzovat agregací výsledků klasifikace. Klasifikátory pro svoje predikce používají síťové profily zařízení, které jsou vypočteny z 1 hodiny provozu daného zařízení. Výčet běžících služeb zařízení se ale pravděpodobně mění s mnohem menší frekvencí. Výsledky získané z klasifikace na síťových profilech za jednu hodinu tak lze agregovat do časového rámce např. 1 dne. Pokud bude třeba štítek `DNS server` klasifikován u 20/24 síťových profilů zařízení daného dne, lze usoudit, že se s vysokou pravděpodobností jedná o zařízení provozující DNS server, i přes absenci štítku u 4 profilů.

Datová sada může být dále použita pro výzkum detekce anomálií v chování jednotlivých zařízení. Např. lze více prozkoumat právě 4 síťové profily, u kterých nedošlo ke klasifikaci štítku `DNS server`. Je možné mít předem vypočítaný dlouhodobý profil zařízení, od kterého lze poté kalkulovat Euklidovu vzdálenost[5] jednotlivých rysů, na základě které lze při vysoké vzdálenosti rysů označit síťový profil za anomálii v chování zařízení.

[5] https://en.wikipedia.org/wiki/Euclidean_distance

Síťové profily je možné také agregovat do větších časových oken, např. výpočet statistik ze síťového provozu za dva týdny nebo měsíc. V tomto scénáři lze použít stejné rysy, které byly použity při krátkých profilech, ale lze přidat i nové rysy, které lépe zachycují dlouhodobý charakter provozu zařízení. Některé další rysy použitelné pro rozhodování lze přidat v jakékoli fázi použití datové sady. Počet rysů je již sice poměrně vysoký, ale kdykoliv může dojít k odhalení nových vhodných rysů a některé méně užitečné lze odstranit, protože klasifikační algoritmus Náhodné lesy dokáže dobře detekovat rysy, které nemají vliv na klasifikaci.

Pro další rozvoj datové sady je klíčové nasazení více různých datových zdrojů nebo rozšíření sběru informací z již nasazených datových zdrojů. Příkladem datového zdroje s budoucím potenciálem pro rozšíření je systém ADiCT. Nyní je v systému množství informací využitelných pro anotaci datové sady síťových zařízení menší, ale zdroje informací, které proudí do systému ADiCT se neustále vyvíjí a testují. V blízké době, až budou zdroje informací více otestovány, bude tedy množství informací použitelných ze systému ADiCT výrazně větší.

Pro rozšíření datové sady se dále nabízí možnost přidání podpory IPv6 do automatizovaného anotátoru. Toto rozšíření je ale problematické pro několik datových zdrojů, především pro globální vyhledávače Shodan a Censys, které IP verze 6 nepodporují. Hlavním důvodem je příliš adresový prostor, který je nemožné celý v rozumném čase proskenovat. Lze ale využít různých heuristik[6], které redukují potřebný čas na proskenování adresového prostoru. Navíc v případě služeb Censys a Shodan by mohlo být při definici vstupního seznamu IP adres požadován od těchto služeb sken právě pouze takových IPv6 adres, které se nacházejí ve vstupním seznamu.

[6] https://tools.ietf.org/html/rfc5157 a https://www.internetsociety.org/blog/2015/02/ipv6-security-myth-4-ipv6-networks-are-too-big-to-scan/

Kapitola 8

Závěr

Monitorování počítačové sítě a udržování přehledu o zařízeních připojených do sítě je velmi důležité. Umožňuje správcům sítě detekovat anomálie v provozu i v konfiguraci, kterým je potřeba věnovat zvýšenou pozornost a je nutné je vyřešit. Pokud tak správce neučiní, může těchto anomálií zneužít útočník. Pokud má správce k dispozici přehled o zařízeních v síti, je pro něho snazší síť spravovat a také aplikovat jednotné bezpečnostní politiky.

Cílem této práce bylo vytvořit automatizovaný nástroj, který uživateli zjednoduší proces vytváření datových sad síťových zařízení, které mohou následně urychlit vývoj nových nebo stávajících metod strojového učení používaných pro klasifikaci síťových zařízení. Dalším úkolem byla demonstrace použitelnosti tohoto nástroje vytvořením datové sady pro klasifikaci síťových zařízení. Posledním bodem této práce bylo předvést využitelnost vytvořené datové sady aplikací vybrané metody strojového učení.

