Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

DIPLOMOVÁ PRÁCE

Automatická detekce DDoS útoku
využívající strojové učení

2025

Bc. Veronika Pavlíková

Vedoucí práce:
RNDr. Martin Trnečka, Ph.D.

Studijní program: Aplikovaná informatika,
Specializace: Vývoj software

Bibliografické údaje

Autor:	Bc. Veronika Pavlíková
Název práce:	Automatická detekce DDoS útoku využívající strojové učení
Typ práce:	diplomová práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2025
Studijní program:	Aplikovaná informatika, Specializace: Vývoj software
Vedoucí práce:	RNDr. Martin Trnečka, Ph.D.
Počet stran:	41
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Bc. Veronika Pavlíková
Title:	Automatic DDoS attack detection using machine learning
Thesis type:	master thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2025
Study program:	Applied Computer Science, Specialization: Software Development
Supervisor:	RNDr. Martin Trnečka, Ph.D.
Page count:	41
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Tato práce se zaměřuje na vývoj prototypu nástroje pro automatickou detekci DDoS útoků s využitím metod strojového učení. Cílem prototypu je experimentálně ověřit možnosti automatické detekce těchto útoků v reálných síťových podmínkách.

Synopsis

This work discusses the development of a prototype tool for automatic detection of DDoS attacks using machine learning methods. The aim of the prototype is to experimentally verify the possibilities of automatic detection of these attacks in real network conditions.

Klíčová slova: Distributed Denial of Service; strojové učení; Electron.js

Keywords: Distributed Denial of Service; machine learning; Electron.js

Ráda bych tímto poděkovala především svému vedoucímu práce RNDr. Martinu Trnečkovi, Ph.D. za odborné vedení, cenné rady, podporu a trpělivost v průběhu tvorby této práce. Velké poděkování patří také mé rodině za podporu a motivaci v náročných chvílích. Bez těchto lidí by práce nikdy nemohla vzniknout.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1 Úvod

V současné době, kdy internetové služby hrají velkou roli ve fungování různých institucí a podniků, představují kybernetické hrozby stále větší nebezpečí. Jejich počet každoročně roste. Pouze na území České republiky došlo od roku 2022 k jejich zdvojnásobení. Za uplynulý rok 2024 evidoval Národní úřad pro kybernetickou a informační bezpečnost (NÚKIB) rekordních 268 kybernetických hrozeb. Nejvýznamnější hrozbou, tvořící téměř 50 % všech útoků, byly útoky na dostupnost služeb, známé jako Distributed Denial of Service (DDoS) [1]. Motivace útočníků se liší. Může jít o konkurenční boj, kdy firmy využívají DDoS útoky k poškození konkurence a snížení jejich dostupnosti pro zákazníky. V jiných případech může být útok motivován aktivismem, takzvaným hacktivismem, kdy útočníci cílí na organizace, které považují za problematické. Další kategorií jsou útoky sloužící k vydírání, kdy útočníci požadují výkupné za ukončení útoku, nebo k demonstraci síly, kdy skupiny hackerů testují své schopnosti či se snaží získat uznání v komunitě. Důsledky těchto útoků se liší podle jejich intenzity, délky trvání a povahy cíle. Krátkodobé útoky mohou způsobit pouze dočasné zpomalení služeb, zatímco rozsáhlé útoky mohou vést k úplné nefunkčnosti online služeb trvající několik hodin až dní. Finanční ztráty mohou být značné – podniky přicházejí o příjmy kvůli nedostupnosti služeb, musí investovat do posílení obrany a řešit poškození své reputace. V extrémních případech mohou DDoS útoky narušit fungování kritických systémů, což může mít dopad na bezpečnost i běžný chod společnosti. Dopadům takového útoku čelilo například Estonsko, které se v roce 2007 stalo obětí masivních DDoS útoků namířených proti státu. Tyto útoky cílily především na vládní weby, média a bankovní sektor a některé z nich trvaly až několik týdnů. V jejich důsledku došlo nejen k výpadku vládních webů, ale i k omezení funkčnosti bankomatů a internetového bankovnictví [2]. Přestože Česká republika doposud nečelila takto rozsáhlým útokům, zaznamenala také DDoS útoky na vládní weby a organizace. Relativně nedávným příkladem je útok z roku 2023, který zasáhl weby Ministertsva vnitra, Policie ČR, Hasičského záchranného sboru ČR a Letiště Václava Havla. Následkem byla dočasná nedostupnost těchto služeb. K vážnějšímu narušení provozu ale nedošlo [3].

1.1 Nástroj pro detekci DDoS

DDoS útoky jsou velmi různorodé a neustále se vyvíjejí, což ztěžuje jejich včasné odhalení. Útočníci navíc mohou v průběhu útoku měnit své taktiky, což detekci přicházejícího DDoS útoku ještě více komplikuje. I když vhodná prevence může pomoci eliminovat mnoho potenciálních útoků, je nezbytné mít připravený plán pro situace, kdy prevence selže. Jedním z efektivních přístupů je monitorování síťové komunikace a identifikace anomálií, které mohou naznačovat probíhající útok. Vzhledem k technickému pokroku a složitosti útoků však není reálné, aby tuto kontrolu prováděl člověk. To mě přivedlo k myšlence vyvinout prototyp nástroje pro automatickou detekci DDoS útoků, který celý proces automatizuje.

Cílem této práce je ověřit teoretické možnosti detekce DDoS útoků pomocí metod strojového učení. Vytvořený nástroj není finálním řešením připraveným k okamžitému nasazení v praxi. Jedná se o experimentální prototyp, který umožňuje testování přesnosti detekce DDoS útoků v reálných podmínkách.

1.1.1 Vlastnosti nástroje

Hlavním cílem vyvíjeného nástroje je včasná a přesná detekce počínajícího DDoS útoku. Vzhledem k tomu, že existuje několik různých typů této kybernetické hrozby (více v kapitole 2), byl kladen důraz na to, aby nástroj dokázal spolehlivě rozpoznat většinu jejích variant.[1] Po úspěšné detekci má nástroj okamžitě informovat uživatele o potenciální hrozbě. Výsledný nástroj proto poskytuje následující funkce:

1. Monitorování síťové komunikace.
2. Průběžná a automatická detekce DDoS útoků.
3. Jasné a včasné informování uživatele o přicházející hrozbě.

Zbytek práce je strukturován následovně. Druhá kapitola se věnuje obecnému popisu DDoS útoků a jejich jednotlivým typům. Třetí kapitola představuje dataset použitý pro trénování modelů strojového učení, jeho strukturu a atributy, společně s jeho preprocessingem. Ve čtvrté kapitole je popsána aplikace různých metod strojového učení na zpracovaná data. Pátá kapitola obsahuje programátorskou a uživatelskou dokumentaci k vyvinutému nástroji. Šestá kapitola popisuje testování vyvinutého nástroje. Práci uzavírá poslední kapitola, která shrnuje dosažené výsledky.

[1] Kvůli technickým důvodům ale nemohly být všechny varianty řádně otestovány.

2 Distributed Denial of Service

Kybernetické útoky typu Distributed Denial of Service (DDoS) patří mezi nejčastější hrozby narušující dostupnost online služeb. Jedná se o variantu útoků Denial of Service (DoS), jejichž cílem je přetížení cílového systému tak, aby nebyl schopen odpovídat na legitimní požadavky. Zatímco DoS útoku pocházejí z jediného zdroje, DDoS útoky jsou distribuované a k provedení proto využívají velké množství různých zařízení. DDoS útoky jsou specifické tím, že generují obrovský objem síťového provozu nebo zneužívají zranitelnosti protokolů k zahlcení cílového systému, čímž způsobují zpomalení nebo úplný výpadek služeb. Kvůli rozvoji botnetů a stále sofistikovanějším technikám útočníků se DDoS útoky staly nejen častějšími, ale také obtížněji odhalitelnými.

DDoS útoky mají více způsobů provedení, přičemž se obvykle dělí do dvou hlavních kategorií, a to na *reflection* útoky a *exploitation* útoky [4].

2.1 Reflection útoky

Reflection útoky jsou typem DDoS útoků, které zneužívají jiné počítače, typicky servery, také známé pod pojmem reflektory, k zahlcení oběti. Útočníci vytvářejí falešné požadavky na reflektory a podvrhují zdrojovou IP adresu tak, aby odpovědi směřovaly na oběť. Tento mechanismus způsobí, že je oběť zahlcena odpověďmi z několika různých serverů, což vede k jejímu přetížení nebo úplnému výpadku. Tyto útoky se obvykle provádějí prostřednictvím protokolů aplikační vrstvy, které využívají transportní protokoly TCP nebo UDP, případně jejich kombinaci. Každý útok cílí na konkrétní službu a zneužívá odpovídající aplikační protokol, podle kterého bývá často pojmenován. Obecný scénář reflection útoku je následovný:

1. Útočník odešle síťový požadavek na reflektor a jako zdrojovou IP adresu nastaví adresu oběti.
2. Reflektor odpoví typicky velkým objemem dat, která pošle na adresu oběti.
3. Oběť je zahlcena odpověďmi z jednoho nebo více reflektorů, což vede k jejímu přetížení nebo výpadku.

Jelikož reflection útoky probíhají vždy podle stejného scénáře, budou v následující části jednotlivé varianty těchto útoků, protokoly, které využívají, a způsob jejich zneužití. Ve všech případech je výsledkem zahlcení oběti.

2.1.1 MSSQL

Microsoft SQL Server Resolution Protocol je protokol určený k získání seznamu dostupných databázových instancí na Microsoft SQL serverech. Tento protokol může být zneužit k provedení MSSQL DDoS útoku. Útočník odešle požadavek na SQL server s podvrženou IP adresou. Server následně odpoví seznamem instancí,

který je odeslán na adresu oběti. Tato odpověď může být výrazně větší než původní požadavek, obzvlášť pokud server obsahuje více databázových instancí.

