VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE ANOMÁLIÍ NA ZÁKLADĚ SNMP KOMUNIKACE

ANOMALY DETECTION BASED ON SNMP COMMUNICATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **DANIEL ŠTĚPÁN**
AUTHOR

VEDOUCÍ PRÁCE **Mgr. Ing. PAVEL OČENÁŠEK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Štěpán Daniel**
Program: Informační technologie
Název: **Detekce anomálií na základě SNMP komunikace**
Anomaly Detection Based on SNMP Communication
Kategorie: Počítačové sítě

Zadání:

1. Seznamte se s principy analýzy anomálií v prostředí systémů počítačových sítí.
2. Analyzujte požadavky na systém umožňující analýzu aktivních dotazů (SNMP, NMAP, případně HIDS) a jejich průběhu v čase a na počítačích podobných či odlišných dle jejich účelu (PC, server, tiskárna), případně dle skupin uživatelů (účetní, administrátor, vývojář).
3. Navrhněte systém pro detekci bezpečnostních anomálií dle předchozího bodu.
4. Navržený systém implementujte dle instrukcí vedoucího práce.
5. Implementovaný systém ověřte na vhodně zvolených reálných datech.
6. Diskutujte získané výsledky, především vztahu anomálií k bezpečnostním událostem a možnosti dalšího rozšíření.

Literatura:

- Kurose, J. F. Computer networking: A top-down approach. Pearson, Essex, 2017, ISBN 978-1-292-15359-9.
- Stallings, W. Network security essentials: Applications and standards. Hoboken, 2016, ISBN 978-0-13-452733-8.
- Bishop, M. Computer security: Art & Science. Addison-Wesley, Boston, 2003, ISBN 0-201-44099-7.
- Ahmed, M., Mahmood Naser, A., Hu, J. A survey of network anomaly detection techniques. Journal of network and computer applications. Elsevier, 2016, 60(C), s. 19-31. ISSN 1084-8045.
- Buczak, A., Guven, E.. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications surveys and tutorials. IEEE, 2016, 18(2), s. 1153-1176.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání.

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

Vedoucí práce: **Očenášek Pavel, Mgr. Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 27. října 2020

Abstrakt

Cílem této práce bylo vytvořit prakticky použitelný soubor metod pro klasifikaci a detekci anomálií v prostředí počítačových sítí. Vytvořil jsem rozšíření pro monitorovací systém sítě ve formě dvou modulů otevřeného nástroje pro síťový monitoring, které jsou založeny na strojovém učení. Vytvořené moduly se dokáží na základě síťového provozu naučit charakteristiky běžného provozu. První modul, který je založen na algoritmu Random Forest Classifier, detekuje a je schopen rozpoznat několik známých útoků typu odepření služby. Druhý modul, založený na algoritmu Local Outlier Factor, detekuje anomální hladiny síťového provozu. Mezi útoky, které je první modul schopen detekovat patří záplavové útoky typu TCP SYN flood, UDP flood a ICMP flood. Dále pak aplikační útoky SSH Bruteforce a pomalý a fragmentovaný útok Slowloris. V průběhu práce jsem provedl testování zařízení výše zmíněnými metodami. Experimenty prokázaly, že první modul založený na klasifikaci je schopen detekovat známé útoky, až na útok Slowloris, jehož charakteristika se příliš neliší od normálního provozu. Druhý modul úspěšně detekuje zvýšenou hladinu provozu na síti, avšak neprovádí klasifikaci útoku.

Abstract

The aim of this thesis was to develop a practically applicable set of methods for classification and detection of anomalies in computer network environments. I have created extensions to the network monitoring system in the form of two modules for an open source network monitoring tool based on machine learning. The created modules can learn the characteristics of normal network traffic. The first module, based on the algorithm Random Forest Classifier, detects and is able to classify several known denial-of-service attacks. The second module, based on the algorithm Local Outlier Factor, detects anomalous levels of network traffic. Attacks that the first module is able to classify are the following: TCP SYN flood, UDP flood and ICMP flood. Moreover, it was trained to detect the SSH Bruteforce attacks and the slow and fragmented Slowloris attack. While working on this thesis, I tested the device using the methods mentioned above. The experiments showed that the classification-based module is able to detect known attacks, except for the Slowloris attack, whose characteristics are not very different from normal traffic. The second module sucessfully detects higher levels of network traffic, but does not perform the classification.

Klíčová slova

monitorování počítačové sítě, LibreNMS, detekce anomálií, SNMP, klasifikace DDoS

Keywords

network monitoring, LibreNMS, anomaly detection, SNMP, DDoS classification

Citace

ŠTĚPÁN, Daniel. *Detekce anomálií na základě SNMP komunikace*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Mgr. Ing. Pavel Očenášek, Ph.D.

Detekce anomálií na základě SNMP komunikace

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Mgr. Ing. Pavla Očenáška, Ph.D. Další informace mi poskytl pan Ing. Petr Chmelař. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Daniel Štěpán
12. května 2021

Poděkování

Chtěl bych poděkovat panu Ing. Petru Chmelařovi za odborné rady a cenné připomínky při psaní této bakalářské práce. Dále bych rád poděkoval mojí rodině za vytrvalou podporu.

Obsah

Kapitola 1

Úvod

Počet uživatelů vlastnících zařízení připojené do internetu a počet útoků na tato zařízení neustále roste. Motivem může být krádež dat, snaha poškodit uživatele, odepření přístupu k nějaké službě, ovládnutí zařízení a jeho využití pro další nekalé činnosti atd. Samozřejmě vzniká potřeba těmto útokům zamezit, a proto se kladou stále větší nároky na monitorování a ochranu všech zařízení před vnějšími i vnitřními vlivy. Zároveň platí, že útoky počítačových útočníků jsou stále sofistikovanější a náročnější na odhalení, z toho důvodu se dostávají do popředí techniky detekce anomálií, které mohou být založeny na statistice, strojovém učení, či korelacích. Tyto techniky nám umožní odhalit útočníka tam, kde ostatní nástroje i lidský faktor selhávají.

Cílem této práce bylo vytvořit takové prostředí, kde bude docházet k detekci anomálního síťového provozu na základě relevantních dat získaných prostřednictvím protokolu SNMP. Po dohodě s vedoucím práce je tato práce zaměřena nejen na detekci anomálního síťového provozu, ale také na klasifikaci nejznámějších útoků typů DoS. Ke sběru dat pomocí protokolu SNMP využiji dostupné open-source řešení, kterým je např. LibreNMS. Následně pro tento systém vytvořím dva moduly. První modul, který využívá algoritmus Random Forest Classifier, provádí klasifikaci případného útoku DoS. Modul bude trénován na datech, které sám vygeneruji simulováním různých útoků záplavového typu. Druhý modul provádí detekci anomálního provozu s využitím algoritmu Local Outlier Factor a bude trénován na datech sesbíraných pouze za normálního provozu. Oba využité algoritmy se nachází v open-source knihovně Scikit-learn. Následně nastavím systém LibreNMS tak, aby při detekci anomálního provozu vytvořil tzv. alert (upozornění), informoval uživatele a uchovával historii těchto alertů.

V první části této bakalářské práce, v kapitole 2, uvádím teoretické poznatky analýzy anomálií v prostředí počítačových sítí, kde popisuji typy anomálií a výstupy technik pro detekci anomálií. Dále jsem vysvětlil důležité vlastnosti algoritmů strojového učení a detekce anomálií, včetně způsobu jejich vyhodnocení. Následně jsem uvedl definici pojmů učení s učitelem a učení bez učitele, následováno výčtem metod používaných pro detekci anomálií se stručným popisem vybraných metod u každé kategorie.

Následující kapitolu 3 jsem věnoval popisu frameworku SNMP, který slouží ke správě a monitorování počítačových sítí a systémů, popsal jsem z jakých komponent se skládá, jejich popis a vysvětlení důležitých prvků. Také popisuji čím se liší jednotlivé verze protokolu SNMP, jaký princip komunikace a jaké metody protokol využívá. Vzhledem k tomu, že tato práce se zabývá daty získanými prostřednictvím tohoto protokolu, považuji za důležité představit pojmy a náležící terminologii tohoto systému.

V kapitole 4 popisuji nasazení vybraného nástroje LibreNMS. Na začátku kapitoly jsou uvedeny požadavky, které jsou kladeny na dohledový systém. Dále kapitola obsahuje nastavení procesu konfigurace klíčových prvků pro řešení této bakalářské práce, kterými jsou zejména přidání nového zařízení, nastavení pravidel, která vytvoří upozornění pokud model detekuje anomální provoz a aktivace mnou vytvořených modulů pro detekci anomálií.

V 5. kapitole je obsažen popis návrhu mnou vytvořených modulů pro detekci anomálií a klasifikaci útoků DoS. Kapitola obsahuje stručný popis obecného principu funkčnosti pluginů nástroje Nagios (protože právě s těmito pluginy nástroj LibreNMS pracuje) a dále obecný popis funkčnosti vlastních modulů. Významnou podkapitolou je část, která se zaobírá generováním dat, kde jsem simuloval různé síťové útoky, převážně útoky typu DoS, ale také útok SSH Bruteforce a Slowloris. Vygenerovaná data slouží k tréninku a otestování mého modelu pro klasifikaci DoS útoků. Dále kapitola popisuje proces získávání dat z databáze RRDtool, zmiňuje proces *Výběr rysů* (Feature selection) a dále uvádí můj model pro klasifikaci dat a druhý model pro detekci anomálního provozu. Je zde vysvětlen princip klasifikačního algoritmu *Náhodný les* (Random forest), následováno popisem skriptu pro vytvoření a trénink modelu. Poté je popsán pomocný skript, který načítá natrénovaný model, získává ze systému aktuální data a dává je na vstup vytvořenému modelu. Z modelu následně získá výsledek klasifikace, který předá skriptu napsanému v Bashi, jenž se načítá v systému LibreNMS jako modul systému Nagios. Na základě výsledku klasifikace vrátí systému LibreNMS jeden ze tří stavů (Ok, Warning, Critical). Podobně je popsán druhý model, který využívá algoritmus *Local Outlier Factor*, včetně skriptů pro vytvoření, trénink a použití modelu. Tento algoritmus porovnává lokální hustotu bodu s hodnotami jeho sousedů. Body s nižší hustotou (čísla nabývají záporných hodnot) jsou považovány za odlehlé hodnoty a tedy potencionálně anomální.