V první fázi této práce jsem zanalyzoval systém ADiCT sdružení Cesnet a jeho dostupné informace využitelné pro klasifikaci síťových zařízení. Následně byla provedena analýza různých externích zdrojů a jejich možností z pohledu informací, které poskytují o síťových zařízeních. Na analýzu datových zdrojů navazuje přehled existujících přístupů ke klasifikaci síťových zařízení s důrazem na metody klasifikující síťová zařízení pomocí statistického chování, které jsou vhodné pro aplikaci metod strojového učení. Poté byl vytvořen návrh využití datových zdrojů k vytvoření anotace datové sady síťových zařízení, z kterého vychází implementace automatizovaného anotátoru. Vytvořený automatizovaný anotátor byl následně použit pro anotaci zachyceného provozu síťových zařízení, čímž vznikla anotovaná datová sada 2 300 síťových zařízení. Datová sada byla poté použita k natrénování a validaci klasifikátorů několika různých typů zařízení, jejichž průměrná celková přesnost klasifikace se pohybuje okolo 93 %. V poslední části této práce byly zhodnoceny dosažené výsledky a možnosti dalšího rozvoje datové sady.

Vyvinutý anotační nástroj i výsledky klasifikace zařízení budou využity pro zvyšování zabezpečení sítě CESNET 2. Z pohledu dalšího rozvoje automatizovaného anotátoru je klíčové přidání dalších datových zdrojů nebo rozšíření sběru informací z již používaných zdrojů (např. systém ADiCT postupem času bude přidávat více informací o síťových zařízení). Další vývoj představeného přístupu klasifikace se může zabývat experimenty, které využívají statistiky síťového provozu zařízení z většího časového okna, než je 1 hodina, např. dva týdny nebo měsíc.

Literatura

[1] Bhuyan, M. H., Bhattacharyya, D. K. a Kalita, J. K. Surveying port scans and their detection methodologies. *The Computer Journal*. Oxford University Press. 2011, sv. 54, č. 10, s. 1565–1581.

[2] Bodenheim, R., Butts, J., Dunlap, S. a Mullins, B. Evaluation of the ability of the Shodan search engine to identify Internet-facing industrial control devices. *International Journal of Critical Infrastructure Protection*. 2014, sv. 7, č. 2, s. 114 – 123. DOI: https://doi.org/10.1016/j.ijcip.2014.03.001. ISSN 1874-5482. Dostupné z: http://www.sciencedirect.com/science/article/pii/S1874548214000213.

[3] Chen, Y., Lian, X., Yu, D., Lv, S., Hao, S. et al. Exploring Shodan From the Perspective of Industrial Control Systems. *IEEE Access*. IEEE. 2020, sv. 8, s. 75359–75369.

[4] Durumeric, Z., Adrian, D., Mirian, A., Bailey, M. a Halderman, J. A. A Search Engine Backed by Internet-Wide Scanning. In:. New York, NY, USA: Association for Computing Machinery, 2015, s. 542–553. CCS '15. DOI: 10.1145/2810103.2813703. ISBN 9781450338325. Dostupné z: https://doi.org/10.1145/2810103.2813703.

[5] Durumeric, Z., Wustrow, E. a Halderman, J. A. ZMap: Fast Internet-wide scanning and its security applications. In: *22nd {USENIX} Security Symposium ({USENIX} Security 13)*. 2013, s. 605–620.

[6] Fernández Delgado, M., Cernadas, E., Barro, S. a Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*. JMLR. org. 2014, sv. 15, č. 1, s. 3133–3181.

[7] Giza, D. A. *Draft Report for the Study of the Accuracy of WHOIS Registrant Contact Information*. 225 Friend St, Boston, MA 02114, Spojené státy: NORC at the University of Chicago, duben 2010. Dostupné z: https://www.icann.org/en/system/files/newsletters/whois-accuracy-study-17jan10-en.pdf.

[8] Haibo He, Yang Bai, Garcia, E. A. a Shutao Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 2008, s. 1322–1328. DOI: 10.1109/IJCNN.2008.4633969.

[9] Haraty, R. A. a Zantout, B. The TOR data communication system. *Journal of Communications and Networks*. IEEE. 2014, sv. 16, č. 4, s. 415–420.

[10] Jakalan, A., Gong, J. a Liu, S. Profiling IP hosts based on traffic behavior. In: IEEE. *2015 IEEE International Conference on Communication Software and Networks (ICCSN)*. 2015, s. 105–111.

[11] Komosnỳ, D., Vozňák, M. a Rehman, S. U. Location accuracy of commercial IP address geolocation databases. *Journal of Information Technology*. Kaunas University of Technology. 2017, sv. 46, č. 3, s. 333–344. DOI: 10.5755/j01.itc.46.3.14451.