2.1.2 SSDP

Simple Service Discovery Protocol (SSDP) je síťový protokol, který se používá pro detekci zařízení a komunikaci mezi nimi v lokální síti. SSDP využívá primárně multicast zprávy pro identifikaci zařízení, která podporují tento protokol. Typické využití je při připojování nových zařízení do sítě nebo při detekci zařízení, která již v síti jsou. Útočníci mohou tento protokol zneužít k DDoS útoku tak, že vygenerují multicast požadavek směřující na SSDP servery v síti, které odpovídají velkým množstvím dat.

2.1.3 DNS

Domain Name System (DNS) je protokol sloužící k překladu doménových jmen na IP adresy. Funguje na principu klient-server architektury. Při překladu doménového jména na IP adresu klient kontaktuje DNS resolver, který vyhledá odpovídající IP adresu na příslušných serverech a vrátí ji klientovi. Při DNS útoku[2] jsou k zahlcení oběti použity DNS resolvery. Útočník postupuje obvykle tak, že na několik DNS resolverů zároveň rozešle požadavky s podvrženou IP adresou. Každý požadavek v sobě obsahuje speciální argument, například `ANY`, který má vrátit seznam všech dostupných záznamů pro doménu, jenž způsobí, že DNS resolver vygeneruje velkou odpověď.

2.1.4 LDAP

Lightweight Directory Access Protocol (LDAP) je protokol, který slouží k práci s různými adresářovými službami. Typicky se používá pro správu informací o uživatelích. Útočník tuto službu může opět zneužít tak, že na LDAP server pošle dotaz, který má za cíl vygenerovat co největší odpověď. Příkladem by byl dotaz `search`, který vrátí rozsáhlá data z adresáře.

2.1.5 NetBIOS

Network Basic Input/Output System (NetBIOS) je síťová služba umožňující komunikaci mezi zařízeními v rámci lokální sítě. Přestože je dnes považována za zastaralou, stále se v některých sítích používá a pokud je nesprávně nakonfigurována nebo veřejně dostupná, může být zneužita k DDoS útoku. Při útoku útočník odešle na server dotaz, například na NetBIOS jména, která jsou přiřazena každému zařízení v síti. Tato jména mohou mít délku až 16 znaků a obsahují identifikátor služby nebo zařízení.

[2]Běžně také známým pod pojmem DNS amplifikační útok.

2.1.6 SNMP

Simple Network Management Protocol (SNMP) je síťový protokol, který je určen pro správu a monitorování zařízení připojených k síti, jako jsou směrovače, routery nebo i tiskárny. Útočníci na server zasílají speciální požadavky, například `GETBULK`, které generují velmi rozsáhlé odpovědi.

2.1.7 Portmap

Portmap je služba běžící na portu 111, která umožňuje klientům zjistit, jaké služby běží na jednotlivých portech daného serveru. Útočníci mohou tuto službu zneužít tím, že najdou zranitelný server s otevřeným portem 111 a pošlou na něj požadavek s podvrženou IP adresou. Server následně odpovídá velkým množstvím dat.

2.1.8 NTP

Network Time Protocol (NTP) je protokol používaný pro synchronizaci času mezi různými uzly v síti. V bežném scénáři se klient dotáže NTP serveru na jeho aktuální čas a na základě odpovědi si synchronizuje své hodiny. NTP server umí kromě poskytnutí času také vrátit seznam posledních 600 a více požadavků, které na něj byly poslány. Pro získání tohoto seznamu je nutné poslat takzvaný `monlist` požadavek, kterého mohou útočníci velmi snadno zneužít.

2.1.9 TFTP

Trivial File Transfer Protocol (TFTP) je jednoduchý protokol původně určený pro přenos souborů nebo konfigurací síťových zařízení. Právě díky své jednoduchosti a chybějícímu zabezpečení jej lze snadno zneužít pro DDoS útok. Útočník zasílá požadavky s podvrženou IP adresou na více TFTP serverů a opakovaně je žádá o soubory, které ale nemusí vůbec existovat. Servery odpovědi posílají oběti a postupně ji vytěžují.

2.2 Exploitation útoky

Při exploitation útocích se útočníci zaměřují na zneužívání slabin v síťových protokolech, které následně využívají k zahlcení oběti. K provedení těchto útoků se opět používají protokoly TCP a UDP, přičemž mezi nejznámější útoky z této kategorie patří SYN Flood, UDP Flood a UDP Lag útoky.

2.2.1 SYN Flood

SYN Flood útok zneužívá třífázového handshake protokolu TCP. Za normálních okolností probíhá spojení klienta se serverem pomocí protokolu TCP následovně:

1. Klient odešle na server TCP segment s příznakem `SYN`.

2. Server odpoví segmentem s příznaky `SYN` a `ACK`.

3. Klient odpoví segmentem s příznakem `ACK`, čímž je handshake ukončen.

Při SYN Flood útoku je ale třetí krok zcela vynechán. Útočník posílá velké množství segmentů s příznakem SYN na cílový server (oběť), ale nikdy neodpoví na segmenty s příznaky `SYN` a `ACK`. Server si udržuje neúplná spojení a postupně mu docházejí systémové prostředky, což vede k odmítnutí legitimních připojení.

2.2.2 UDP Flood

UDP Flood útok zneužívá bezstavový protokol UDP, který na rozdíl od TCP nevyžaduje navázání spojení. Útočníci zneužívají způsobu, jakým server zpracovává příchozí UDP datagramy:

1. Server zkontroluje, že na požadovaném portu běží aplikace, která přijímá požadavky.

2. Pokud na daném portu žádná aplikace neběží, vygeneruje ICMP zprávu a pošle ji klientovi.

Při UDP Flood útoku útočníci generují velké množství UDP datagramů s náhodnými porty a posílají je na server. Ten musí každý datagram zpracovat a ve většině případů odpovědět ICMP zprávou. Kvůli tomu dojde brzy k vyčerpání jeho výpočetních zdrojů a odmítnutí služby legitimním klientům.

2.2.3 UDP Lag

UDP Lag je útok zaměřený na narušení komunikace mezi klientem a serverem, což způsobuje vysokou latenci, nestabilitu nebo úplné přerušení spojení. Tento typ útoku se často využívá v online hrách, kde útočník (obvykle hráč) manipuluje s latencí, aby získal neférovou výhodu. Útok má v podstatě dvě podoby. V první variantě využívá útočník software, který běží v síti a zabírá šířku pásma ostatním hráčům s cílem je zpomalit. Druhá varianta využívá hardwarového zařízení známého jako lag switch, které přeruší útočníkovo spojení, čímž dojde k desynchronizaci mezi hráči a útočníkovi po opětovném připojení poskytne nespravedlivou výhodu.

Existuje mnoho různých typů DDoS útoků [5]. Jejich seznam přitom není nikdy kompletní, jelikož útočníci neustále vyvíjejí nové techniky a varianty. Tato kapitola představila pouze několik nejznámějších útoků, které jsou zároveň součástí datasetu použitého v této práci.

3 Data

Při vývoji metody pro analýzu a detekci probíhajícího DDoS útoku, využívající strojové učení, je nutné pracovat s kvalitními daty. Výběr vhodného datasetu byl proto zásadním krokem této práce. Bylo klíčové, aby dataset obsahoval co nejrealističtější síťový provoz a zároveň dostatečně reprezentativní vzorky různých typů DDoS útoků. V minulosti bylo vytvořeno mnoho datasetů určených pro detekci DDoS útoků, avšak ne všechny splňují požadavky na aktuálnost, kvalitu a variabilitu útoků. Proto bylo před finálním výběrem nutné provést důkladnou analýzu dostupných možností.

3.1 Analýza datasetů

Při výběru datasetu hrálo klíčovou roli to, aby byl veřejně dostupný a obsahoval reálný síťový provoz s větším počtem různých typů DDoS útoků. Tři zvažované datasety byly DARPA 1999 [6], KDD Cup 1999 [7] a CIC-DDoS2019 dataset [8].

3.1.1 DARPA 1999

DARPA 1999 Intrusion Detection Evaluation Dataset patří mezi nejznámější datasety používané pro detekci neobvyklých aktivit v síťové komunikaci. Byl vytovřen MIT Lincoln Laboratory ve spolupráci s Defense Advanced Research Projects Agency (DARPA) a Air Force Research Laboratory (AFRL/SNHS). Dataset obsahuje několik týdnů síťové komunikace a kromě *benigní*[3] komunikace obsahuje i několik útoků, mezi které patří i DDoS. Přestože se jedná o velmi známý a rozsáhlý dataset, dnes je považován za zastaralý, a to hlavně kvůli tomu, že neodpovídá soudobému reálnému provozu v počítačové síti.

3.1.2 KDD Cup 1999

Dalším zvažovaným datasetem byl dataset KDD Cup 1999, který byl vytvořen pro stejnojmennou soutěž zaměřenou na detekci síťových útoků. Cílem této soutěže bylo vytvořit model, který by uměl rozlišovat mezi benigní a škodlivou komunikací. KDD Cup 1999 dataset vychází z datasetu DARPA 1998. Narozdíl od něj je ale předzpracovaný a díky tomu je vhodný pro strojové účení. I přes jeho rozsáhlost a dobré zpracování se dnes jedná o zastaralý dataset, který nereflektuje moderní DDoS útoky.

3.1.3 CIC-DDoS2019 dataset

Posledním zvažovaným datasetem byl CIC-DDoS2019 dataset. CIC-DDOS2019 je veřejně dostupný dataset vytvořený Canadian Institute for Cybersecurity

[3]Benigní (z latinského benignus, což znamená dobrý) označuje v kontextu síťové komunikace normální neškodnou komunikaci, která neohrožuje bezpečnost systému a nevede k jeho poškození nebo přetížení.

(CIC). Tento dataset obsahuje realistickou síťovou komunikaci zaměřenou na DDoS útoky. Byl navržen tak, aby obsahoval co nejrealističtější síťový provoz, který odpovídá běžné benigní komunikaci. Kromě toho obsahuje několik moderních DDoS útoků, které byly podrobněji popsány v kapitole 2.