V kapitole 6, po dohodě s vedoucím práce, provádím analýzu dat nasbíraných pod vlivem jednotlivých útoků. Analýzu každého jednotlivého útoku jsem rozčlenil do tří částí. V první části, zvané *Grafická analýza*, jsem popsal vliv útoku na grafu, který je zobrazen v nástroji LibreNMS a který podává základní informace o stavu na síti s ohledem na konkrétní skupiny SNMP proměnných, konkrétně na skupiny IP, TCP, UDP a ICMP. Ve druhé části, zvané *Výběr rysů*, jsem vytvořil grafy na základě hodnot uložených v modelu *RandomForestClassifier* ve vlastnosti `feature_importances_`. Tyto hodnoty podávají informaci o tom, které proměnné mají nejvyšší váhu během procesu klasifikace. Ve třetí části *Numerická analýza* vytvářím tabulku 5 hodnot, které byly pro každý útok charakterizovány jako ty s nejvyšším vlivem při rozhodování zda se jedná o daný typ útoku. Pro tyto hodnoty počítám jejich průměr a směrodatnou odchylku. Na závěr kapitoly, v sekci *Testování*, provádím test obou modulů na datech v reálném čase, kdy nasimuluji jeden z natrénovaných typů útoků a čekám, zda systém LibreNMS tuto aktivitu vyhodnotí jako útok a vytvoří v systému upozornění.

Závěrečná kapitola 7 je věnována zhodnocení výsledků práce, především zde diskutuji dosažené výsledky, vztah anomálií k bezpečnostním událostem a zvažuji možnosti dalšího rozšíření.

Kapitola 2

Analýza anomálií v prostředí systémů počítačových sítí

Vzhledem k tomu, že téma detekce anomálií v počítačových sítí spadá do kategorie počítačové bezpečnosti, tak považuji za vhodné v první řadě nastínit aktuální bezpečností situaci internetu v České republice. Jak uvádí Zpráva o stavu kybernetické bezpečnosti České republiky za rok 2019 [19] vydaná Národním úřadem pro kybernetickou a informační bezpečnost (NÚKIB), opět došlo k meziročnímu nárůstu počtu kybernetických útoků vůči naší zemi. Nejčastějším vektorem útoku byly v roce 2019 podvodné e-maily a s ním spojený spam a phishing. Tyto typy se mohou jevit na první pohled jako neškodné, avšak pokud se útočníkům povede vylákat z oběti informace o platební kartě, důvěrné firemní informace apod., tak to může mít i fatální následky. Pokud jde o útoky typu ransomware, jenž zašifrují počítač a požadují po uživateli výkupné, tak tyto se meziročně stávají sofistikovanější a jsou více cílené, cílí hlavně na kritické infrastruktury, mezi jimiž jsou například nemocnice. Nejčastějšími aktéry, kteří ohrožovali Českou republiku v roce 2019 byli především kyberzločinci a kyberteroristé, jenž často pro své útoky používají nástroje vyvinuté jinými útočníky [19].

Očekává se, že tento trend růstu počtu kyberútoků bude pokračovat i v budoucnosti, z toho důvodu je potřeba klást stále větší důraz na zabezpečení infrastruktury a monitorování síťového provozu. Jedním ze způsobů, jak vyhodnotit bezpečnostní situaci je na síti je s využitím systémů pro detekci útoků IDS (intrusion detection systems). Tyto systémy využívají dva přístupy [22]:

- Detekce na základě signatur (Signature-based Detection).
- Detekce anomálií (Anomaly Detection).

Detekce na základě signatur

Funguje na analýze a detekci již známých útoků. V systému se udržuje databáze pravidel, které ovšem musí být pravidelně aktualizovány. Systém poté vyhledává známé vzory, může se jednat např. o sekvence bitů typických pro malware, známé hashe škodlivých souborů a další. Pokud je detekován takový software, proud dat, síťový provoz apod., jehož charakteristika odpovídá těmto pravidlům, tak systém na tuto aktivitu neprodleně upozorní.

Detekce anomálií

Úkolem je naučit systém to, jak vypadá počítačová síť za normálního stavu. Za této podmínky je systém následně schopen rozlišit chování, které se odchyluje od standardního či očekávaného stavu, a které se dá považovat za anomální. Také je schopen detekovat nové, neznámé útoky. Mezi používané metody tohoto přístupu patří statistická analýza časově označených dat a využití strojového učení (machine learning). Algoritmy využívající strojové učení rozdělujeme do čtyř kategorií [22], [3]:

- Algoritmy využívající učení s učitelem (Supervised Learning).
- Algoritmy využívající učení bez učitele (Unsupervised Learning).
- Algoritmy využívající kombinaci učení s učitelem a učení bez učitele (Semi-supervised Learning).
- Algoritmy využívající zpětnovazební učení (Reinforcement Learning).

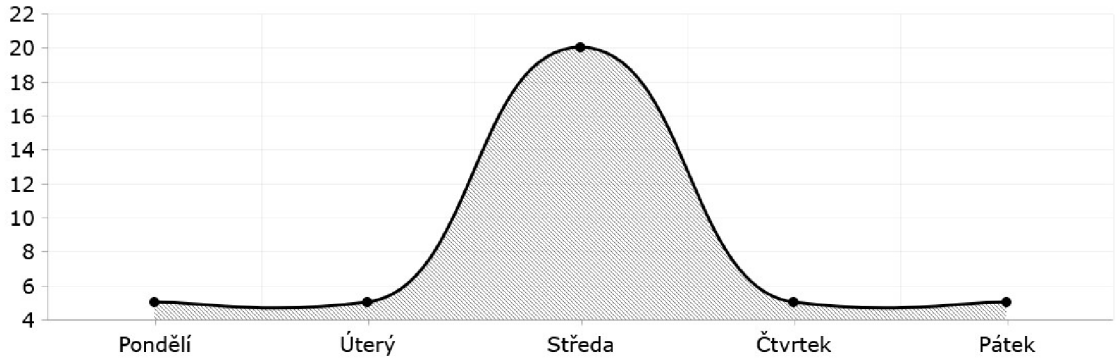
2.1 Typy anomálií

Anomálie lze dle jejich povahy zařadit do jedné ze tří kategorií [1]:

- Bodová anomálie (Point Anomaly)
- Kontextová anomálie (Contextual Anomaly)
- Kolektivní anomálie (Collective Anomaly)

Bodová anomálie

Bodová anomálie je označení pro takovou anomálii, kde jeden vzorek dat se odlišuje od ostatních vzorků. Je považována za nejjednodušší typ anomálie. Příkladem této anomálie může být auto, jenž každý den spotřebuje 5 litrů benzínu, ovšem v jeden konkrétní den tato spotřeba náhle vzroste na 20 litrů. Tato hodnota, vzhledem k tomu, že se výrazně liší od ostatních hodnot, je považována za anomální [1]. Příklad je uveden v grafické podobě v obr. 2.1.

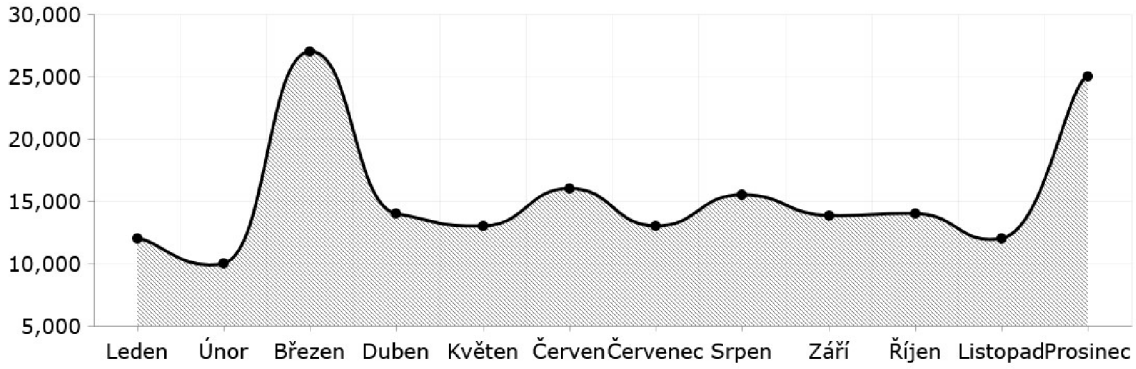


Obrázek 2.1: Příklad bodové anomálie. Ilustrace byla vytvořena na základního slovního popisu anomálie s využitím online nástroje pro tvorbu grafů AmCharts[2]

[2] https://live.amcharts.com/

Kontextová anomálie

Za kontextovou anomálii se označuje taková anomálie, kdy instance dat je anomální pouze v určitém kontextu (také se označuje jako podmíněná anomálie). Praktickým příkladem z reálného života je následující situace: Je monitorována útrata uživatele na kreditní kartě. Na základě jeho platební historie víme to, že uživatel pravidelně utrácí nejvíce peněz kolem poloviny prosince (přičemž částka může být mnohonásobně vyšší než průměrná týdenní útrata během celého kalendářního roku). Důvod zvýšené útraty v tomto případě tvoří vánoční nákupy. Tento předvánoční čas tvoří již dříve zmiňovaný kontext. Navýšení útraty v tuto dobu je předem očekávané, tudíž tato instance není označena jako anomální. Opačná situace by nastala v případě, že by k abnormálnímu navýšení útraty došlo v jiném kalendářním měsíci [1]. Příklad je vizualizován v obr. 2.2, kde útrata uživatele za měsíc prosinec nebude vyhodnocena jako anomálie, nicméně podobná útrata za měsíc březen již jako anomálie vyhodnocena bude.