[12] Kondo, T. S. a Mselle, L. J. Penetration testing with banner grabbers and packet sniffers. *Journal of Emerging Trends in computing and information sciences*. Citeseer. 2014, sv. 5, č. 4, s. 321–327.

[13] Koumar, J. *Automatické rozpoznávání síťových zařízení a jejich závislostí*. Praha, CZ, 2020. Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií. Dostupné z: https://dspace.cvut.cz/handle/10467/88277.

[14] Lee, S., Shin, S. a Roh, B. Abnormal Behavior-Based Detection of Shodan and Censys-Like Scanning. In: *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*. 2017, s. 1048–1052. DOI: 10.1109/ICUFN.2017.7993960.

[15] Liu, S., Foster, I., Savage, S., Voelker, G. M. a Saul, L. K. Who is .Com? Learning to Parse WHOIS Records. In: *Proceedings of the 2015 Internet Measurement Conference*. New York, NY, USA: Association for Computing Machinery, 2015, s. 369–380. IMC '15. DOI: 10.1145/2815675.2815693. ISBN 9781450338486. Dostupné z: https://doi.org/10.1145/2815675.2815693.

[16] Matherly, J. Complete guide to Shodan. *Shodan, LLC (2016-02-25)*. 2015, sv. 1.

[17] Moore, A., Zuev, D. a Crogan, M. *Discriminators for use in flow-based classification*. 2013.

[18] Nguyen, T. T. a Armitage, G. A survey of techniques for internet traffic classification using machine learning. *IEEE communications surveys & tutorials*. IEEE. 2008, sv. 10, č. 4, s. 56–76.

[19] Novák, P. *Identifikace podobných zařízení vzhledem k chování v počítačové síti*. Brno, CZ, 2020. Bakalářská práce. Masarykova univerzita, Fakulta informačních technologií. Dostupné z: https://is.muni.cz/th/gxehh/.

[20] Oshiro, T. M., Perez, P. S. a Baranauskas, J. A. How many trees in a random forest? In: Springer. *International workshop on machine learning and data mining in pattern recognition*. 2012, s. 154–168.

[21] Poese, I., Uhlig, S., Kaafar, M. A., Donnet, B. a Gueye, B. IP Geolocation Databases: Unreliable? *SIGCOMM Comput. Commun. Rev.* New York, NY, USA: Association for Computing Machinery. duben 2011, sv. 41, č. 2, s. 53–56. DOI: 10.1145/1971162.1971171. ISSN 0146-4833. Dostupné z: https://doi.org/10.1145/1971162.1971171.

[22] Shodan. *Shodan.io – The search engine for the Internet of Things* [online]. 2013 [cit. 2020-11-15]. Dostupné z: https://www.shodan.io/.

[23] Torabi, S., Boukhtouta, A., Assi, C. a Debbabi, M. Detecting Internet Abuse by Analyzing Passive DNS Traffic: A Survey of Implemented Systems. *IEEE Communications Surveys Tutorials*. 2018, sv. 20, č. 4, s. 3389–3415. DOI: 10.1109/COMST.2018.2849614.

[24] Wang, M., Hu, H. a Hu, G. A survey on traffic-behavioral profiling of network end-target. In: *Proceedings of the ACM Turing Celebration Conference-China*. 2019, s. 1–7.

[25] Weimer, F. Passive DNS replication. In: *FIRST conference on computer security incident*. 2005, s. 98.

[26] Xue, Y., Wang, D. a Zhang, L. Traffic classification: Issues and challenges. In: IEEE. *2013 International Conference on Computing, Networking and Communications (ICNC)*. 2013, s. 545–549.

[27] Yiu, T. *Understanding Random Forest*. 2019. [online], [vid. 4.3.2021]. Dostupné z: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

[28] Zaitsev, I. *The Best Format to Save Pandas Data*. 2019. [online], [vid. 12.3.2021]. Dostupné z: https://towardsdatascience.com/the-best-format-to-save-pandas-data-414dca023e0d.

Příloha A

Obsah přiloženého paměťového média - CD

V kořenovém adresáři se nachází adresáře s následujícím obsahem:

- **dataset** - obsahuje anonymizovaný jednodenní provoz síťových zařízení a jejich anotaci
- **dataset_annotator** - implementace automatizovaného anotátoru
- **documentation** - zdrojové latexové soubory technické zprávy včetně její finální podoby v PDF souboru
- **ml_experiments** - implementace klasifikačních experimentů - výpočet krátkodobých síťových profilů a trénování klasifikátorů

Příloha B

Seznam typů síťových zařízení

Tabulka B.1: Převodová tabulka převádějící služby na typ zařízení. Typ zařízení je štítek přiřazený síťovému zařízení a zahrnuté služby ukazují čísla portů služeb, které spadají pod daný štítek.