Z výše uvedených důvodů byl pro tuto práci vybrán CIC-DDoS2019 dataset. Tomuto datasetu se bude věnovat následující podkapitola.

3.2 Popis a struktura zvoleného datasetu

CIC-DDoS2019 dataset poskytuje realistická data pro výzkum a testování systémů zaměřených na detekci a mitigaci DDoS útoků. Pro simulaci benigní komunikace bylo simulováno chování 25 uživatelů, kteří využívají různé běžně používané protokoly, jako jsou HTTP, HTTPS, FTP,[4] SSH a e-mailové protokoly. Tato data byla zaznamenána v kontrolovaném testovacím prostředí. Strukturu tohoto prostředí lze podrobněji vidět na obrázku 1.

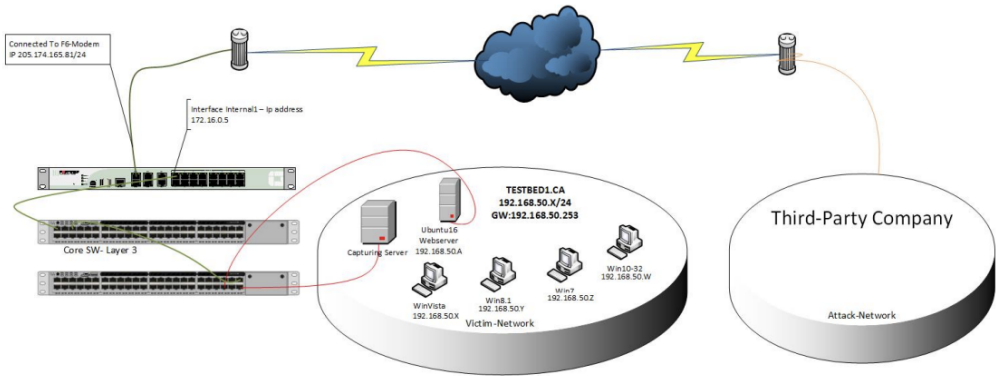


Figure 2: Testbed Architecture

Obrázek 1: Struktura testovacího prostředí [8].

Dataset sestává z komunikace zachycené v průběhu dvou dnů, přičemž první den je považován za trénovací a druhý za testovací. V průběhu každého dne bylo provedeno několik DDoS útoků. V průběhu trénovacího dne to bylo 12 útoků: NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP Flood, UDP Lag, WebDDoS,[5] SYN Flood a TFTP. V průběhu testovacího dne bylo provedeno 7 útoků: Portmap,[6] NetBIOS, LDAP, MSSQL, UDP Flood, UDP Lag a SYN Flood.

[4] V době psaní této práce se FTP protokol téměř nepoužívá.

[5] WebDDoS je označení pro kategorii DDoS útoků, které mají za cíl omezit webové aplikace.

[6] Útok Portmap byl proveden pouze v testovacím dni a proto bude pro vyvíjenou metodu neznámý.

Dataset se skládá ze dvou samostatných datových sad: PCAP[7] souborů s nezpracovanými síťovými daty a CSV souborů s již analyzovanými charakteristikami síťového provozu. Obě sady jsou rozděleny do dvou celků podle dnů, tedy do dne trénovacího a testovacího. V první datové sadě obsahují oba dny několik PCAP souborů se surovými daty. Druhá sada, tvořená CSV soubory, je pro každý den dále rozdělena podle časových úseků, ve kterých probíhaly jednotlivé útoky. Tyto CSV soubory byly vytvořeny z první datové sady pomocí nástroje CICflowMeter-V3 [9], který data analyzoval a extrahoval z nich více než 80 síťových charakteristik (features).

Pro potřeby této práce byla využita právě analyzovaná data ve formátu CSV, která jsou vhodná pro přímé použití ve strojovém učení.

3.3 Zpracování dat

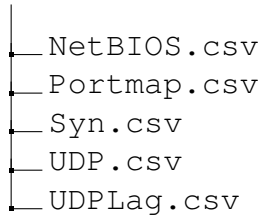
Před samotným použitím strojového učení bylo nutné data dále zpracovat. Toto zpracování zahrnovalo sloučení souborů, redukci dimenze, čištění dat, výběr relevantních charakteristik a další transformace nezbytné pro efektivní trénování modelů.

3.3.1 Sloučení souborů

CSV verze datasetu obsahuje dva adresáře s několika soubory, které bylo nutné sloučit pro získání finálních dat. Záznamy z CSV souborů ze složky `01-12` odpovídající trénovacímu dni byly použity k vytvoření trénovací množiny. Soubory ze složky `03-11` odpovídající testovacímu dni tvoří množinu testovací. Adresářová struktura datasetu byla po jeho stažení následující:

```
CSV/
├── 01-12/
│   ├── DrDos_DNS.csv
│   ├── DrDos_LDAP.csv
│   ├── DrDos_MSSQL.csv
│   ├── DrDos_NetBIOS.csv
│   ├── DrDos_NTP.csv
│   ├── DrDos_SNMP.csv
│   ├── DrDos_SSDP.csv
│   ├── DrDos_UDP.csv
│   ├── SYN.csv
│   ├── TFTP.csv
│   └── UDPLag.csv
└── 03-11/
    ├── LDAP.csv
    └── MSSQL.csv
```

[7] PCAP je souborový formát používaný k ukládání paketů.



Výsledné množiny měly vzhledem ke zpracování nástrojem CICFlowMeter-V3 88 sloupců. Jejich kompletní seznam je k dispozici v příloze A. Trénovací množina sestávala z 50 063 112 řádků, testovací množina z 20 364 525 řádků, přičemž každý řádek reprezentoval jeden síťový tok.[8]

3.3.2 Vyvažování dat

Prvním krokem po sloučení dat bylo prozkoumání sloupce *Label* s cílem zkontrolovat a případně vyvážit poměr nejen mezi jednotlivými útoky, ale také mezi DDoS a benigní komunikací.

Počet toků patřících k jednotlivým typům útoků se v obou množinách lišil. V trénovací množině se pohyboval mezi jedním a pěti miliony. Výjimku tvořil pouze TFTP útok s více než dvaceti miliony toky. V testovací množině byly počty toků u jednotlivých útoků podobné. Vzhledem k obrovské velikosti TFTP útoku bylo nutné jeho dimenzi snížit. Optimální se jevilo snížení na 20 % jeho původní velikosti, aby se zachovala určitá vyváženost mezi útoky.

Poměr mezi DDoS a benigními toky byl v obou množinách výrazně nevyvážený. Benigní komunikace tvořila v trénovací množině pouze 0,11 % a v testovací 0,28 %. Pro natrénování modelů a jejich schopnost správně rozeznat normální a DDoS komunikaci bylo nutné tento poměr upravit. Byl zvolen poměr 1:3 (benigní vs. DDoS), aby měly modely dostatek vzorků z obou tříd.

Původně sloučená data ze souborů tedy nebyla použitelná kvůli výše zmíněným nevyváženostem a musela být znovu sloučena tak, aby respektovala zohledněné faktory. Byly vytvořeny nové množiny, které byly následně použity v dalších úpravách.

3.3.3 Výběr vhodných atributů

Po vytvoření vyvážené trénovací a testovací množiny bylo nutné prozkoumat a odstranit nepotřebné atributy. Dále bylo třeba data vyčistit od chybných nebo chybějících hodnot. Analýza atributů byla provedena na trénovací množině a veškeré úpravy následně aplikovány i na množinu testovací.

Nejprve bylo nutné zjistit, které atributy obsahují chybějící nebo nulové hodnoty. Po prozkoumání dat bylo zjištěno, že v trénovací množině má 12 atributů nulové hodnoty ve všech řádcích. Konkrétně se jednalo o atributy: *Bwd PSH Flags*, *Fwd URG Flags*, *FIN Flag Count*, *PSH Flag Count*, *ECE Flag Count*,

[8] Síťový tok je soubor paketů, které jsou přenášeny mezi dvěma zařízeními v síti a sdílejí společné charakteristiky, jako jsou zdrojová a cílová IP adresa, porty a transportní protokol.

Fwd Avg Bytes/Bulk, *Fwd Avg Packets/Bulk*, *Fwd Avg Bulk Rate*, *Bwd Avg Bytes/Bulk*, *Bwd Avg Packets/Bulk*, *Bwd Avg Bulk Rate*. Tyto atributy byly z obou množin odstraněny. Kromě toho bylo zjištěno, že některé atributy mají nulové hodnoty ve většině řádků. Tyto byly také odstraněny.

V dalším kroku byl odstraněn chybný atribut *Unnamed: 0*. Dále nepotřebné atributy *Flow ID*, *Timestamp*, *Inbound*, *Source IP* a *Destination IP*.

Zbývající atributy a jejich význam byly analyzovány v následujícím kroku. Při rozhodování, zda atribut zachovat, hrálo roli, jak výrazně se jeho hodnoty lišily mezi DDoS a benigní komunikací. Trénovací množina byla proto rozdělena na DDoS a benigní toky, a pro každý atribut byly porovnány mediány hodnot v obou skupinách. Pokud se jejich mediány signifikantně lišily (konkrétní hodnoty závisely na aktuálně zkoumanén atributu), atribut byl ponechán; v opačném případě byl odstraněn.

3.3.4 Čištění a finální úpravy dat

Následně bylo nutné data vyčistit od chybějících a nesprávných hodnot. Nejprve byly zjištěny řádky se zápornými hodnotami, s hodnotou `null` nebo nekonečno. Vzhledem k jejich nízkému počtu byly tyto řádky odstraněny. V dalším kroku byly odstraněny všechny řádky, kde měl atribut *Protocol* hodnotu 0, jelikož se jednalo o chybná data.

Pro lepší přehlednost a efektivnější použití v modelech strojového učení byly atributy *Source Port* a *Destination Port* škálovány na tři binární atributy:

1. *common* (číslo portu 0–1023),
2. *registered* (číslo portu 1024–49151) a
3. *private* (číslo portu 49152–65535).