Obrázek 2.2: Příklad kontextové anomálie. Ilustrace byla vytvořena na základního slovního popisu anomálie s využitím online nástroje pro tvorbu grafů AmCharts[4]

Kolektivní anomálie

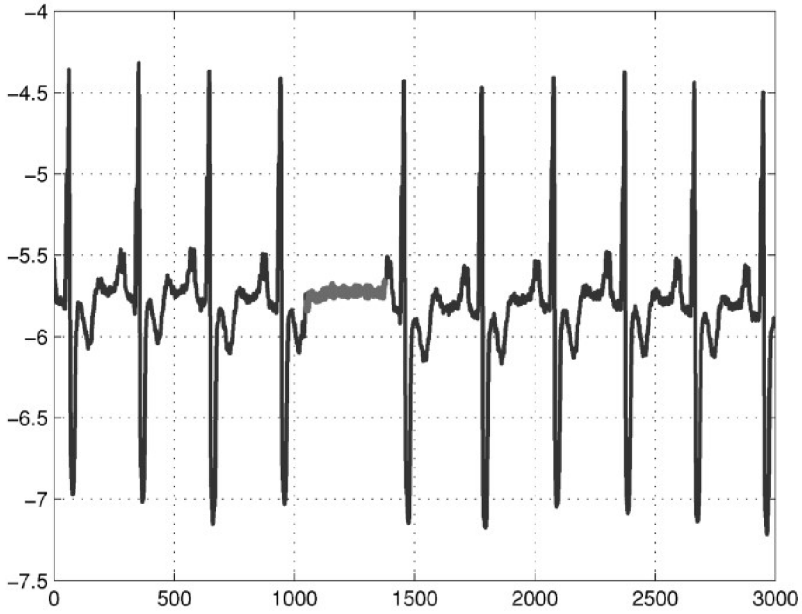
Kolektivní anomálií nazýváme kolekci souvisejících instancí dat, neboli takový úsek, jenž je vzhledem k celému souboru dat anomální. Přičemž jednotlivé instance dat v kolektivní anomálii samy o sobě nemusí být anomálií. Příklad této anomálie lze demonstrovat na výstupu lékařského systému EKG, kde série nízkých hodnot po určitý časový okamžik může indikovat arytmii srdeční aktivit, např. výskyt extrasystoly (předčasná kontrakce) [1]. Znázorněno na obr. 2.3.

2.2 Výstup technik pro detekci anomálií

Důležitou vlastností algoritmů používaných pro detekci anomálií je způsob, kterým se reprezentuje nalezená anomálie. Výstup je zpravidla reprezentován pomocí skóre či binárního značení:

- Skóre: Techniky založené na tomto principu přiřadí každé instanci dat určité skóre, které reprezentuje pravděpodobnost, s kterou se jedná o anomální výskyt. Je možné

[4] https://live.amcharts.com/



Obrázek 2.3: Příklad kolektivní anomálie [7]

stanovit hraniční hodnotu, a pokud dojde k přesažení této hranice, instance bude považována za anomálii. V tabulce 2.1 jsou instance dat reprezentovány jako A, B, C, D, E a každé je přiřazeno skóre v rozmezí 0 a 1. Instance D nabývá vyšší hodnoty než ostatní, v tomto případě je vyhodnocena jako anomální.

- Binární označení: Výstupem těchto technik je proměnná, která může nabývat dvou hodnot. Zpravidla se jedná o jednoduché označení typu 'je anomálie' či 'není anomálie'. Příklad je uveden rovněž v tabulce 2.1. Algoritmy tohoto typu jsou výpočetně jednodušší, poněvadž nemusí počítat konkrétní skóre.

Tabulka 2.1: Výstupy technik používaných pro detekci anomálií [1]

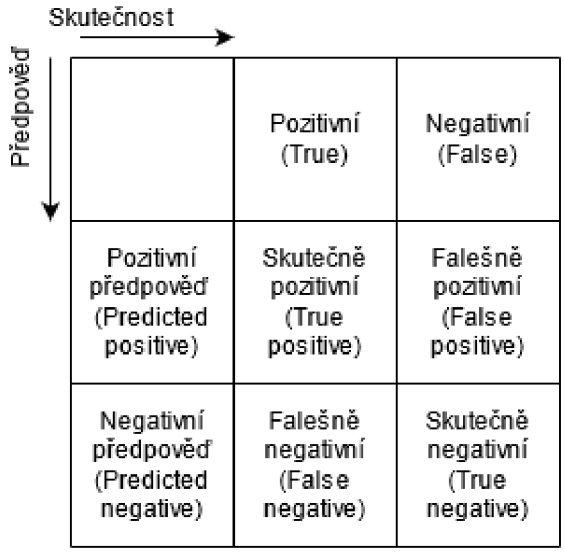
Instance dat	Skóre	Binární označení výskytu
A	0.3	Normální
B	0.4	Normální
C	0.2	Normální
D	0.8	Anomální
E	0.1	Normální

2.3 Hodnocení úspěšnosti klasifikace

Cílem detekce anomálií je rozhodnout, zda je daná hodnota anomální či ne. Výsledek rozhodovacího procesu následně spadá do jedné ze čtyř kategorií: skutečně pozitivní, falešně negativní, skutečně negativní, falešně pozitivní. Nejlépe si to lze představit na následujícím příkladu z oblasti medicíny: Mějme nějaký virus a testy, které detekují přítomnost tohoto viru v organismu.

- Skutečně pozitivní (True Positive – TP): Test byl vyhodnocen jako pozitivní a virus je opravdu přítomen v těle pacienta.
- Skutečně negativní (True Negative – TN): Test byl vyhodnocen jako negativní a virus v organismu opravdu přítomen není.
- Falešně negativní (False Negative – FN): Test byl vyhodnocen jako negativní, přestože virus v organismu přítomen je.
- Falešně pozitivní (False Positive – FP): Test byl vyhodnocen jako pozitivní, přestože virus v organismu přítomen není.

Tyto vztahy lze zobrazit graficky pomocí tzv. matice záměn (Confusion Matrix) [2], která je znázorněna v obr. 2.4. Z matice záměn lze odvodit následující hodnoty:



Obrázek 2.4: Matice záměn [2]

- Senzitivita (sensitivity, recall): Hodnota, která vyjadřuje, kolik ze všech opravdu pozitivních výsledků jsou označeny jako pozitivní.

$$sensitivity = \frac{TP}{TP + FN} \tag{2.1}$$

- Preciznost (precision): Hodnota, která vyjadřuje, kolik ze všech pozitivně označených výsledků jsou skutečně pozitivní.

$$precision = \frac{TP}{TP + FP} \tag{2.2}$$

- Specificita (specificity): Hodnota, která vyjadřuje, kolik negativních výsledků bylo označeno jako negativní.

$$specificity = \frac{TN}{TN + FP} \tag{2.3}$$

2.4 Učení s učitelem a učení bez učitele

Obecně, metody strojového učení lze rozdělit na metody učení s učitelem, metody učení bez učitele, existuje i jejich kombinace [2]. Poslední kategorií je tzv. zpětnovazební učení.

- Učení s učitelem (Supervised Learning): Tyto metody potřebují ke své funkci tzv. trénovací soubor dat (dataset), jenž obsahuje označená normální data a označená anomální data. Na těchto datech se systém naučí správně rozpoznat a zařadit nová a neznámá data.
- Učení bez učitele (Unsupervised Learning): Tyto metody nepotřebují označená trénovací data. Obecně fungují tak, že předpokládají, že výskyty správných hodnot jsou mnohem častější než výskyty anomálií, respektive vybírají ta data, která se určitým způsobem odlišují od většiny. Algoritmus se učí rozpoznat vazby mezi daty a sdružuje ta data, jenž mezi sebou sdílejí příbuzné charakteristiky. Na konci procesu trénování je model schopen rozpoznat, která data jsou v rámci souboru dat normální a která jsou anomální.
- Kombinace učení s učitelem a učení bez učitele (Semi-supervised Learning): Tento způsob kombinuje oba předchozí přístupy a spočívá v tom, že pouze část dat je označená. Například mohou být označena pouze normální data. Systém se tak naučí rozpoznávat nová anomální data, která se liší od normálních.
- Zpětnovazební učení (Reinforcement Learning): Algoritmy založené na zpětnovazebním učení fungují na tom principu, že průběžně adaptují a upravují svůj výsledek procesu učení na základě podané zpětné vazby. Zpětná vazba může být pozitivní (odměna) nebo negativní. Algoritmus se snaží maximalizovat odměnu, a tím hledá nejlepší cestu, která vede k požadovanému výsledku [22].

2.5 Metody využívané pro detekci anomálií

Pro účely detekce anomálií v prostředí počítačových sítí se využívá řada metod, které budou probrány v této podkapitole. Jedná se o klasifikační metody, statistické metody a metody využívající shlukové analýzy.

2.5.1 Klasifikační metody (Classification-based Techniques)

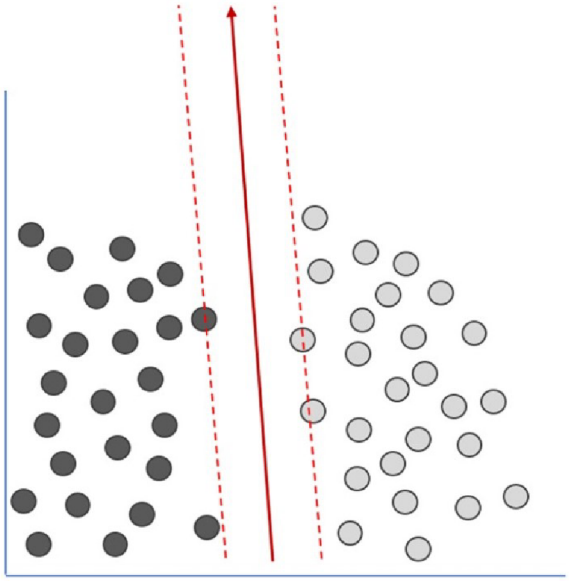
Tyto metody jsou založeny na principu učení s učitelem. Fungují na tom principu, že natrénují model (klasifikátor) z označených trénovacích dat, který poté klasifikuje nová data a rozděluje je do požadovaných tříd.

Tyto techniky lze rozdělit dále do dvou kategorií dle počtu tříd, do které řadí nový vzorek, a to na tzv. *one-class*, kde je pouze jedna třída a vše ostatní, co do ní nepatří je považováno za anomálii. Dále existují tzv. *multi-class* techniky, kde se klasifikátor musí prvně naučit rozlišit mezi těmito vícero třídami. Pokud nelze hodnotu zařadit s vysokou pravděpodobností do žádné třídy, lze ji prohlásit za anomální [7]. Tento způsob je vhodný pro detekci známých typů útoků.

Metoda podpůrných vektorů (Support Vector Machine)

Základní podstatou algoritmu SVM je to, že se snaží nalézt takovou nadrovinu, která prostor rozdělí tak, aby různé třídy ležely v opačných poloprostorech. Využívá pomocné vektory,

jak je možné vidět na obr. 2.5. Při určování polohy a směru vektorů se algoritmus snaží maximalizovat velikost mezi vektory a nadrovinou [1].