Typ zařízení (štítek)	**Zahrnuté služby (čísla portů)**	**Popis**
`Apple`	HAP (661), AFP (548), ...	Zařízení značky Apple
`authentication server`	Kerberos (88, 752, 753, ...), Radius (1645, 1646, ...), ...	Služby poskytující autentizaci
`BitTorrent peer`	BitTorrent(6881, 6882, ..., 6889)	Uzel sítě BitTorrent
`CMIP agent`	CMIP(164)	Zařízení využívající protokol CMIP
`CMIP manager`	CMIP(163)	Zařízení využívající protokol CMIP
`database server`	MySQL (3306), PostgreSQL(5432), ...	Databázový server
`DHCP server`	DHCP(67), DHCP(547), ...	Server služby DHCP
`DHCP client`	DHCP(68), DHCP(546), ...	Klient služby DHCP
`dictionary server`	LDAP(389), LDAPS(636), ...	Adresářový server
`DNS server`	DNS(53)	Server služby DNS
`end device`	DOOM(666), Garena(1513)	Koncové zařízení
`file server`	FTP (20, 21), TFTP (69), NFS (111), ...	Služby poskytující síťový přístup
Pokračování na další straně		

Tabulka B.1 – pokračování z předchozí strany

Typ zařízení (štítek)	Zahrnuté služby (čísla portů)	Popis
		k souborům
`HTTP manager`	HTTP-mgmt(280)	Management služby HTTP
`chat server`	bnetgame (1119), IRC (194), ...	Servery provozující chatovací služby
`ICS`	Modbus(502), DNP(20000), ...	Zařízení průmyslového kontrolní systém
`ID entity`	IDXP(603)	Detektor narušení sítě
`IoT`	HAP (661), Amazon Echo (4070), Google Chromecast (8009), ...	Protokoly IoT zařízení
`IP camera`	Dlink Cam (16119), ...	IP kamera
`Mac OS`	MAC-SRV-ADMIN (660), ...	Zařízení s operačním systémem Mac OS
`mail server`	SMTP(25), IMAP(143), ...	Poštovní server
`message broker`	AMQP(5672), Kafka(9092)	Distributor zpráv
`mobile device`	UMA(144), ...	Mobilní zařízení
`news server`	NNTP(119), NNTPS(563), ...	Zpravodajský server
`log server`	SYSLOG (514), RELP(20514), ...	Logovací server
`optical network device`	LMP(701)	Zařízení používající optické připojení k internetu
`password server`	APPLE-SASL(3659), Kerberos(752), ...	Server s hesly
`printer`	NPP (92), IPP (631), ...	Služby, které používají tiskárny
`proxy server`	SOCKS(1080), TPROXY(3345), ...	Proxy server

Pokračování na další straně

Tabulka B.1 – pokračování z předchozí strany

Typ zařízení (štítek)	Zahrnuté služby (čísla portů)	Popis
`RMC daemon`	RMC(657)	Monitorovací zařízení
`router`	BGP (179), RIP (520), ...	Router
`SNMP agent`	SNMP(161), AGENTX(705), ...	Zařízení monitorované protokolem SNMP
`storage device`	iSCSI(860), RSYNC(873), ...	Datové uložiště
`streaming server`	RTSP(554), RTMP(1935), ...	Server provozující stream videa
`time server`	NTP(123), TIMED(525), ...	Server pro synchronizaci času
`UNIX`	UTIME(519), rlogin(513), ...	Zařízení s Unixovým operačním systémem
`VoIP telephone`	H.323(1720), H.248(2944), ...	Síťový telefon
`VPN`	PPTP (1723), OpenVPN(1194), ...	Zařízení používající službu VPN
`web server`	HTTP (80, 8080), HTTPS (443, 8443)	Webové služby
`Windows`	SMB-NetBIOS(137), WINS(1512), ...	Zařízení s operačním systémem Windows
`wireless device`	AODV(654), TBRPF(712)	Bezdrátové zařízení

Příloha C

Dodatečné informace o datové sadě a k experimentům klasifikace

```
['unix', 'cisco ios', 'raspbian', 'windows server 2019 datacenter 17763',
 'windows', 'debian', 'centos', 'ubuntu', 'freebsd', 'win32']
```