Podobná úprava se týkala i atributu *Label*, který původně rozlišoval různé typy DDoS útoků a benigní komunikaci. Všechny DDoS útoky byly sjednoceny pod binární atribut*DDoS*.

Po provedených úpravách zůstalo v datasetu 23 atributů. Pro lepší přehlednost a usnadnění práce byly jejich názvy převedeny na malá písmena a mezery nahrazeny podtržítky. Atributy jsou následující:

- `protocol`,
- `flow_duration`,
- `total_fwd_packets`,
- `total_length_of_fwd_packets`,
- `fwd_packet_length_mean`,
- `flow_bytes/s`,

- flow_packets/s,
- fwd_iat_total,
- fwd_iat_mean,
- fwd_packets/s,
- packet_length_mean,
- ack_flag_count,
- urg_flag_count,
- average_packet_size,
- avg_fwd_segment_size,
- init_win_bytes_forward,
- ddos,
- source_port_common,
- source_port_private,
- source_port_registered,
- destination_port_common,
- destination_port_private a
- destination_port_registered.

Posledním krokem v přípravě dat byla redukce trénovací a testovací množiny. Vzhledem k jejich původní velikost byly obě množiny zredukovány na 50 000 řádků.

4 Modely strojového učení

Po zpracování dat a přípravě trénovací a testovací množiny byly na data aplikovány různé metody strojového učení, přičemž pro každou z nich byl vytvořen odpovídající model. Tyto modely byly následně porovnány a pro výsledný nástroj byl vybrán ten nejpřesnější. Konkrétně byly použity rozhodovací stromy, support vector machines a neuronové sítě.

Testování proběhlo na náhodně vybrané podmnožině testovací sady o velikosti 20 % jejího původního rozsahu. Pro vyhodnocení modelů byly použity dvě metody vyhodnocení *classification report* a *confusion matrix* z knihovny scikit-learn [10], které poskytují detailní přehled o výkonu modelu. Zkoumány byly metriky accuracy, precision, recall a F1-score. Navíc classification report obsahoval support, který udává počet vzorků v benigní a DDoS třídě, macro average, což je průměr metrik pro jednotlivé třídy bez ohledu na jejich četnost, a weighted average, který představuje vážený průměr metrik zohledňující četnost jednotlivých tříd.

4.1 Rozhodovací stromy

Model rozhodovacího stromu byl vytvořen pomocí Python knihovny scikit-learn. Ta poskytuje širokou škálu nástrojů pro strojové učení. Z této knihovny byla použita třída `DecisionTreeClassifier` [11], která umožňuje snadné vytvoření rozhodovacího stromu na zadaných datech. Pro tento konkrétní model byla výška stromu omezena na sedm úrovní. Při testování byl zkoumán i jiný počet úrovní, od čtyř do deseti. Přestože stromy s těmito úrovněmi dosahovaly podobných výsledků, sedm úrovní se ukázalo jako nejpřesnější varianta.

Největší chybovost modelu se projevila u falešně pozitivních predikcí (false positives), kde model chybně klasifikoval 1,45 % testovacích vzorků. Naopak u falešně negativních výsledků (false negatives) byla chybovost pouze 0,04 %. Výsledky v classification report ukázaly, že model dosáhl 99 % přesnosti.[9] V dalších metrikách, jako jsou precision, recall a F1-score, model rovněž vykázal velmi

	precision	**recall**	**f1-score**	**support**
benigní	1,00	0,94	0,97	2 524
ddos	0,98	1,00	0,99	7 476
accuracy			0,99	10 000
macro avg	0,99	0,97	0,98	10 000
weighted avg	0,99	0,99	0,98	10 000

Tabulka 1: Classification Report rozhodovacího stromu.

[9] Třída `DecisionTreeClassifier` se ve výchozím nastavení může chovat nedeterministicky, což znamená, že každé její spuštění může vést k mírně odlišným výsledkům. V tomto případě model dosáhl v nejlepším případě přesnosti 99 %, v nejhorším 98 %.

dobré výsledky. Pro třídu *ddos* dosáhl model hodnoty 98 % pro precision, 100 % pro recall a 100 % pro F1-score. Podrobnější výsledky lze vidět v tabulce 1.

4.2 Support vector machines

Další metodou strojového učení, která byla zkoumána, je support vector machines (SVM). Byly vytvořeny dva modely SVM s různými typy jader.

První model byl vytvořen použitím třídy `LinearSVC` z knihovny scikit-learn, která využívá lineární jádro. Použití tohoto jádra je vhodné především pro lineárně separovatelná data. Druhý model využívá RBF jádro (Radial Basis Function), které je vhodné pro lineárně neseparovatelná data. Testování těchto dvou variant umožnilo vyhodnotit, zda jsou data lineárně separovatelná či nikoliv, a porovnat výkon SVM v obou případech.

Nejlepší přesnosti dosahoval model SVM s lineárním jádrem, který správně klasifikoval 87 % testovacích dat.

	precision	**recall**	**f1-score**	**support**
benigní	0,88	0,58	0,78	2 524
ddos	0,87	0,97	0,92	7 476
accuracy			0,87	10 000
macro avg	0,88	0,78	0,81	10 000
weighted avg	0,87	0,87	0,86	10 000

Tabulka 2: Classification Report SVM s lineárním jádrem.

Největší problém měl model při správné klasifikaci benigní komunikace, kdy jako falešně pozitivní označil 10,63 % všech testovacích dat. Tento problém je patrný i z metriky recall, která pro benigní třídu dosahuje pouze 58 %. Výsledky všech metrik jsou vidět v tabulce 2.

SVM model s RBF jádrem dosáhl pouze 76 % přesnosti.

	precision	**recall**	**f1-score**	**support**
benigní	0,71	0,09	0,17	2 524
ddos	0,76	0,99	0,86	7 476
accuracy			0,76	10 000
macro avg	0,74	0,54	0,51	10 000
weighted avg	0,75	0,76	0,69	10 000

Tabulka 3: Classification Report SVM s RBF jádrem.

Model měl také problémy se správnou klasifikací benigní komunikace, což potvrzuje velmi nízká hodnota recall pro benigní třídu, která činí pouze 9 %. Podrobnější výsledky jsou vidět v tabulce 3.

4.3 Neuronové sítě

Poslední zkoumanou metodou byly neuronové sítě. K tvorbě těchto sítí byla využita Python knihovna TensorFlow [12], která nabízí širokou škálu nástrojů pro vytváření neuronových sítí.

Bylo vytvořeno několik neuronových sítí s různými architekturami s cílem najít tu s nejlepšími výsledky. Každá síť obsahovala alespoň jednu skrytou vrstvu se 128 neurony. Následně se experimentovalo s dalšími skrytými vrstvami, dropout vrstvami a batch normalizací. Jako nejpřesnější síť se ukázala ta s nejjednodušší architekturou. Výsledný model má jednu skrytou vrstvu se 128 neurony a aktivační funkcí *ReLU*. Výstupní vrstva obsahuje jediný neuron a aktivační funkci *sigmoid* vhodnou pro binární klasifikaci. Pro prevenci přeučení byla do modelu přidána vrstva `Dropout`, která náhodně vynechává 50 % neuronů během trénování. Za stejným účelem byl implementován callback `EarlyStopping`, který monitoruje validační ztrátu a zastaví učení, pokud se ztráta nezlepšuje po dobu 5 epoch. Model byl následně optimalizován pomocí optimizéru `Adam` s učební rychlostí 0,001 a trénován s použitím binary crossentropy jako ztrátové funkce.

Po dokončení trénování byl model vyhodnocen na testovacích datech a pro změření přesnosti opět vygenerován classification report. Přestože tato neuronová síť dosahovala přesnoti 86 %, i ona měla problém se správnou klasifikací benigní komunikace. Výsledky všech metrik jsou vidět v tabulce 4.

	precision	**recall**	**f1-score**	**support**
benigní	0,99	0,46	0,63	2 524
ddos	0,85	1,00	0,92	7 476
accuracy			0,86	10 000
macro avg	0,92	0,73	0,77	10 000
weighted avg	0,88	0,86	0,84	10 000

Tabulka 4: Classification Report neuronové sítě.

V této kapitole byly představeny tři modely strojového učení a vyhodnoceny jejich výsledky na testovacích datech. Vzhledem k tomu, že neuronové sítě i SVM modely selhávaly ve správném rozpoznání benigní komunikace, byl jako jednoznačně nejúspěšnější model vybrán rozhodovací strom. Ten byl následně použit při implementaci nástroje.

5 Dokumentace vytvořeného nástroje

Tato kapitola podrobně popisuje dokumentaci vytvořeného nástroje *DDoShield*. Je důležité zdůraznit, že tento nástroj není plnohodnotnou aplikací pro detekci DDoS útoků, ale pouze prototypem. Jeho cílem bylo ověřit, zda model strojového učení, natrénovaný na konkrétním datasetu, obstojí i v reálném prostředí.

Následující kapitola je rozdělena do dvou částí. První podkapitola slouží jako programátorská dokumentace a popisuje technické řešení výsledného nástroje, včetně použitých technologií, knihoven a výzev při vývoji. Druhá podkapitola se zaměřuje na uživatelský pohled – zahrnuje instrukce k instalaci a popis uživatelského rozhraní.

5.1 Programátorská dokumentace

Hlavním cílem nástroje je umožnit nepřetržité monitorování síťové komunikace, automatickou detekci DDoS útoků v reálném čase a včasné varování uživatele před potenciální hrozbou. Jako cílová platforma byl zvolen desktop.

5.1.1 Architektura aplikace

Architekturu aplikace lze rozdělit na tři hlavní části.

1. *Frontend*, napsaný pomocí webových technologií, který zajišťuje uživatelské rozhraní a interakci s uživatelem.

2. *Backend*, napsaný v Pythonu, který zpracovává data, komunikuje s externími nástroji a pomocí rozhodovacího stromu detekuje podezřelou aktivitu.

3. *Externí analyzátor*, konkrétně CICFlowMeter-V4, který je využíván pro zpracování a detailní analýzu síťových toků.