Obrázek 2.5: Metoda Support Vector Machine [2]

Bayesovské metody (Bayesian Methods)

Bayesovské metody se používají v systémech pro klasifikaci dat typu multi-class. Fungují na tom principu, že se snaží určit tzv. posteriorní pravděpodobnost, že prvek spadá do konkrétní třídy. Vždy je vybrána třída s největší pravděpodobností. Základní technika Bayesovských metod předpokládá nezávislost mezi různými atributy. Avšak existují určité varianty této techniky využívající komplexní Bayesovské sítě, které mohou zachytit podmíněné závislosti mezi atributy [7].

Neuronové sítě (Neural Networks)

Jedná se o techniku strojového učení, která simuluje fungování lidského nervového systému. Neuronovou síť lze rozdělit na tři části – vstupní vrstva, skrytá vrstva (skrytých vrstev může být i více) a výstupní vrstva. Na začátku procesu učení jsou každé vstupní hodnotě náhodně přiděleny váhy. Výstup sítě je porovnán s očekávaným výsledkem a na základě velikosti chyby dojde k úpravě velikosti vah v další iteraci. Proces se opakuje dokud není dosaženo minimální chyby. Existuje více druhů neuronových sítí. Nejjednodušší neuronová síť, obsahující pouze jeden neuron a která provádí binární klasifikaci, se nazývá *perceptron*. Dále existují tzv. *konvoluční neuronové sítě*, které se používají často v oblastech zpracování obrazu. dalším typem jsou *rekurentní neuronové sítě*, které mají speciální architekturu pro zpracování sekvenčních dat. Neuronové sítě s větším množstvím skrytých vrstev spadají do kategorie *hlubokého učení* (deep learning), které je podmnožinou strojového učení [3].

Metody založené na detekčních pravidlech (Rule-based Methods)

Tyto techniky se učí pravidla, která zachycují běžné chování systému. Instance dat, na kterou se nevztahuje žádné takové pravidlo je považováno za anomálii. Používají se jak v *one-class* klasifikaci, tak v *multi-class* klasifikaci. Použití těchto technik sestává se dvou kroků. Prvním krokem je naučit se pravidla z trénovací sady sad pomocí vhodného algoritmu, kterým může být např. Ripper nebo algoritmus *Rozhodovací stromy* (Decision Trees), který tvoří základ algoritmu *Random Forest*, jenž je popsán v podkapitole 5.3. Každému pravidlu se přiřadí určitá hodnota, tzv. *confidence value*, která je úměrná poměru mezi počtem případů správně klasifikovaných pravidlem a celkovým počtem případů pokrytých pravidlem. Druhým krokem je najít pro každou testovací instanci nejvhodnější pravidlo, které ji zachycuje [7].

2.5.2 Statistické metody (Statistical Methods)

Statistické metody předpokládají, že výskyt normálních instancí je ve stochastickém modelu v oblasti s vysokou pravděpodobností, zatímco anomální instance jsou v oblasti s nízkou pravděpodobností.

V prostředí počítačových sítí jsou tyto metody obvykle založeny na předpokladu, že síťový provoz odpovídá určitému rozložení. Nejjednodušší způsob, jak vytvořit statistický model je spočítat parametry funkce hustoty pravděpodobnosti pro každou známou třídu síťového provozu a poté testovat neznámý vzorek, abychom určili, do jaké třídy s nejvyšší pravděpodobností patří [4].

Analýza hlavních komponent (Principal Component Analysis)

Jde o techniku, která má své využití zejména pro hledání vzorů (patterns) v datech o vysoké dimenzi. Redukuje počet dimenzí bez přílišné ztráty informace [26]. Na vstupu má množinu dat (dataset) a vrátí nový, zrekonstruovaný dataset stejného tvaru. V tomto transformovaném prostoru, všechny tzv. *features*, nazývané komponenty, jsou na sobě vzájemně nezávislé a komponenty s největší informační hodnotou jsou na začátku datasetu [17].

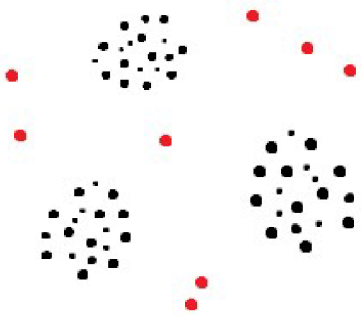
2.5.3 Shluková analýza (Clustering)

Shluková analýza je technika, využívající učení bez učitele, při které se podobné instance dat sdružují do tzv. shluků. Při využití techniky shlukování pro detekci anomálií vždy platí následující tři předpoklady [1]:

- Předpoklad 1: Můžeme vytvářet pouze shluky normálních (validních dat). Proto jakékoliv nové instance dat, které se nehodí do žádného shluku, jsou považovány za anomálie.
- Předpoklad 2: Pokud shluk obsahuje normální i anomální data, tak anomální data, na rozdíl od normálních dat, leží daleko od geometrického středu[5] shluku. Na základě tohoto předpokladu a znalosti vzdálenosti instance dat od středu lze detekovat anomálii.
- Předpoklad 3: Pokud máme více shluků různých velikostí, tak data nacházející se v menších a řidších shlucích jsou považovány za anomální. Naopak za anomální nejsou

[5] https://cs.wikipedia.org/wiki/Geometrick%C3%BD_st%C5%99ed

považovány instance dat ve větších a hustějších shlucích. Ilustrováno na příkladu v obr. 2.6, kde anomální hodnoty jsou jsou označené červenou barvou. Obvykle je určena nějaká prahová hodnota (threshold), oproti které se vlastnosti shluku porovnávají.



Obrázek 2.6: Shluky a anomální instance dat. Validní data ve shlucích jsou vyznačena černou barvou. Anomální data jsou vyznačena červenou barvou. Vytvořeno v programu Malování.

2.6 Shrnutí

V této kapitole jsou popsány základní principy analýzy anomálií. Na začátku kapitoly byla zmíněna bezpečnostní situace v České republice jako motivace pro to, proč je důležité zaobírat se tímto tématem a mít dobře nastavené monitorování sítě.

Popsal jsem dva základní přístupy, které se využívají u systémů pro detekci útoků IDS, a to detekce na základě signatur a detekce anomálií. Dále byly rozebrány typy anomálií a u každého typu byl uveden příklad této anomálie.

Následně došlo na popis výstupu technik detekce anomálií. Výstupem může mít buď skóre nebo binární stav, který pouze říká, zda došlo či nedošlo k výskytu anomálie, případně pravděpodobnost. V případě skóre lze určit hranici, od které se daná hodnota bude považovat za anomální. Dále jsem popsal způsoby hodnocení úspěšnosti klasifikace, během které mohou nastat 4 případy. Výsledek může být buď skutečně pozitivní, skutečně negativní, falešně pozitivní či falešně negativní. Poté jsem vysvětlil související pojmy jako senzitivita, preciznost a specificita.

V další podkapitole byly vysvětleny pojmy učení s učitelem a bez učitele. Součástí poslední podkapitoly je rozdělení metod pro strojové učení a detekci anomálií a stručný popis často používaných technik a algoritmů.

V následující kapitole, která se věnuje SNMP, je popsána architektura a související pojmy.

Kapitola 3

SNMP

Simple Network Management Protocol (SNMP) je protokol, jenž se používá ke správě síťových prvků v počítačových sítích. Tento protokol je podporován velkým množstvím zařízeních, včetně směrovačů, přepínačů, pracovních stanic, tiskáren, modemů a dalších. Nicméně, protokol SNMP tvoří pouze část většího celku, který se nazývá Internet Standard Management Framework (SNMP Framework) [6]. Právě SNMP Framework bude popsán v této kapitole, jeho architektura a popis komponent ze kterých se skládá. I přesto, že popularita protokolu SNMP je na ústupu, stále je široce využívaný zejména pro svoji dosavadní rozšířenost a absenci jiného protokolu, který by jeho funkci mohl plně zastoupit [29].

3.1 Architektura SNMP Frameworku

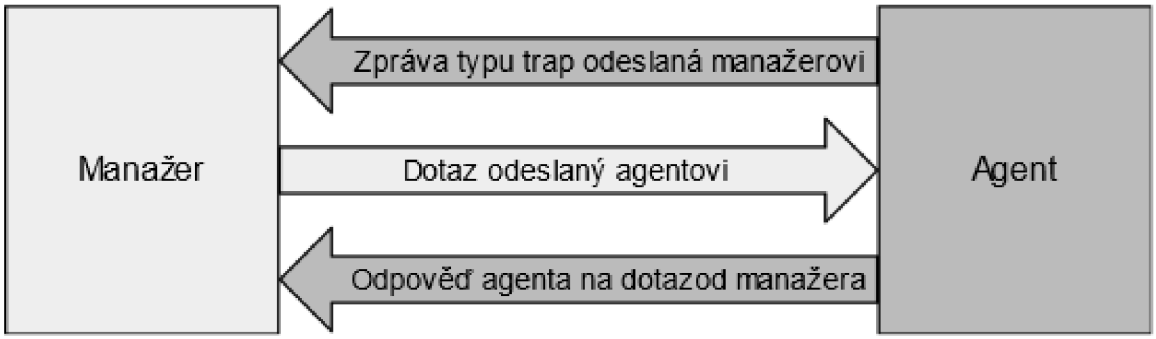
Tento internetový model pro správu sítě tvoří komponenty, které definují nejen to, jak jsou data vyměňována a posílána protokolem SNMP, ale také to, jak jsou spravovaná data strukturována a ukládána. Tyto prvky jsou následující [11]:

- SMI (Structure of Management Information) – Konzistentní způsob popisu charakteristik SNMP spravovaných zařízení.
- MIBs (Management Information Bases) – Databáze monitorovaných objektů.
- SNMP (Simple Network Management Protocol) – Komunikační protokol SNMP definuje to, jak dochází k výměně informací mezi SNMP agenty a řídící stanicí.
- Zabezpečení a administrace – K třem hlavním komponentám přidává vylepšení z hlediska zabezpečení a řeší problémy související s implementací SNMP.