Výpis C.1: Seznam všech detekovaných operačních systémů vyhledávači Shodan a Censys.

```
['mini_httpd', 'lighttpd', 'httpd', 'VNC', 'Cesar FTP', 'vsftpd', 'Coyote',
 'rdate', 'Postfix', 'Remote Desktop', 'OpenSSH', 'PRTG',
 'Microsoft IIS httpd', 'Sun SSH', 'Jetty', 'rsyncd',
 'Erlang Port Mapper Daemon', 'Mosquitto', 'HTTPAPI', 'InfluxDB',
 'Kojoney SSH honeypot', 'VMware Authentication Daemon', 'MiniServ',
 'OpenBSD', 'Microsoft SQL Server', 'BSD TCP Wrappers', 'Sendmail', '
 Cisco IOS', 'nginx', 'CerberusFTPServer', 'MQTT', 'Werkzeug httpd',
 'IIS', 'Dovecot', 'ProFTPD', 'Apache httpd', 'GlassFish']
```

Výpis C.2: Seznam všech detekovaných aplikací vyhledávači Shodan a Censys.

```
[0, 22, 53, 80, 123, 443, 445, 993, 8291, 3389, 3390, 3489, 6000, 8080,
 8728, 30303, 32029]
```

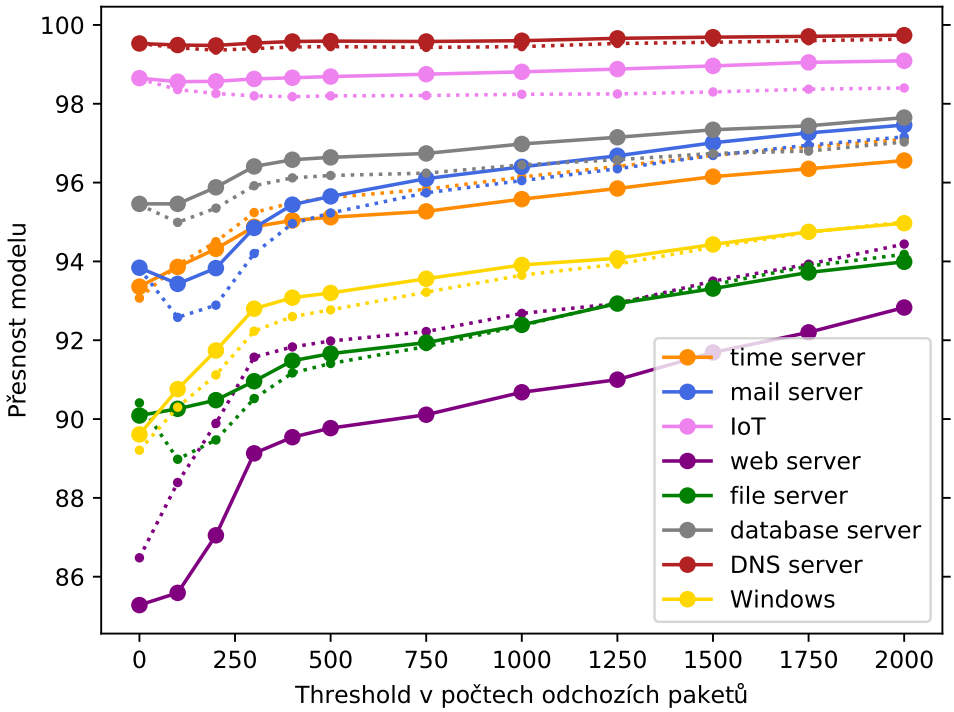
Výpis C.3: Seznam nejvíce používaných portů v síťové komunikaci vytvořené datové sady.

Název atributu	**Popis atributu**
DST_IP	Cílová IP adresa
SRC_IP	Zdrojová IP adresa
DST_PORT	Cílový port toku
SRC_PORT	Zdrojový port toku
BYTES	Počet přenesených bytů
PACKETS	Počet přenesených paketů
TIME_FIRST	Datum a čas počátku toku
TIME_LAST	Datum a čas konce toku
LINK_BIT_FIELD	Identifikace linky
DIR_BIT_FIELD	Identifikace směru na lince
PROTOCOL	Použitý IP protokol
TCP_FLAGS	Příznaky TCP protokolu
TOS	Typ použité služby
TTL	Zbývající počet skoků