Tato architektura jasně odděluje jednotlivé části a funkce aplikace, což přispívá k její přehlednosti, škálovatelnosti a snadnější údržbě. Díky použití technologií Electron.js pro frontend a Python pro backend je aplikace flexibilní a lze ji snadno rozšířit o další funkcionality. V případě potřeby lze téměř kteroukoliv z komponent vyměnit za jinou, aniž by bylo nutné provádět zásadní změny v celkové struktuře.

5.1.2 Použité technologie

Na následujících řádcích budou stručně popsány použité technologie.

5.1.2.1 Electron.js

Electron.js [13] je framework pro vývoj desktopových aplikací pro Windows, Linux a macOS s využitím webových technologií. Jeho hlavní výhodou je možnost

vytvářet aplikace podobně jako webové stránky – pomocí HTML, CSS a JavaScriptu, přičemž zároveň poskytuje přístup k nativním funkcím operačního systému. Electron je postaven na Chromiu a technologii Node.js. Díky jednoduché konfiguraci patří dnes mezi nejčastěji používané nástroje pro multiplatformní vývoj.

5.1.2.2 Python

Python [14] je interpretovaný a univerzálně použitelný programovací jazyk. Vzhledem ke své jednoduché syntaxi a rozsáhlému ekosystému knihoven nachází uplatnění v široké škále oblastí – od vývoje webových aplikací až po modelování neuronových sítí. Obzvlášť oblíbený je v datové analýze a strojovém učení.

5.1.2.3 CICFlowMeter-V4

CICFlowMeter-V4 [15] je open-source nástroj určený k analýze síťového provozu. Ze souborů ve formátu PCAP dokáže generovat obousměrné síťové toky a extrahovat z nich více než 80 síťových charakteristik. Hlavní nevýhodou nástroje je jeho chybějící podpora pro macOS, což omezuje jeho použití na operační systémy Windows a Linux.

5.1.3 Použité knihovny

Při vývoji byl kladen důraz na minimální použití externích knihoven. Frontend aplikace byl vyvinut bez jakýchkoliv externích řešení. V backendových skriptech bylo nutné integrovat čtyři knihovny, které slouží výhradně k zachytávání síťové komunikace a následné analýze dat.

5.1.3.1 Scapy

Scapy [16] je interaktivní knihovna napsaná v Pythonu, jejímž hlavním cílem je manipulace se síťovými pakety. Umožňuje zachytávání, generování a dekódování paketů v různých protokolech. Scapy je kompatibilní s operačními systémy Linux a macOS, a po instalaci externí knihovny Npcap [17] také s Windows.

Ve vytvořené aplikaci byla knihovna Scapy použita k zachytávání síťové komunikace a jejímu zápisu do PCAP souboru, který mohl být následně předán analyzátoru.

5.1.3.2 scikit-learn

Scikit-learn [10] je knihovna pro strojové učení v jazyce Python. Nabízí širokou škálu algoritmů pro klasifikaci, regresi a shlukování, stejně jako nástroje pro předzpracování dat, výběr vlastností a vyhodnocení modelů.

Ve vytvořené aplikaci byl pomocí scikit-learn natrénován model rozhodovacího stromu, který byl následně uložen a využit k analýze síťových dat. V bac-

kendových skriptech byla knihovna použita pro načtení tohoto modelu a jeho aplikaci na síťový provoz, čímž umožnila detekci DDoS útoků.

5.1.3.3 NumPy

NumPy [18] je knihovna pro matematické výpočty a práci s vícerozměrnými poli v Pythonu. Poskytuje rozsáhlou sadu funkcí pro matematické operace, statistické výpočty a efektivní manipulaci s daty.

V této aplikaci byla knihovna NumPy využita v backendu pro zpracování dat, zejména k detekci a opravě chybných hodnot.

5.1.3.4 Pandas

Pandas [19] je knihovna pro zpracování a analýzu dat v Pythonu. Poskytuje datové struktury DataFrame a Series, které umožňují efektivní práci s tabulkovými a časovými daty. Díky široké sadě funkcí umožňuje snadné filtrování, agregaci a transformaci dat.

V této aplikaci byla knihovna Pandas využita v backendu pro ukládání síťových dat. Byla využita pro načítání a úpravu datových souborů, zejména při zpracování výstupů z analyzátoru před jejich použitím v detekci útoků.

5.1.4 Struktura projektu

Projekt má vzhledem ke své povaze poměrně jednoduchou strukturu, jejíž základ byl vygenerován pomocí příkazu `npm init`. Do tohoto základu byly následně přidány všechny další soubory potřebné pro běh aplikace. Výsledná struktura projektu je následující.

```
ddos-detection-app/
├── assets/
├── CICFlowMeter-4.0/
├── node_modules/
├── python/
├── app.js
├── index.js
├── index.html
├── config.json
├── package.json
├── package-lock.json
├── style.css
└── tree_model.pkl
```

Uživatelské rozhraní aplikace implementují soubor `index.html` a soubor `style.css`. Soubor `index.html` obsahuje strukturu a obsah okna desktopové aplikace, tedy všechny prvky, se kterými uživatel interaguje. Ikony použité

v aplikaci jsou obsaženy ve složce `/assets`. Soubor `style.css` zajišťuje vizuální podobu aplikace. Interakce mezi uživatelem a těmito soubory je zajištěna skriptem `app.js`, který obsahuje JavaScriptový kód. Tento soubor obsahuje logiku pro obsluhu uživatelské akce a rovněž se stará o zobrazení dat přicházejících z backendu. Komunikace s backendovými skripty a dávkování dat je implementováno v souboru `index.js`.

Nejdůležitější částí aplikace jsou dva backendové Python skripty, uložené ve složce `python`. Soubor `packet_capture.py` má několik funkcí. Zachytává síťovou komunikaci, ukládá ji do PCAP souboru a kontroluje textový soubor s pozitivními DDoS predikcemi, přičemž všechna data o síťové komunikaci jsou průběžně zasílána na frontend.

Jakmile jsou data uložena do PCAP souboru, skript `process_packets.py` přebírá zodpovědnost za analýzu a zpracování těchto dat. Nejprve načte model rozhodovacího stromu ze souboru `tree_model.pkl`. Poté použije nástroj CICFlowMeter k analýze PCAP souboru a uložení síťových toků do CSV formátu. Tento CSV soubor je následně načten, zpracován do formátu, který je kompatibilní s rozhodovacím stromem, a na základě analýzy jsou provedeny predikce.

Výsledky analýzy jsou filtrované podle zdrojové IP adresy a pozitivní predikce jsou uloženy do textového souboru. Tento soubor je pojmenován podle aktuálního data ve formátu `DD-MM-YYYY_detected-ddos.txt`. Každý řádek souboru obsahuje tři údaje: zdrojovou IP adresu, zdrojový port a cílový port. Tento soubor slouží koncovým uživatelům pro zobrazení podezřelých IP adres, které mohou být indikací DDoS útoku.

Jedním z klíčových souborů je JSON soubor `config.json`, který uživateli umožňuje přizpůsobit určité části aplikace podle jeho potřeb. Tento soubor obsahuje tři parametry:

1. `batch_size`,
2. `max_threads`, a
3. `interface`.

Parametr `batch_size` určuje počet paketů v jednom PCAP souboru, který lze nastavit libovolně, přičemž hodnota může být různá. Vzhledem k tomu, že nástroj CICFlowMeter zpracovává oboustranné síťové toky, je doporučeno nastavit tento parametr na vyšší hodnotu. Pro detekci benigní komunikace stačí tento parametr nastavit na vyšší stovky, zatímco pro detekci DDoS útoků je nutné jeho hodnotu zvýšit na několik tisíc. Příliš nízká hodnota by mohla způsobit nesprávné fungování nástroje z důvodu nerozpoznání síťového toku. Parametr `max_threads` určuje maximální počet vláken, která budou zpracovávat síťovou komunikaci. Parametr `interface` definuje síťové rozhraní, na kterém bude komunikace odposlouchávána. Tento parametr se liší podle operačního systému a je třeba ho nakonfigurovat podle cílového zařízení. Výchozí hodnoty těchto parametrů jsou `1 000` pro `batch_size`, `20` pro `max_threads` a *Ethernet 2*[10]

[10]Ethernet 2 je síťové rozhraní v operačním systému Windows.

pro `interface`.

5.1.5 Výzvy při vývoji

Od počátku vývoje byl kladen důraz na to, aby nástroj byl plně kompatibilní s desktopovými zařízeními a fungoval na operačních systémech Linux, Windows a macOS. Cílem bylo minimalizovat počet externích závislostí a zároveň zajistit vysokou spolehlivost a uživatelskou přívětivost nástroje. Největší výzvou při vývoji se z těchto důvodů ukázal výběr nástroje pro analýzu síťového provozu.

Prvním vybraným nástrojem byl CICFlowMeter, protože byl použit při generování datasetu CIC-DDoS2019. Tento nástroj je však napsán v Javě a vyžaduje její instalaci pro správný běh. Kromě toho není kompatibilní s operačním systém macOS.

Poměrně dobrým řešením se jevila knihovna cicflowmeter [20] napsaná v Pythonu, která fungovala jako wrapper pro nástroj CICFlowMeter. Tato knihovna měla výhodu v tom, že byla kompatibilní s Pythonem a nevyžadovala nainstalovanou Javu. V době vývoje ale byla závislá na zastaralých verzích knihoven třetích stran. Jelikož knihovna nebyla aktivně udržována a závisela na nekompatibilních verzích, nemohla být v tomto nástroji použita.[11]

Další alternativou bylo použití Python knihovny NTLFlowLyzer [21] od autorů nástroje CICFlowMeter. Tento nástroj funguje podobně jako CICFlowMeter a dokáže z obousměrných síťových toků extrahovat více než 300 síťových charakteristik. Jeho instalace byla snadná, nicméně zpracování dat se, na rozdíl od CICFlowMeteru, lišilo. Rozhodovací strom dosahoval s takto zpracovanými daty velmi špatných výsledků a proto nemohl být ani tento analyzátor pro finální řešení použit.