V SNMP systému jsou dvě důležité entity, kterými jsou řídící stanice (manažer) a agent. Řídící stanice je obvykle server, na kterém běží systém pro správu sítě. Na řídící stanice se často také odkazuje jako na NMSs (Network Management Stations). Jsou zodpovědné za stahování informací a přijímání zpráv typu trap od agentů v síti. Získané informace se následně využívají k diagnóze zařízení, monitorování výkonu, či mohou být využity k detekci potencionálního útoku na zařízení.

Druhou entitou je agent, jedná se zpravidla o software běžící na monitorovaném zařízení. Může jít buď o samostatný program (např. daemon v systémech typu Unix), nebo může být součástí operačního systému (např. router poháněný systémem Cisco IOS). Agentem sledované veličiny mohou být například stavy síťových rozhraní a síťový provoz, který

přes ně prochází. Dále je možné sledovat vytížení interních pamětí, zaplnění disků a další veličiny [15]. Vztah mezi manažerem a agentem je ilustrován na obr. 3.1.



Obrázek 3.1: Vztah mezi manažerem a agentem [15]

3.2 SMI (Structure of Management Information)

Jedná se o standard RFC 1155 [24], který definuje strukturu, syntax a charakteristiky monitorovaných dat. Jazyk SMI je založen na notaci ASN.1 (Abstract Syntax Notation One). Využití tohoto standardu zajišťuje, že přijatá data pocházející od různých výrobců, s různou architekturou a s různými operačními systémy, jsou stejně interpretována řídicí stanicí.

Každý objekt popisovaný pomocí ASN.1 má své *jméno*, které je dáno jednoznačným identifikátorem OID (Object Identifier), *syntax*, jenž definuje abstraktní datový typ spojený s objektem a v neposlední řadě *kódování*, které určuje, jak jsou instance objektu reprezentovány při přenosu po síti [14]. Základní datové typy ASN.1 použité k tvorbě informací o monitorovaných objektech jsou popsány v tabulce 3.1.

Tabulka 3.1: Vybrané základní datové typy ASN.1 [14]

Datový typ	Popis
INTEGER	32bitové číslo definované v ASN.1
OCTET STRING	binární či textový řetězec (až 64kB definovaný ASN.1)
OBJECT IDENTIFIER	přiřazeno ASN.1
Integer32	32bitové celé číslo
Unsigned32	kladné 32bitové číslo
IPaddress	32bitová IP adresa, síťový formát
NetworkAddress	pro reprezentaci jiných typů adres
Counter32	32bitový čítač, po přetečení se nastaví na 0
Counter64	64bitový čítač
Gauge32	32bitový čítač, po přetečení uchová max. hodnotu až do resetu
TimeTicks	čas měřený v setinách sekundy od dané události
Opaque	neinterpretovaný řetězec podle ASN.1

3.3 MIBs (Management Information Bases)

Každé spravované zařízení obsahuje řadu proměnných, jejichž hodnota může být čtena či nastavena. Tyto proměnné podávají informace o stavu zařízení, které se zasílají do tzv. řídící stanice, nebo lze s jejich pomocí zařízení ovládat. Databáze monitorovaných objektů MIB, definována v RFC 1213 [16], je úplná sada těchto proměnných, které popisují charakteristiky konkrétního typu zařízení [11]. Standardní databáze MIB-II obsahuje 10 základních skupin objektů, jak je uvedeno v tabulce 3.2. Přičemž každá skupina obsahuje množinu objektů, kde každá množina obsahuje další proměnné.

Tabulka 3.2: 10 základních skupin objektů definovaných v MIB-II [14]

Jméno podstromu	OID	Popis
system	1.3.6.1.2.1.1	operační systém: jméno OS, systémový čas, kontakt
interface	1.3.6.1.2.1.2	stav síťového rozhraní
at	1.3.6.1.2.1.3	překlad adres (address translation) – nepoužívá se
ip	1.3.6.1.2.1.4	IP adresa, směrovací informace
icmp	1.3.6.1.2.1.5	sleduje chyby ICMP
tcp	1.3.6.1.2.1.6	stav spojení TCP (closed, listen, synSent)
udp	1.3.6.1.2.1.7	statistiky UDP - přišlé/odešlé pakety
egp	1.3.6.1.2.1.8	statistiky EGP
transmission	1.3.6.1.2.1.10	objekty závisející na přenosovém médiu
snmp	1.3.6.1.2.1.11	počet vyslaných/přijatých SNMP paketů

Existuje více databází MIB. Kromě databází standardizovaných IETF RFC (MIB-I, MIB-II), existují i databáze privátní, kde výrobci síťového HW i SW popisují objekty a atributy potřebné pro správu svých vlastních zařízeních. [14] Jedná se o rozšíření těchto standardizovaných MIB databází. Mnohé nástroje pro monitorování síťové infrastruktury pomocí SNMP technologie již obsahují databáze MIB od řady výrobců, příkladem může být systém LibreNMS[1].

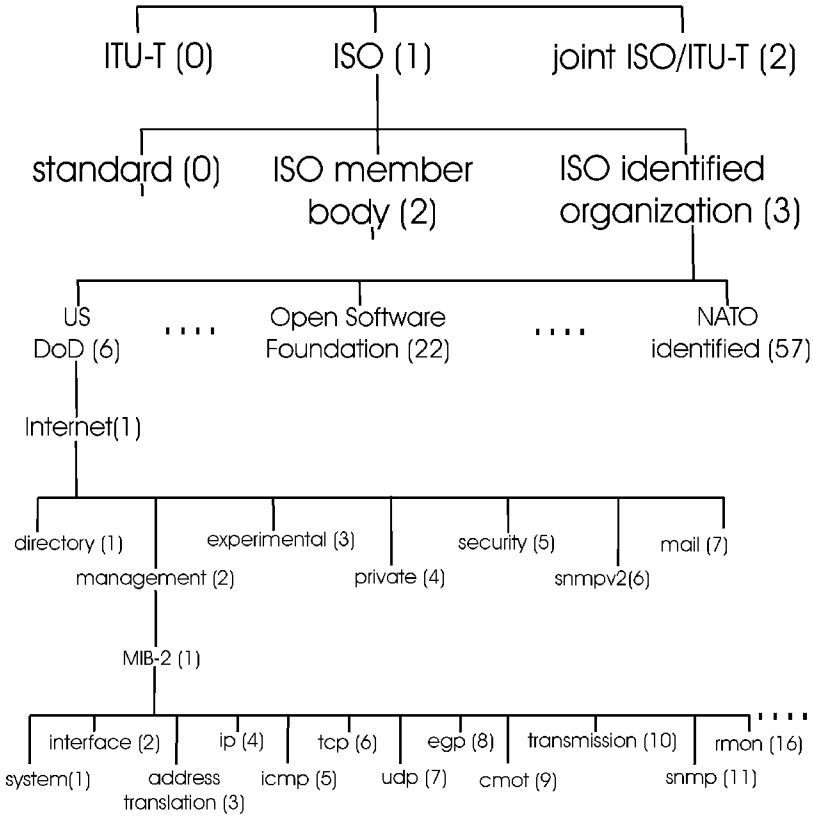
Jak je patrné z obrázku 3.2, objekty v MIB jsou strukturovány hierarchicky. Každý objekt je identifikován jak jménem, tak číslem (v závorce). Libovolný bod ve stromu je tudíž identifikovatelný posloupností jmen nebo čísel, které určují cestu od kořene k tomuto bodu ve stromu identifikátorů OID [28].

3.4 SNMP (Simple Network Management Protocol)

Protokol SNMP, popsaný ve standardu RFC 1157 [5] je aplikační protokol TCP/IP přenášený transportním protokolem UDP na portu 161. V případě asynchronní zprávy typu *trap* je použit port 162.

SNMP funguje na principu dotaz-odpověď (request-response). Řídící jednotka odesílá SNMP dotazy agentovi SNMP, který přijme dotaz, načte ho, vykoná požadovanou operaci a odpověď zašle zpět řídící jednotce. Požadavkem může být například výpis aktuální hodnoty objektu či změna jeho hodnoty. Tato činnost se opakuje v pravidelných časových intervalech, kdy agent pasivně čeká na dotaz od řídící jednotky (tento typ komunikace se nazývá vyzývání neboli polling) [14].

[1] https://github.com/librenms/librenms/tree/master/mibs



Obrázek 3.2: Strom identifikátorů objektů ASN.1 [28]

Informace jsou poté uloženy do databáze a často jsou vizualizovány pomocí specializovaného nástroje.

Verze protokolu

- SNMP verze 1 (SNMPv1) – jedná se o počáteční verzi SNMP protokolu, která je definovaná v RFC 1157 [5]. Zabezpečení SNMPv1 je založeno na tzv. community-string, což je v podstatě textový řetězec ve formě plain-text (prostý text), který sloužící jako heslo. Každá aplikace založená na SNMP může při znalosti tohoto textového řetězce získat přístup k SNMP informacím na zařízení.
- SNMP verze 2 (SNMPv2) – tato verze přináší několik úprav a vylepšení. Například k metodám Get, GetNext, Set a Trap z protokolu SNMPv1 přidává metody GetBulk a Inform. Rovněž došlo ke změně TRAP PDU formátu, který je od této verze stejný jako PDU ostatních metod (Get, Set, ...). Standard je popsán v RFC 3416 [23]. Veškeré příkazy této verze jsou popsány v tabulce 3.3.
- SNMP verze 3 (SNMPv3) – jedná se o poslední verzi SNMP. Jejím přínosem je zejména vylepšená bezpečnost. Přidává podporu autentizace a šifrované komunikace mezi spravovanými entitami [15].

Tabulka 3.3: Příkazy protokolu SNMP verze 2 [14]

Příkaz	Typ	Směr	Popis
get	0	manager-to-agent	získání hodnoty jednoho či více objektů MIB
get-next	1	manager-to-agent	hodnota dalšího objektu v seznamu/tabulce
get-response	2	agent-to-manager, manager-to-manager	vrací odpověď na předchozí dotaz
set	3	manager-to-agent	nastaví hodnotu jednoho či více objektů MIB
get-bulk	5	manager-to-agent	získání bloku dat
inform	6	manager-to-manager	informace o vzdálené řídící jednotce o přístupových hodnotách v MIB
snmpv2-trap	7	agent-to-manager	informace o asynchronní události

SNMP Trap

Zpráva typu *trap* od agenta směrem k řídící stanici je asynchronní notifikace používaná k oznámení náhlé změny, o které by měl být administrátor neprodleně informován. Příkladem takové situace může být změna stavu síťového rozhraní, příliš vysoká teplota, dosažení určité hodnoty zaplnění disku apod. Reakcí na zprávu typu trap může být např. odeslání e-mailové zprávy administrátorovi, nebo může být spuštěn nějaký skript, který například provede restartování služby. Vždy záleží na konkrétní situaci a konkrétním problému.