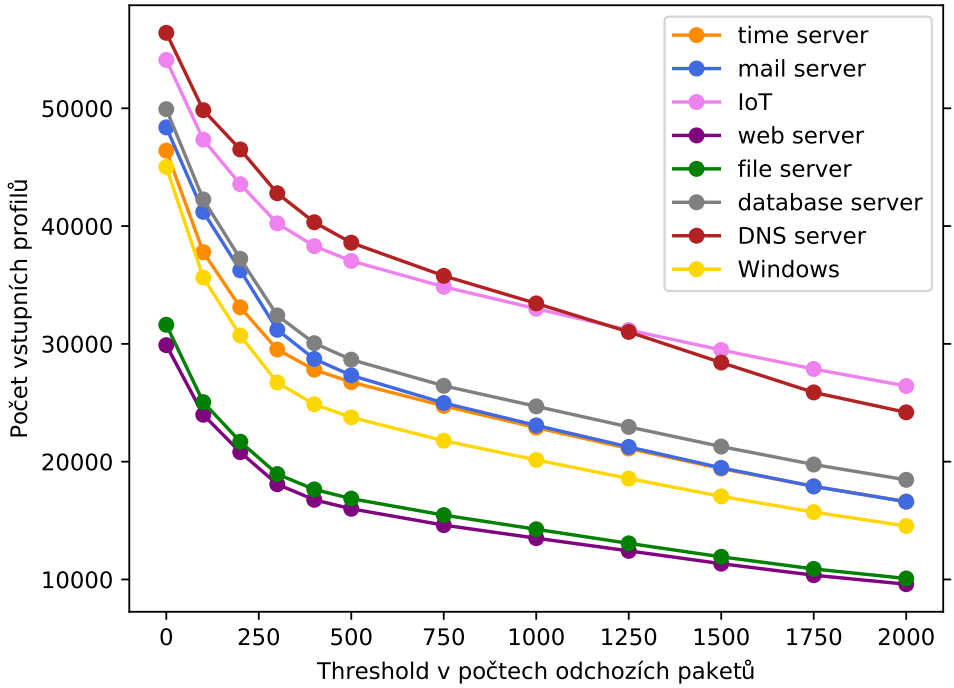
Tabulka C.1: Přehled jednotlivých položek základních Flow dat, které jsou součástí vytvořené datové sady.

Popis statistiky	**Důvod výpočtu**
Směrodatná odchylka počtu přenesených paketů	Ukazuje rozpětí v počtu paketů na tok
Směrodatná odchylka počtu přenesených bytů	Ukazuje rozpětí velikosti datových toků hosta
Směrodatná odchylka doby trvání toku	Zobrazuje velikost různorodosti v trvání toku
Směrodatná odchylka velikosti paketu	Zobrazuje různorodost ve velikosti paketu
Průměrný počet provedených toků protihosta	Informuje o chování komunikujících hostů
Průměrný počet přenesených bytů protihosta	Informuje o chování komunikujících hostů
Průměrný počet přenesených paketů protihosta	Informuje o chování komunikujících hostů

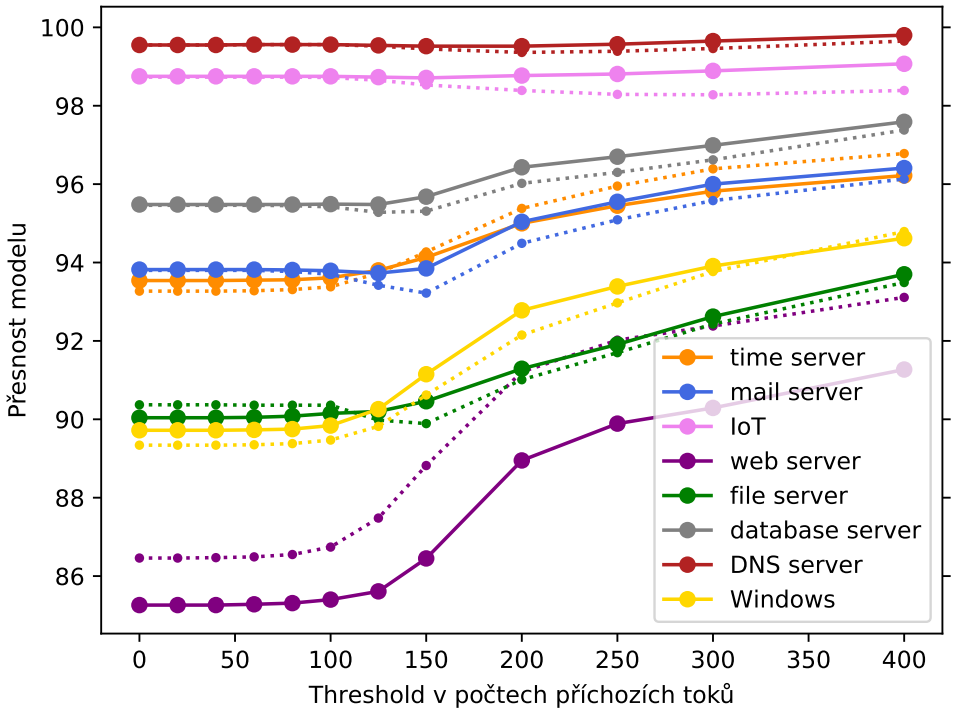
Tabulka C.2: Dodatečné rysy použité pro rozhodování klasifikátoru.