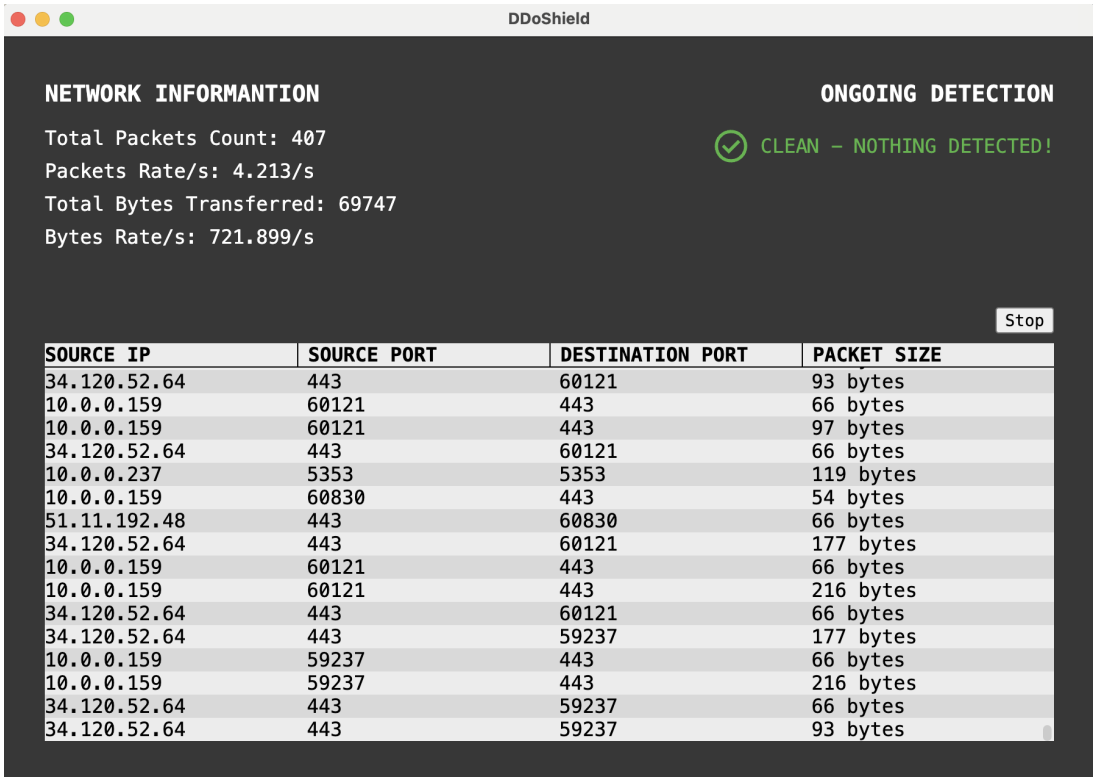
Po těchto průzkumech se nakonec vrátilo k původnímu řešení, a jako analyzátor byl zvolen nástroj CICFlowMeter. Toto řešení sice zajišťuje, že jsou data zpracována stejně jako v používaném datasetu, nicméně nepodporuje operační systém macOS a obsahuje chyby, které nejsou dobře zdokumentované. Sami autoři tohoto projektu v současné době při výskytu jakéhokoliv problému odkazují na nástroj NTLFlowLyzer. Jedná se tedy o kompromis, který, bohužel, nefunguje úplně spolehlivě.

5.2 Uživatelské rozhraní

Při vývoji byl kladen důraz na korektní detekce a správné zpracování dat. Z tohoto důvodu je uživatelské rozhraní aplikace velmi jednoduché a poskytuje uživateli pouze několik klíčových informací. Za účelem univerzálnějšího použití je rozhraní v angličtině.

Celá aplikace sestává pouze z jediné obrazovky, na které má uživatel k dispozici tabulku se síťovou komunikací v reálném čase. Ta obsahuje následující informace:

[11] V době psaní tohoto textu došlo k aktualizaci verzí externích knihoven.



Obrázek 2: Uživatelské rozhraní nástroje.

1. *Source IP*,
2. *Source Port*,
3. *Destination Port*, a
4. *Packet Size*.

Kromě toho jsou k dispozici další metriky, které jsou s každým příchozím paketem aktualizovány, a to:

1. *Total Packets Count*,
2. *Packet Rate/s*,
3. *Total Bytes Transferred*, a
4. *Bytes Rate/s*.

Dále je k dispozici tlačítko, které umožňuje uživateli pozastavit nebo obnovit zachytávání komunikace.

Aplikace uživateli poskytuje nejen základní informace o síťové komunikaci, ale také aktuální stav detekce DDoS útoků. Pokud v daný den nebyly žádné síťové

toky detekovány jako DDoS, aplikace zobrazí zprávu, která uživatele informuje o této skutečnosti. Tímto způsobem je uživatel ujištěn, že v síti neprobíhá žádný podezřelý provoz. Naopak, pokud dojde k detekci potenciálního DDoS útoku, aplikace okamžitě upozorní uživatele. Toto upozornění je vizuálně zvýrazněné, aby bylo pro uživatele dobře viditelné a aby si mohl včas všimnout možné hrozby. V případě jakékoliv chyby nebo problému v aplikaci se také zobrazuje upozornění, které je viditelné po dobu několika málo sekund. Podrobněji je uživatelské rozhraní vidět na obrázku 2.

6 Testování nástroje

Důležitou součástí této práce bylo testování vyvinutého nástroje na skutečném zařízení s co nejrealističtějším síťovým provozem. Cílem bylo simulovat běžnou benigní komunikaci uživatelů a současně generovat DDoS útoky pomocí specializovaných nástrojů. Testování mělo za úkol ověřit, jak dobře si model poradí s klasifikací provozu v reálných podmínkách a zda je schopen spolehlivě rozpoznat DDoS útoky od benigního provozu.

Za účelem testování vyvinutého nástroje byl zřízen virtuální server s operačním systémem Windows, na který byla aplikace nainstalována. Konkrétní IP adresy v lokálním testovacím prostředí jsou uvedeny v tabulce 5. Kromě těchto adres se v síťové komunikaci objevovala i další síťová aktivita, například systémová komunikace. Ta ale nebyla cíleně testována a proto byla při výsledném vyhodnocení přesnosti testování vynechána.

Zařízení	**IP adresa**
Server	158.194.92.94
Útočník	195.113.148.145
Uživatel 1	158.194.92.183
Uživatel 2	158.194.92.184
Uživatel 3	158.194.92.185

Tabulka 5: Lokální testovací prostředí.

6.1 Nástroje použité při testování

Simulace benigní komunikace nevyžadovala žádné speciální nástroje a postačovalo napodobit chování uživatelů. Provedení DDoS útoků již vyžadovalo použití specializovaných nástrojů. Pro realizaci základních DDoS útoků byly zvoleny veřejně dostupné nástroje Nmap [22] a hping3 [23]. Nmap byl využit pouze v počáteční fázi testování, ale kvůli svým omezeným možnostem byl následně nahrazen nástrojem hping3.

6.1.1 Nmap

Nmap [22] je veřejně dostupný a bezplatný nástroj určený ke skenování sítí. Slouží primárně k detekci otevřených portů, identifikaci běžících služeb a analýze bezpečnostních rizik. Kromě základních funkcí nabízí také pokročilé nástroje pro práci se sítí, mezi něž patří například Nping, který umožňuje generování a analýzu síťových paketů. Tento nástroj lze využít i k simulaci DDoS útoku, ačkoliv jsou jeho možnosti v tomto ohledu omezené.

6.1.2 hping3

Hping3 [23] je pokročilý síťový nástroj často používaný k testování bezpečnosti počítačových sítí. Umožňuje odesílání TCP, UDP a ICMP zpráv s detailním nastavením parametrů, jako je velikost zpráv, rychlost odesílání či specifikace hlaviček. Kromě toho disponuje parametrem `-flood`, který umožňuje extrémně rychlé odesílání zpráv v krátkých intervalech, čímž efektivně simuluje útoky. Díky těmto funkcím je hping3 účinným nástrojem pro simulaci DDoS útoků a testování odolnosti sítí vůči různým typům zatížení.

6.2 Benigní komunikace

Pro testování benigní komunikace byl na Windows server nainstalován balíček XAMPP, který obsahuje webový server Apache. V rámci tohoto balíčku byly na serveru umístěny výchozí webové stránky. K nim byla přidána jednoduchá webová aplikace napsaná v PHP. Běžná uživatelská aktivita byla simulována připojením k webovému serveru, prohlížením stránek a interakcí s webovou aplikací.

Testování probíhalo ve třech fázích. V první fázi se k webovému serveru připojil pouze jeden uživatel, ve druhé fázi dva a ve třetí fázi tři uživatelé. Jejich úkolem bylo prohlížet si webový obsah a interagovat s webovou aplikací. Pro analýzu byl kvůli nízké frekvenci příchozích paketů nastaven parametr `batch_size` na hodnotu 300. V každé fázi byly analyzovány dvě dávky. Přehled analyzovaných síťových toků a detekcí jsou k nahlédnutí v tabulce 6.

Počet uživatelů	Počet toků	Celkový počet detekcí
1	34	9
2	22	4
3	20	4

Tabulka 6: Výsledky testování benigní komunikace.

Ačkoli se na první pohled může zdát, že nástroj při detekci benigní komunikace nefungoval ideálně, po bližším prozkoumání detekovaných síťových toků se ukázalo, že šlo o nesimulovanou[12] systémovou komunikaci, specifickou pro každé zařízení. Pokud tuto komunikaci vyloučíme, lze test považovat za 100% úspěšný, protože žádný ze sledovaných síťových toků nebyl označen jako DDoS útok.

6.3 DDoS komunikace

Testování DDoS útoků probíhalo s využitím nástrojů Nmap a hping3, které v omezené míře umožňují simulovat tyto útoky i v lokálním testovacím prostředí. Útoky byly prováděny jak na veřejné, tak na privátní porty. Vzhledem

[12] Jedná se o komunikaci, která nebyla v průběhu testování cíleně generována.

k rozsáhlosti datasetu a technickým omezením nebylo možné simulovat všechny typy útoků, proto byly vybrány pouze ty, které byly relevantní pro testování.