Jak již bylo zmíněno, trap funguje na portu 162 a využívá komunikační protokol UDP. Jedná se tedy o *nespolehlivou* službu a odesílatel si nemůže být stoprocentně jistý, že zpráva byla doručena. Přijetí SNMP trap se nijak nepotvrzuje [15].

3.5 Shrnutí

V této kapitole byla představena architektura a komponenty SNMP Frameworku. Byla zde popsána funkce SNMP manažera v roli řídící stanice a také role SNMP agenta. Následně bylo vysvětleno, jak je definována struktura monitorovaných dat pomocí SMI a také to, jak jsou objekty uloženy v databázi MIB. Poté byla popsána stromová struktura MIB včetně skupin objektů. Dále byl popsán samotný protokol SNMP, jeho verze a rozdíly mezi nimi. Na závěr byl představen speciální typ zprávy SNMP Trap.

V další kapitole se zaměřuji na parametry, které by se měly brát v úvahu při výběru nástroje k monitorování počítačové sítě, vyberu jeden z dostupných nástrojů a představím konfiguraci nejdůležitějších komponent.

Kapitola 4

Konfigurace systému pro monitorování pomocí SNMP protokolu

Tato kapitola se bude zabývat výběrem a nasazením vhodného nástroje pro monitorování sítě, který využívá ke své činnosti protokol SNMP. Dále popíši požadavky na systém, které veličiny je možné monitorovat a popíši proces konfigurace klíčových komponent vybraného systému, kterými jsou přidání nového zařízení a aktivace mnou vytvořených modulů k monitorování anomální síťové aktivity sledovaných zařízení.

4.1 Výběr vhodného nástroje

Vzhledem ke stoupající komplexitě počítačových sítí je potřeba brát v potaz to, že počet zařízení v síti je různý, zařízení mohou být různých typů, od různých výrobců. Může se jednat o koncové stanice, mobilní telefony, tablety, routery, servery, tiskárny atd. Nabízí se tedy otázka, jaký nástroj k monitorování sítě použít. Při výběru vhodného nástroje potřeba brát v úvahu následující kritéria:

- zda vybraný nástroj je či není open-source,
- zda je vybraný produkt zdarma, funguje na bázi předplatného, či je potřeba jednorázově zakoupit licenci,
- zařízení jakého typu máme v plánu monitorovat,
- operační systém monitorovaných zařízení,
- zda je nástroj aktivně vyvíjen,
- jaký bude počet monitorovaných zařízení,
- jaké veličiny je potřeba monitorovat,
- jaké nabízí nástroj možnosti upozornění administrátora a zda je schopen ho automaticky v případě potřeby kontaktovat.

Ačkoliv na trhu existuje nepřeberné množství nástrojů pro komplexní monitorování síťové infrastruktury, tak řada těchto nástrojů se odchýlila pryč od využívání SNMP a na místo toho využívá své proprietární protokoly, např. Microsoft se snaží prosadit svůj protokol WMI (Windows Management Instrumentation). S ohledem na povahu této práce, kterou je detekce anomálií na základě SNMP komunikace, se budu zabývat pouze nástroji, které využívají technologii SNMP.

V příloze C je uvedeno stručné porovnání existujících nástrojů. Na základě tohoto porovnání jsem se rozhodl využít nástroj *LibreNMS*, pro který budu vyvíjet modul, který bude rozpoznávat anomální provoz na síti a v případě známého síťového útoku provede jeho klasifikaci.

4.2 Požadavky na systém

Nástroj LibreNMS, vzhledem ke své rozsáhlé databázi MIB souborů pro zařízení od různých výrobců (včetně různých typů zařízení, příkladem koncové stanice, routery, tiskárny aj.), nabízí poměrně komplexní možnosti monitorování prostřednictvím protokolu SNMP. Dokáže monitorovat a přehledně zobrazovat zobrazovat metriky typu:

- Základní informace – název, operační systém, uptime, připojená zařízení.
- Využití systémových zdrojů – procesor, virtuální a fyzická vnitřní paměť, obsazenost pevných disků.
- Síťové statistiky – síťová rozhraní a jejich provoz, ARP tabulka.

Všechny tyto získávané metriky ukládá do databáze typu *round-robin* nástroje *RRDTool*, která je blíže popsaná v podkapitole 5.2.2. Obsahuje rovněž propracovaný systém vytváření tzv. *alertů*, jde o hlášení, která budou vyvolána na základě splnění nastavené podmínky. Klíčovou komponentou je možnost aplikace pluginů nástroje Nagios. Posledních zmíněných vlastností budu využívat při vytváření a implementaci mých pluginů pro detekci anomálního provozu a klasifikaci síťového útoku. Tomuto tématu se věnuji podrobně v kapitole 5.

4.3 Instalace a poinstalační konfigurace

Dle oficiální dokumentace nástroje[1] existují 3 možnosti instalace. První možností, která se nabízí, je využít nástroj Docker a spuštění instance nástroje v kontejneru. Druhou možností je provést čistě manuální instalaci, která zahrnuje vytvoření uživatele, nastavení databáze, firewallu, webového serveru a dalších. Poslední možností je využití obrazu systému v nástroji VirtualBox, který obsahuje již nainstalovaný systém LibreNMS. Vzhledem k okolnostem jsem se rozhodl využít poslední možnosti, která mi umožňuje pohodlně vytvářet zálohy systému.

Po instalaci systému (ze kteréhokoliv zdroje) je třeba provést určité úpravy konfigurace. Jednou z těchto úprav je změna intervalu získávání dat SNMP (polling) z pěti minut na jednu minutu. Potřebné kroky (které mj. zahrnují i konverzi existujících RRD souborů) jsou přístupné v oficiální databázi systému pod heslem *1 Minute Polling*. Dále je třeba, aby nebyl aktivní nástroj *RRDCached*, protože zápisy do databáze se tímto občas zpožďují a data tak nemusí být dostupná v moment, kdy si o ně bude databáze žádat.

[1] https://docs.librenms.org/Installation/

Zásadní úpravou, kterou je třeba provést, je aktivace *Services*. Jde o aktivace možnosti využívat pluginy systému Nagios. Tato akce sestává z následujících kroků:

1. V souboru `config.php` je třeba přidat tento řádek: `$config['show_services']=1;`
2. Pod tento řádek se umístí lokace, kde se budou pluginy nacházet (lib64 pro CentOS, lib pro Ubuntu):

   ```
   $config['nagios_plugins']="/usr/lib64/nagios/plugins";
   ```

3. Všechny soubory tohoto adresáře se nastaví jako spustitelné příkazem:

   ```
   chmod +x /usr/lib64/nagios/plugins/*
   ```

4. Nakonec je třeba přidat informaci o souboru *services-wrapper.py* do souboru cron, aby docházelo k periodickému spouštění aktivních pluginů. Konkrétně se do cronu přidá tento řádek:

   ```
   * * * * * librenms /opt/librenms/services-wrapper.py 1.
   ```

Kompletní proces je rozepsán detailněji v dokumentaci nástroje LibreNMS. (Poznámka: nejnovější obraz systému z nástroje VirtualBox již obsahuje patřičnou konfiguraci k vytvoření Services, při využití tohoto nástroje stačí provést pouze poslední krok.)

4.4 Přidání nového zařízení

Proces přidání nového zařízení je v LibreNMS poměrně jednoduchý a přímočarý. Po kliknutí na položku *Devices* se otevře rozbalovací menu, ve kterém je třeba kliknout na tlačítko *Add Device*. Následně je potřeba vyplnit informace o zařízení. Příklad konfigurace je uveden v obr. 4.1.

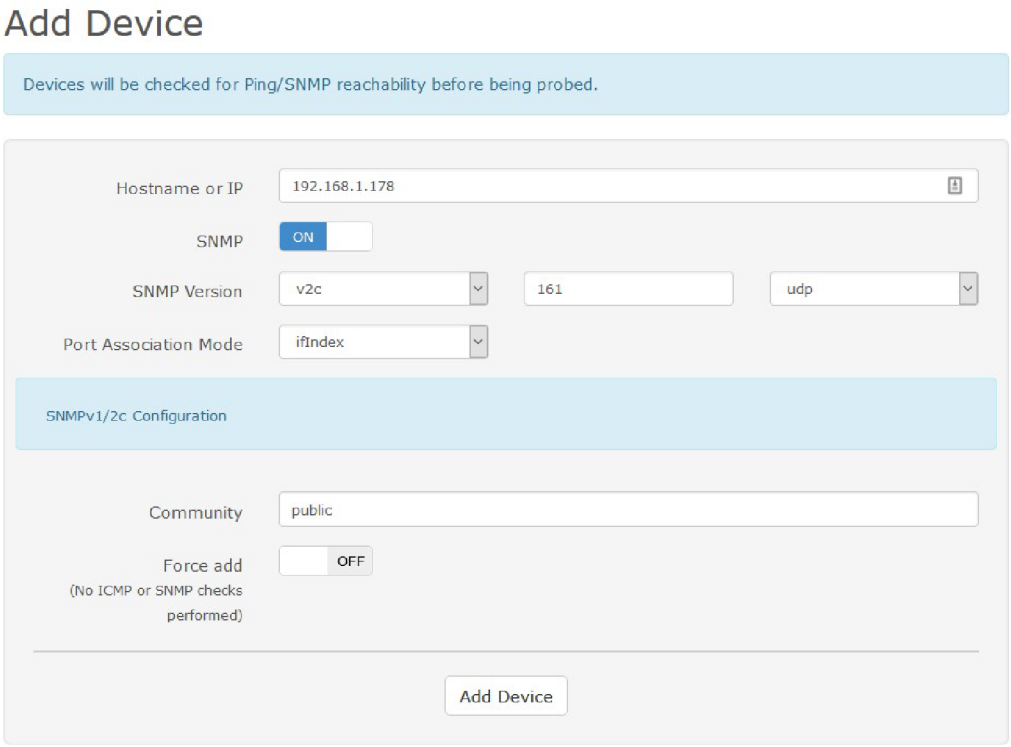
Ve formuláři je třeba vyplnit IP adresu zařízení nebo jeho hostname, dále verzi SNMP společně s portem a typem protokolu, port association mode, který uloží interface na základě jeho indexu v monitorovaném zařízení a v neposlední řadě komunitní string. Tlačítkem *Add Device* dojde k navázání spojení s hostem a k jeho monitorování, na pozadí dojde k jeho přidání do databáze a vytvoření RRD souborů, kde se budou ukládat získané a zobrazované metriky.