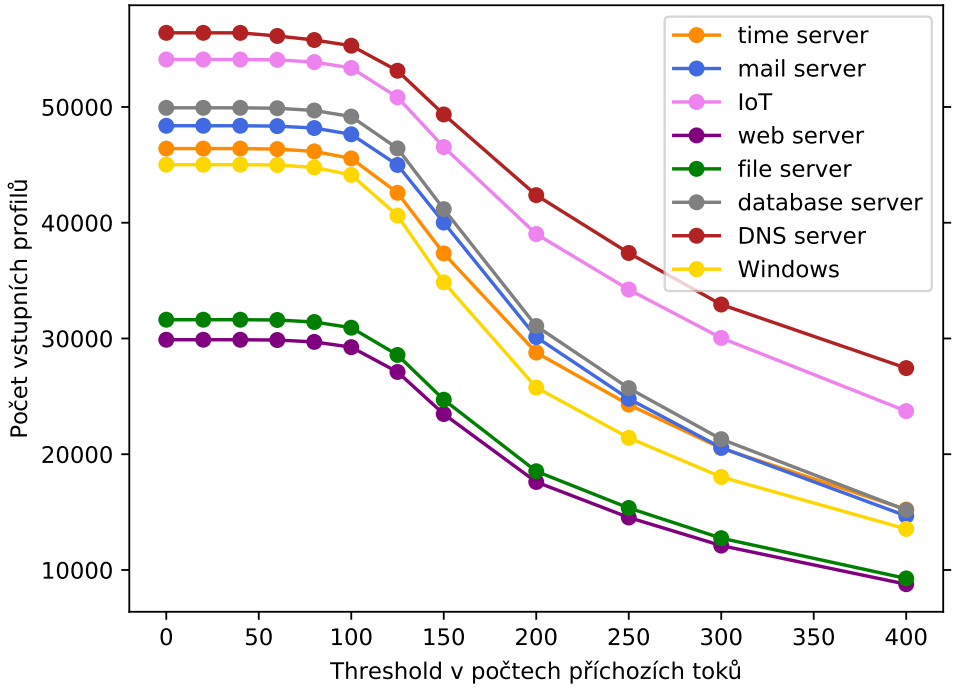
Obrázek C.1: Přehled přesnosti jednotlivých klasifikátorů při aplikaci určitého thresholdu v počtu **odchozích paketů**. Plná čára klasifikátoru představuje celkovou přesnost modelu, přerušovaná čára zobrazuje metriku *F1-score* predikovaného štítku (typu zařízení).