Testování jednotlivých typů útoků probíhalo ve dvou fázích. Nejprve byl simulován útok s maximální intenzitou, při němž bylo v krátkém časovém intervalu odesláno velké množství paketů. Následně byl útok zopakován s nižší intenzitou, kdy mezi jednotlivými pakety byla nastavena prodleva, čímž byla simulována méně agresivní forma útoku.

Pro správnou detekci byl parametr `batch_size` na hodnotu 30 000. Každý útok byl prováděn po dobu několika sekund a při každé simulaci bylo zachyceno více než 60 000 paketů.

6.3.1 SYN Flood

Jako první byl nasimulován SYN Flood útok, jehož provedení bylo poměrně snadné. Nástroj hping3 nabízí širokou škálu konfigurovatelných parametrů, které umožňují odesílat segmenty s příznakem SYN z různých portů ve velmi krátkých časových intervalech.

6.3.1.1 Veřejný port

V první fázi byl otestován SYN Flood útok na veřejný port 80, na kterém běžela služba webového serveru Apache. Nejprve byl simulován intenzivní útok pomocí příkazu následujícího příkazu:

```
hping3 -S --flood -p 80 158.194.92.95
```

Detekce tohoto intenzivního SYN Flood útoku na veřejný port 80 byla neúspěšná. Ačkoli nástroj správně zachytil a analyzoval příchozí pakety, model nedokázal probíhající útok identifikovat.

V druhém kroce byl opět simulován SYN Flood útok na veřejný port, tentokrát však s nižší intenzitou. Tento útok byl spuštěn pomocí příkazu:

```
hping3 -S -p 80 -i u500 158.194.92.95
```

Ani tento méně intenzivní útok se však nástroji nepodařilo správně identifikovat.

6.3.1.2 Privátní port

Ve druhé fázi testování SYN Flood útoku byl proveden útok na privátní port. Na Windows serveru byl nalezen otevřený privátní port 49664, na který byly útoky směrovány. Stejně jako u veřejného portu byl nejprve testován velmi intenzivní útok pomocí nástroje hping3, později útok s menší intenzitou. Příkazy pro spuštění útoků se, až na číslo portu, shodovaly s příkazy výše.

Detekce SYN Flood útoku na privátní port byla v obou případech úspěšná. Při intenzivním útoku bylo po první analýze síťové komunikace nalezeno celkem 6 515 síťových toků, z nichž 6 508 bylo detekováno jako DDoS komunikace. Druhá analýza přinesla velmi podobné výsledky. U méně intenzivního útoku bylo po

první analýze rozpoznáno 7 440 síťových toků, z nichž 7 427 bylo identifikováno jako DDoS komunikace.

6.3.2 UDP Flood

Druhým testovaným DDoS útokem byl UDP Flood, který byl stejně jako předchozí útok realizován pomocí nástroje hping3. Tento nástroj umožňuje snadnou simulaci útoků i prostřednictvím protokolu UDP. Testování probíhalo podle výše popsaného scénáře.

6.3.2.1 Veřejný port

Testování v první fázi probíhalo obdobně jako u SYN Flood útoku. Nejprve byly simulovány útoky na veřejný port 80, přičemž v prvním kroku byl proveden velmi intenzivní útok pomocí příkazu:

```
hping3 --udp --flood -p 80 158.194.92.95
```

Ani po opakovaných analýzách však nástroj nedokázal probíhající DDoS útok rozpoznat. Stejně neúspěšná byla detekce i při simulaci méně intenzivního UDP Flood útoku spuštěného příkazem:

```
hping3 --udp -p 80 -i u500 158.194.92.95
```

Ten zůstal rovněž neodhalen.

6.3.2.2 Privátní port

Stejně jako u SYN Flood útoku byly i v tomto případě simulovány dva útoky na privátní port. Nejprve byl proveden velmi intenzivní útok, zaměřený na port 49670. Po první analýze bylo rozpoznáno 6 839 síťových toků, přičemž 5 831 z nich bylo detekováno jako DDoS útok. Při testování méně intenzivní varianty útoku nalezl nástroj po první asnalýze 7 635 síťových toků a z toho 7 630 označil jako DDoS. Detekce útoků na privátní porty tedy byla v obou případech úspěšná.

6.4 Závěry testování

Analýza benigní komunikace dopadla nad rámec očekávání, kde ani po opakovaných testech nebyla simulovaná komunikace mířící na port 80 označena jako DDoS útok. Jediné síťové toky, které byly v rámci benigního testování označeny jako DDoS, představovaly systémovou a jinou nesimulovanou komunikaci, mířící převážně na privátní porty. Při bližším prozkoumání datasetu bylo zjištěnu, že benigní komunikace obsahuje jenom velmi malé procento komunikace směřující na privátní porty.

Detekce DDoS útoků přinesla na první pohled rozporuplné výsledky. Zatímco útoky na privátní porty byly ve většině případů úspěšně identifikovány, útoky

směřované na veřejný port nástroj detekovat nedokázal. Po zpětné analýze datasetu se ukázalo, že v trénovacích datech bylo pouze velmi malé množství DDoS útoků cílících na veřejný port.

Bližší prozkoumání rozhodovacího stromu a pravidel na jeho jednotlivých úrovních odhalilo, že cílový port představuje primární a nejvýznamnější rozhodovací kritérium. Kvůli této nevyváženosti v trénovacích datech nebylo možné model správně natrénovat na všechny scénáře, což vedlo k neúspěšné detekci útoků na veřejný port a falešně pozitivní detekci komunikace mířící na privátní port.

Přesto analýza naznačuje, že model správně reaguje na vzory přítomné v trénovacích datech. V rámci dostupného datasetu dokázal spolehlivě rozlišit mezi benigní a DDoS komunikací. Lze tedy předpokládat, že rozšířením a vyvážením trénovací množiny by mohlo dojít k významnému zlepšení přesnosti detekce. Výsledky tak potvrzují funkčnost modelu v rámci podmínek, na které byl natrénován.

Závěr

Výsledkem této diplomové práce je analýza možností detekce DDoS útoků pomocí strojového učení a prototyp nástroje pro jejich automatickou detekci. Při vývoji prototypu byl kladen důraz na zpracování dat, vytvoření a porovnání různých modelů strojového učení a jejich implementaci do desktopové aplikace.

Testování nástroje probíhalo v několika fázích, během kterých byla testována jak benigní, tak i DDoS komunikace. Nástroj vykázal dobrou přesnost, jelikož úspěšně detekoval DDoS útoky zaměřené na privátní porty a nikdy neoznačil benigní komunikaci jako DDoS útok. Testování zároveň odhalilo některé nedostatky v trénovacích datech. Pro zajištění přesnější analýzy by bylo nutné rozšířit dataset o DDoS útoky cílící na veřejné porty a benigní komunikaci směřující na privátní porty. Rozšíření datové sady by modelu umožnilo natrénovat se na širší spektrum situací a tím zlepšit jeho detekční schopnosti.

Tato práce ukázala, že je možné vytvořit model pro automatickou detekci DDoS útoků a využít jej pro detekci v reálném čase. I přesto, že vytvořený prototyp neposkytuje 100% přesnost, poskytuje solidní základ pro další vývoj a zdokonalování nástrojů pro detekci DDoS útoků. Práce zároveň ukazuje klíčové oblasti, na které je potřeba se zaměřit, aby bylo dosaženo lepších výsledků v reálných podmínkách.

Conclusions

The result of this thesis is an analysis of the possibilities of detecting DDoS attacks using machine learning and a prototype of a tool for their automatic detection. During the development of the prototype, emphasis was placed on data processing, creation and comparison of various machine learning models and their implementation into a desktop application.

The tool was tested in several phases, during which both benign and DDoS communication were tested. The tool showed good accuracy, as it successfully detected DDoS attacks targeting private ports and never marked benign communication as a DDoS attack. Testing also revealed some shortcomings in the training data. To ensure a more accurate analysis, it would be necessary to expand the dataset to include DDoS attacks targeting public ports and benign communication directed to private ports. Expanding the dataset would allow the model to train on a wider range of situations and thus improve its detection capabilities.

This work showed that it is possible to create a model for automatic detection of DDoS attacks and use it for real-time detection. Although the prototype does not provide 100% accuracy, it provides a solid foundation for further development and improvement of DDoS attack detection tools. The work also shows key areas that need to be focused on in order to achieve better results in real-world conditions.

A Atributy datasetu CICDDoS2019

Tabulka 7: Seznam atributů

Atribut	**Popis**
Flow duration	Doba trvání toku v mikrosekundách.
Total Fwd Packet	Celkový počet paketů v dopředném směru.
Total Bwd Packets	Celkový počet paketů ve zpětném směru.
Total Length of Fwd Packet	Celková velikost paketů v dopředném směru.
Total Length of Bwd Packet	Celková velikost paketů ve zpětném směru.
Fwd Packet Length Min	Minimální velikost paketu v dopředném směru.
Fwd Packet Length Max	Maximální velikost paketu v dopředném směru.
Fwd Packet Length Mean	Průměrná velikost paketu v dopředném směru.
Fwd Packet Length Std	Směrodatná odchylka velikosti paketu v dopředném směru.
Bwd Packet Length Min	Minimální velikost paketu ve zpětném směru.
Bwd Packet Length Max	Maximální velikost paketu ve zpětném směru.
Bwd Packet Length Mean	Průměrná velikost paketu ve zpětném směru.
Bwd Packet Length Std	Směrodatná odchylka velikosti paketu ve zpětném směru.
Flow Bytes/s	Počet bajtů toku za sekundu.
Flow Packets/s	Počet paketů toku za sekundu.
Flow IAT Mean	Průměrný čas mezi dvěma pakety v toku.
Flow IAT Std	Směrodatná odchylka času mezi dvěma pakety v toku.
Flow IAT Max	Maximální čas mezi dvěma pakety v toku.
Flow IAT Min	Minimální čas mezi dvěma pakety v toku.
Fwd IAT Min	Minimální čas mezi dvěma pakety v dopředném směru.
Fwd IAT Max	Maximální čas mezi dvěma pakety v dopředném směru.
Fwd IAT Mean	Průměrný čas mezi dvěma pakety v dopředném směru.
Fwd IAT Std	Směrodatná odchylka času mezi dvěma pakety v dopředném směru.
Fwd IAT Total	Celkový čas mezi dvěma pakety v dopředném směru.