4.5 Nastavení alertů u zařízení

LibreNMS obsahuje velice flexibilní systém varování a odesílání upozornění v případě, že určité veličiny se nachází ve vybraném stavu. Systém rovněž uchovává historii hlášení, tak je možné zpětně dohledat a zobrazit, jaký byl stav modulů v minulosti a kdy došlo ke změně jejich stavu.

Na následujícím příkladě vysvětlím vytvoření upozornění, které nastane v okamžiku, kdy můj modul pro detekci útoků hlásí detekci útoku UDP flood. Následují příklady hodnot, které je třeba vyplnit pro vytvoření hlášení i pro ostatní typy útoků. Nakonec této podkapitoly je doplněna ukázka pravidel pro vytvoření upozornění i při detekci anomálního provozu mým druhým modulem.

Po stisknutí tlačítka *Alerts*, které se nachází v horním panelu je třeba pokračovat stiskem tlačítka *Alert rules*. Na této obrazovce se nachází přehled všech vytvořených pravidel, je zde rovněž možnost tato pravidla vypnout, editovat či smazat. K vytvoření nového pravidla



Obrázek 4.1: Příklad manuálního přidání nového zařízení

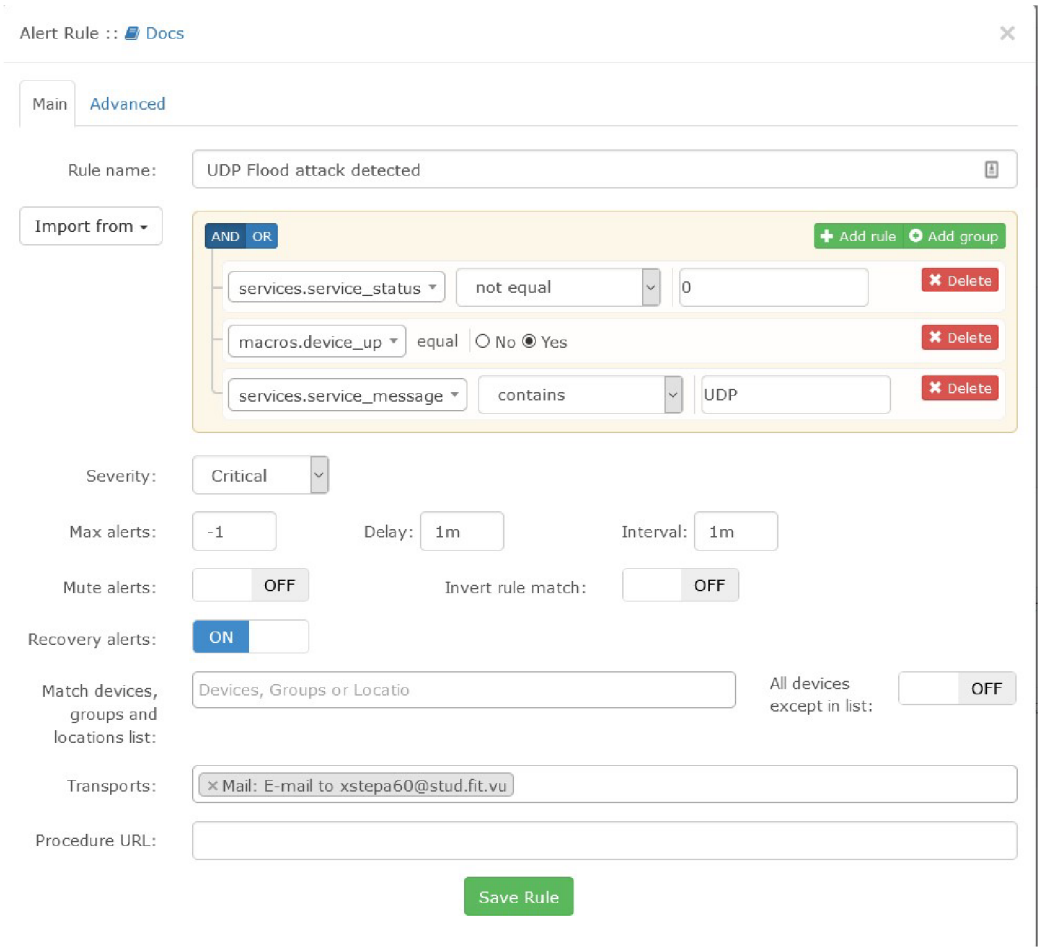
je třeba stiknout tlačítko *+ Create new alert rule*. Nyní se zobrazí formulář, ve kterém se nastavují veškeré vlastnosti upozornění. Konkrétní nastavení z mého příkladu je zobrazeno v obr. 4.2.

Prvním řádkem formuláře je *Rule name*, jedná se o jméno pravidla. Následuje nastavení sady pravidel. Pravidlo *services.service_status not equal 0* znamená, že alarm bude vyvolán, pokud status služby je cokoliv jiného než *OK*. Pravidlo je zkombinováno s pravidlem *macros.device_up equal Yes*, které říká, že dané zařízení musí být zároveň zapnuté a dostupné. Poslední přidružené pravidlo *services.service_message contains UDP* vytváří kontrolu obsahu zprávy modulu, která se vytváří na základě typu detekované anomálie.

V případě detekce útoku typu UDP flood je obsah zprávy modulu "UDP Flood attack detected", proto na ni bude toto pravidlo reagovat. Kolonka *Severity* nastavuje příznak vážnosti upozornění. Hodnota *Max alerts* nastavuje maximální počet vyvolaných upozornění po dobu trvání podmínky pro vyvolání upozornění. *Delay* nastavuje zpoždění. Pokud je splněná podmínka pro vyvolání upozornění, ale během této doby uvedené v *Delay* nastane opět normální stav, tak alarm vyvolán nebude. Zde minimální nastavitelná hodnota je 1 minuta, protože to je zároveň frekvence s jakou se získávají nová data. *Interval* značí s jakou frekvencí dojde k vytvoření nové notifikace, pokud je alert stále aktivní.

Dalšími důležitými údaji je kolonka *Match devices, groups and locations list*, kde lze nastavit, aby se toto upozornění vyvolávalo pouze u zařízení z konkrétní skupiny zařízení. Kolonka *Transports* nastavuje způsob notifikace vybraného uživatele nebo skupiny uživatelů. Může jít o notifikaci prostřednictvím e-mailu, či nějaké aplikace, např. Slack. Tlačítkem *Save Rule* se pravidlo uloží a od této doby je aktivní.

Analogicky je potřeba vytvořit upozornění i na ostatní útoky, tyto se budou lišit v poli *Rule name* a budou následující: "TCP-SYN Flood attack detected", "ICMP Flood attack de-



Obrázek 4.2: Nastavení upozornění při detekci útoku UDP flood

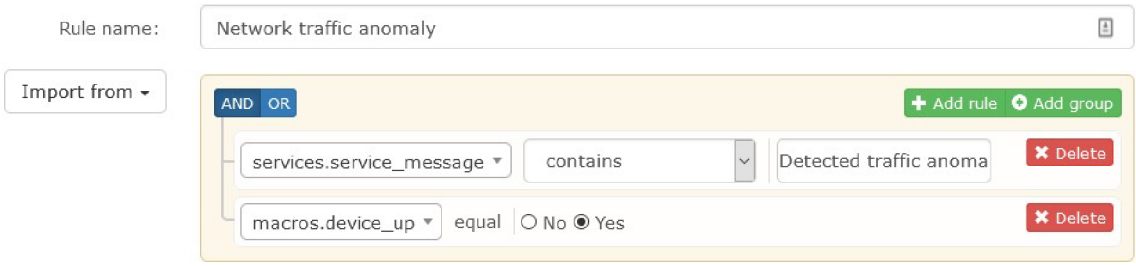
tected", "SSH Bruteforce attack detected", "Slowloris attack detected", také dojde k úpravě v poli *services.service_message*, které se přizpůsobí ostatním útokům a jeho hodnota bude tedy vždy jedna z následujících: "TCP-SYN", "ICMP", "SSH Bruteforce", "Slowloris".

Podobně se musí nastavit i *Alert Rule* pro druhý modul, který provádí detekci anomálního provozu. V tomto případě jsou hodnoty *Rule name* a sada pravidel vyobrazeny na obr. 4.3. Hodnota položky *services.service_message contains* je "Detected traffic anomaly". Zbytek formuláře zůstane stejný jako v předchozím případě.

4.6 Aktivace modulu detekce anomálií pro zařízení

Jeden ze způsobů, jak aktivovat vytvořený modul pro nějaké zařízení, je vybrat v hlavním menu v záložce *Services* položku *Add Service*. Následně se zobrazí formulář, který je uveden v obr. 4.4. Tento příklad reprezentuje aktivaci mého modulu `ANOMALY_TRAFFIC` pro můj notebook s IP adresou 192.168.1.178.

První položkou formuláře je pole *Name*, ačkoliv je toto pole nepovinné, je lepší si vytvářenou službu pojmenovat. Druhou položkou je rozbalovací menu s názvem *Device*, zde se vybírá stanice, pro kterou se služba, respektive modul aktivuje. Pole *Check Type* slouží k výběru konkrétního modulu, který bude využit. Pole *Description* je nepovinnou položkou,



Obrázek 4.3: Část nastavení Alert Rule pro plugin provádějící detekci anomálního provozu

která slouží k popisu vytvářené služby. Pole *Remote Host* zůstane prázdné, protože zařízení již bylo vybráno dříve v poli *Device*. Můj modul rovněž nevyžaduje žádné parametry, proto pole *Parameters* zůstane prázdné. Zaškrtávací políčko *Ignore alert tag* značí, že se na tuto konkrétní instanci služby u tohoto zařízení nebude brát zřetel v případě kontroly stavu služeb. Zaškrtávací políčko *Disable Polling and Allerting* způsobí to, že nebude docházet k pollingu a v podstatě služba bude neaktivní. K uložení konfigurace modulu pro dané zařízení dojde stisknutím tlačítka *Add Service*.