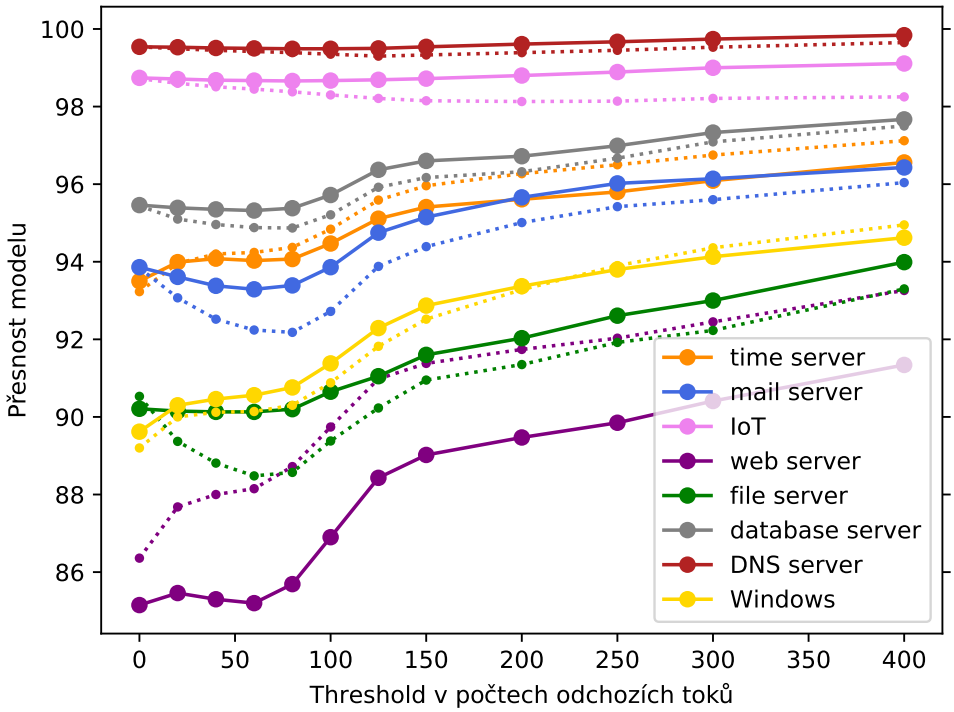
Obrázek C.2: Přehled množství síťových profilů, které jsou puštěny ke klasifikaci po aplikaci určité velikosti thresholdu v počtu **odchozích paketů**.



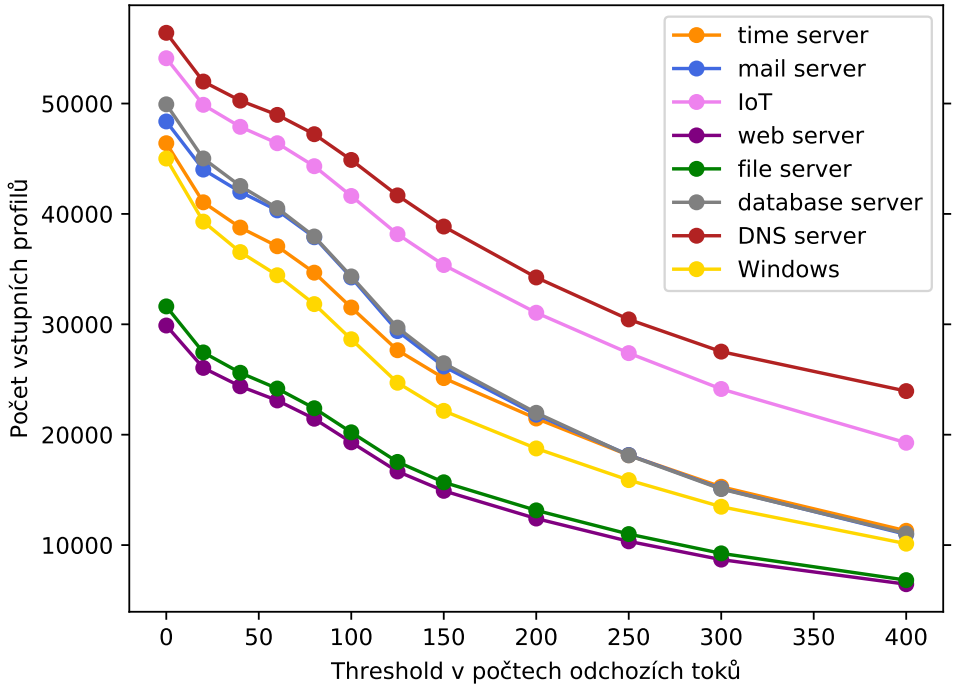
Obrázek C.3: Přehled přesnosti jednotlivých klasifikátorů při aplikaci určitého thresholdu v počtu **příchozích toků**. Plná čára klasifikátoru představuje celkovou přesnost modelu, přerušovaná čára zobrazuje metriku *F1-score* predikovaného štítku (typu zařízení).



Obrázek C.4: Přehled množství síťových profilů, které jsou puštěny ke klasifikaci po aplikaci určité velikosti thresholdu v počtu **příchozích toků**.



Obrázek C.5: Přehled přesnosti jednotlivých klasifikátorů při aplikaci určitého thresholdu v počtu **odchozích toků**. Plná čára klasifikátoru představuje celkovou přesnost modelu, přerušovaná čára zobrazuje metriku *F1-score* predikovaného štítku (typu zařízení).



Obrázek C.6: Přehled množství síťových profilů, které jsou puštěny ke klasifikaci po aplikaci určité velikosti thresholdu v počtu **odchozích toků**.