Atribut	Popis
Bwd IAT Min	Minimální čas mezi dvěma pakety ve zpětném směru.
Bwd IAT Max	Maximální čas mezi dvěma pakety ve zpětném směru.
Bwd IAT Mean	Průměrný čas mezi dvěma pakety ve zpětném směru.
Bwd IAT Std	Směrodatná odchylka času mezi dvěma pakety ve zpětném směru.
Bwd IAT Total	Celkový čas mezi dvěma pakety ve zpětném směru.
Fwd PSH Flags	Počet paketů s nastaveným PSH příznakem v dopředném směru (0 pro UDP).
Bwd PSH Flags	Počet paketů s nastaveným PSH příznakem ve zpětném směru (0 pro UDP).
Fwd URG Flags	Počet paketů s nastaveným URG příznakem v dopředném směru (0 pro UDP).
Bwd URG Flags	Počet paketů s nastaveným URG příznakem ve zpětném směru (0 pro UDP).
Fwd Header Length	Celkový počet bajtů použitých pro hlavičky v dopředném směru.
Bwd Header Length	Celkový počet bajtů použitých pro hlavičky ve zpětném směru.
FWD Packets/s	Počet dopředných paketů za sekundu.
Bwd Packets/s	Počet zpětných paketů za sekundu.
Packet Length Min	Minimální délka paketu.
Packet Length Max	Maximální délka paketu.
Packet Length Mean	Průměrná délka paketu.
Packet Length Std	Směrodatná odchylka délky paketu.
Packet Length Variance	Rozptyl délky paketu.
FIN Flag Count	Počet paketů s FIN příznakem.
SYN Flag Count	Počet paketů s SYN příznakem.
RST Flag Count	Počet paketů s RST příznakem.
PSH Flag Count	Počet paketů s PSH příznakem.
ACK Flag Count	Počet paketů s ACK příznakem.
URG Flag Count	Počet paketů s URG příznakem.
CWR Flag Count	Počet paketů s CWR příznakem.
ECE Flag Count	Počet paketů s ECE příznakem.
Down/Up Ratio	Poměr stahování a odesílání.

Atribut	Popis
Average Packet Size	Průměrná velikost paketu.
Fwd Segment Size Avg	Průměrná velikost segmentu v dopředném směru.
Bwd Segment Size Avg	Průměrná velikost segmentu ve zpětném směru.
Fwd Bytes/Bulk Avg	Průměrný počet bajtů ve velkém přenosu v dopředném směru.
Bwd Bytes/Bulk Avg	Průměrný počet bajtů ve velkém přenosu ve zpětném směru.
Fwd Init Win Bytes	Celkový počet bajtů odeslaných v počátečním okně v dopředném směru.
Bwd Init Win Bytes	Celkový počet bajtů odeslaných v počátečním okně ve zpětném směru.
Fwd Act Data Pkts	Počet paketů s alespoň 1 bajtem TCP datového obsahu v dopředném směru.
Active Min	Minimální doba, po kterou byl tok aktivní, než se stal nečinným.
Active Mean	Průměrná doba, po kterou byl tok aktivní, než se stal nečinným.
Active Max	Maximální doba, po kterou byl tok aktivní, než se stal nečinným.
Active Std	Směrodatná odchylka doby, po kterou byl tok aktivní, než se stal nečinným.
Idle Min	Minimální doba, po kterou byl tok nečinný, než se stal aktivním.
Idle Mean	Průměrná doba, po kterou byl tok nečinný, než se stal aktivním.
Idle Max	Maximální doba, po kterou byl tok nečinný, než se stal aktivním.
Idle Std	Směrodatná odchylka doby, po kterou byl tok nečinný, než se stal aktivním.

B Obsah elektronických dat

src/
: Tato složka obsahuje veškeré zdrojové kódy aplikace.

README.txt
: Textový soubor obsahuje informace k instalaci a spuštění aplikace, stejně jako návod na výměnu stávajícího modelu rozhodovacího stromu za nový, vytvořený pomocí Python skriptu ve složce `tree_model`.

text/
: Složka obsahuje text práce ve formátu PDF, včetně všech souborů potřebných pro vygenerování PDF dokumentu textu.

data/
: Tato složka obsahuje trénovací a testovací data, která sloužila k vytvoření a otestování rozhodovacího stromu.

tree_model/
: Složka obsahuje Python skript pro vytvoření modelu rozhodovacího stromu s využitím dat umístěných ve složce `data`. Skript lze znovu použít pro vytvoření modelu v případě, že aktuálně používaná verze knihovny scikit-learn zastará.

Literatura

[1] NÚKIB. *NÚKIB v roce 2024 zaznamenal více kybernetických incidentů než v předchozích letech* [online]. 2024 [cit. 2025-2-6]. Dostupný z: ⟨https://nukib.gov.cz/cs/infoservis/aktuality/2215-nukib-v-roce-2024-zaznamenal-vice-kybernetickych-incidentu-nez-v-predchozich-letech/⟩.

[2] BBC News. *Estonia hit by 'Russia cyber attack'* [online]. 2017 [cit. 2025-2-6]. Dostupný z: ⟨https://www.bbc.com/news/39655415⟩.

[3] ČT24. *Weby ministerstva vnitra a policie čelily kybernetickému útoku* [online]. 2023 [cit. 2025-2-6]. Dostupný z: ⟨https://ct24.ceskatelevize.cz/clanek/domaci/weby-ministerstva-vnitra-a-policie-celily-kybernetickemu-utoku-124⟩.

[4] University of New Brunswick. *CICDDoS2019 Dataset* [online]. 2019 [cit. 2025-2-6]. Dostupný z: ⟨https://www.unb.ca/cic/datasets/ddos-2019.html⟩.

[5] Mirkovic, Jelena. *DDoS Attack List*. n.d. Accessed: 2025-02-07. Dostupný z: ⟨https://www.isi.edu/people-mirkovic/ddos-benchmarks/ddos-attack-list/⟩.

[6] DARPA. *1999 DARPA Intrusion Detection Evaluation Dataset*. 1999. Accessed: 2025-02-07. Dostupný z: ⟨https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset⟩.

[7] UCI KDD Archive. *KDD Cup 1999 Data*. 1999. Accessed: 2025-02-14. Dostupný z: ⟨https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html⟩.

[8] Sharafaldin, Iman; Habibi Lashkari, Arash; Sahib, Iqball; Ghorbani, Ali. Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy. In. *Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy*. 2019, s. 1–8. Dostupný také z: ⟨http://dx.doi.org/10.1109/CCST.2019.8888419⟩.

[9] Canadian Institute for Cybersecurity. *CICFlowMeter - Applications*. 2025. Accessed: 2025-02-14. Dostupný z: ⟨https://www.unb.ca/cic/research/applications.html#CICFlowMeter⟩.

[10] developers, Scikit-learn. *Scikit-learn: Machine Learning in Python* [online]. n.d. [Cit. 2025-2-20]. Accessed: 2025-02-20. Dostupný z: ⟨https://scikit-learn.org/stable/index.html⟩.

[11] developers, Scikit-learn. *sklearn.tree.DecisionTreeClassifier* [online]. n.d. [Cit. 2025-2-20]. Accessed: 2025-02-20. Dostupný z: ⟨https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html⟩.

[12] TensorFlow. *TensorFlow: An Open Source Machine Learning Framework*. 2025. Accessed: 2025-02-20. Dostupný z: ⟨https://www.tensorflow.org/⟩.

[13] Electron Team. *Electron - Build cross-platform desktop apps with JavaScript, HTML, and CSS*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://www.electronjs.org/⟩.

[14] Python Software Foundation. *Python - The official home of the Python Programming Language*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://www.python.org/⟩.

[15] Ahlashkari, M.; contributors. *CICFlowMeter - An open-source tool for network traffic flow analysis*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://github.com/ahlashkari/CICFlowMeter⟩.

[16] Scapy Development Team. *Scapy - Powerful Python-based interactive packet manipulation program*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://scapy.net/⟩.

[17] Npcap Developers. *Npcap - High-performance packet capture library for Windows*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://npcap.com/⟩.

[18] NumPy Developers. *NumPy - The fundamental package for numerical computing with Python*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://numpy.org/⟩.

[19] pandas Development Team. *pandas - Python Data Analysis Library*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://pandas.pydata.org/⟩.

[20] Hieu, L. W.; contributors. *CICFlowMeter - GitHub repository*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://github.com/hieulw/cicflowmeter/tree/master⟩.

[21] Ahlashkari, M.; contributors. *NTLFlowLyzer - GitHub repository*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://github.com/ahlashkari/NTLFlowLyzer⟩.

[22] Lyon, Gordon. *Nmap: Network Mapper*. 2025. Accessed: 2025-03-20. Dostupný z: ⟨https://nmap.org/⟩.

[23] Linux, Kali. *hping3*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://www.kali.org/tools/hping3/⟩.

[24] The Chromium Project. *Chromium Projects - Open-source web browser and related projects*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://www.chromium.org/chromium-projects/⟩.

[25] OpenJS Foundation. *Node.js - JavaScript runtime built on Chrome's V8 engine*. 2025. Accessed: 2025-03-02. Dostupný z: ⟨https://nodejs.org/en⟩.

[26] WinPcap Team. *WinPcap - The packet capture and network analysis library for Windows*. 2025. Accessed: 2025-03-20. Dostupný z: ⟨https://www.winpcap.org/⟩.