Pro aktivaci druhého modulu, který klasifikuje útoky, je třeba tento proces opakovat a v poli *Check Type* vybrat hodnotu `ATTACK_CLASSIFICATION`, dle vlastní úvahy lze opět vyplnit i políčka *Name* a *Description*.

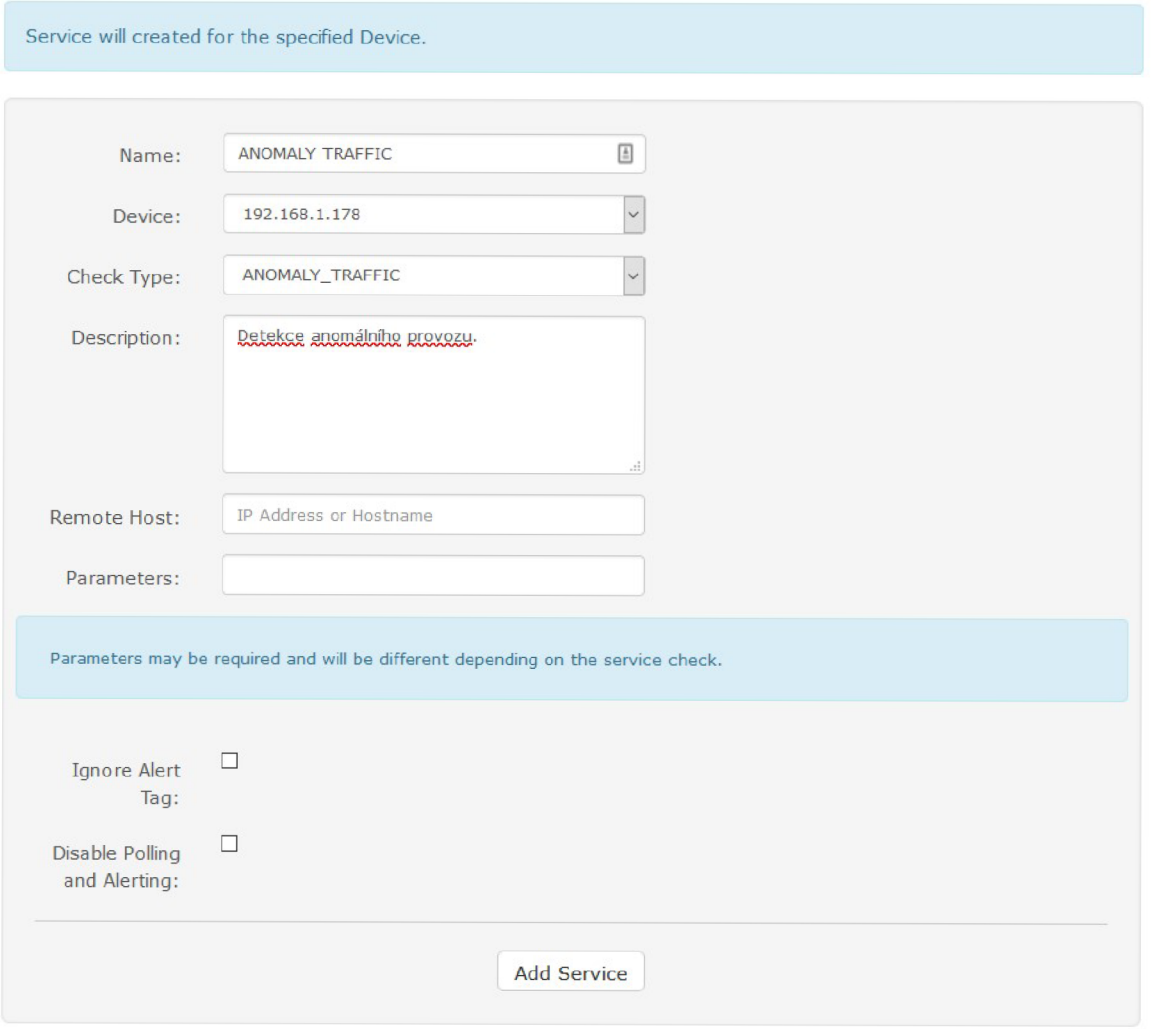
4.7 Zobrazení historie hlášení

Jakmile jsou moduly aktivní, přiřazené vybraným zařízením a zároveň nastavený systém pro vytváření hlášení při změně stavu modulů, což bylo popsáno v předchozích dvou podkapitolách, je možné zobrazit aktivní hlášení či jejich historii. Aktivní hlášení se zobrazí po stisknutí tlačítka *Alerts* a výběru položky *Notifications*. Pro zobrazení historie hlášení je třeba vybrat položku *Alert History*, zobrazí se které hlášení (alert) bylo kdy vyvoláno, jeho vážnost (severity) a pro které zařízení. Historie hlášení je demonstrována v obr. 4.5.

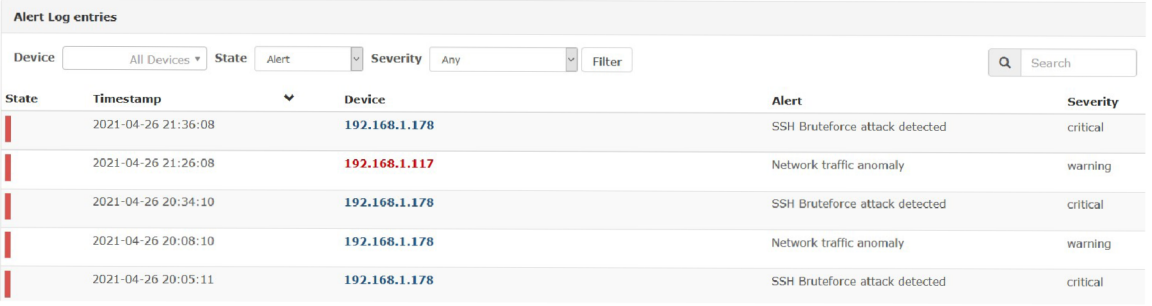
4.8 Shrnutí

V této kapitole jsem popsal kladené požadavky na systém, který využívám ke sběru informací z cílových stanic pomocí protokolu SNMP. Tento systém slouží jako základ do kterého implementuji dva pluginy. První plugin slouží pro klasifikaci známých síťových útoků a druhý pro analýzu anomálního provozu na síti. Po vyjmenování možností, jak lze vybraný systém získat a nainstalovat, jsem popsal prvky, které je potřeba nakonfigurovat. Jmenovitě se jedná zejména o *1 Minute Polling* a aktivaci *Services*. Services jsou v podstatě pluginy, které typově odpovídají pluginům systému Nagios. Dále jsem popsal klíčové prvky, jako přidání nového zařízení k monitorování, aktivace modulů (pluginů) k vybraným zařízením a nastavení *Alert Rules*, aby se vytvářela upozornění na kritické stavy pluginů, kdy došlo k detekci útoku a anomálního provozu. A zároveň se ukládala historie těchto hlášení.

V příští kapitole popíši, jak fungují pluginy v nástroji LibreNMS. Popíši vytvoření tréninkové sady dat a budu se zabývat popisem obou mých modelů, včetně popisu algoritmů, které využívají.



Obrázek 4.4: Konfigurace modulu pro detekci anomálií



Obrázek 4.5: Historie vyvolaných hlášení

Kapitola 5

Implementace modulů pro klasifikaci a detekci anomálií

Tato kapitola bude popisovat návrh a implementaci obou modulů, které provádí analýzu síťového provozu, klasifikaci případného známého útoku a detekci anomálního provozu. Tyto moduly pracují s vybranými daty, které převezmou z databáze nasazeného dohledového nástroje. Data jsou ze skupin IP, TCP, UDP, ICMP ze SNMP databáze MIB.

5.1 Návrh modulů

Pro účely vytvoření mých modulů využiji integrované vlastnosti nástroje LibreNMS využívat pluginy nástroje Nagios. Pluginy nástroje Nagios fungují obecně tak, že provádí kontrolu dostupnosti nějaké služby a následně vrací jeden ze tří stavů:

- OK – návratová hodnota pluginu 0,
- WARNING – návratová hodnota pluginu 1,
- CRITICAL – návratová hodnota pluginu 2.

Následně při každém monitorovaném zařízení v nástroji LibreNMS je zobrazen výčet všech kontrolovaných služeb a jejich poslední známý stav. LibreNMS také umožňuje nastavit pravidla, že pokud nějaká služba bude přepnuta do určitého stavu, dojde k informování administrátora vybraným způsobem.

Této vlastnosti budu využívat při vytváření mých modulů, které budou fakticky v nástroji implementovány jako Nagios pluginy, tudíž musí být koncipovány jako shellový skript a jejich funkce bude následující:

1. Modul bude spouštěn v periodických intervalech.
2. Modul z databáze převezme aktuální vybrané metriky pro danou stanici.
3. Modul s využitím pomocného skriptu zjistí zda došlo k výskytu anomálie, respektive provede klasifikaci síťového provozu.
4. Modul vrátí patřičnou návratovou hodnotu.
5. Výsledek bude v systému LibreNMS patřičně reprezentován, případně dojde k informování administrátora.

Moduly budou umístěny v adresáři `/usr/lib64/nagios/plugins`. Aby mohly být načteny nástrojem LibreNMS, musí jejich název obsahovat prefix `check_`. Proto název prvního modulu, který provádí klasifikaci útoků DoS a využívá model popsaný v sekci 5.3, je `check_ATTACK_CLASSIFICATION`. Název druhého modulu, který detekuje anomální provoz a využívá model popsaný v sekci 5.4, je `check_ANOMALY_TRAFFIC`.

5.2 Vytvoření tréninkové sady dat

Pro správnou funkci klasifikátoru, který bude rozpoznávat jednotlivé typy síťových útoků a řadit je do správných tříd, je potřeba mít k dispozici kvalitní trénovací data, na kterých se model naučí klasifikaci provádět. Tato data budu sám generovat simulací různých síťových útoků převážně typu *odepření služby* (DoS – Denial of Service).

5.2.1 Generování dat – simulace síťového útoku

Mojí snahou je vytvořit takovou datovou sadu, která bude obsahovat označená data, sesbíraná jak za normálního provozu, tak i pod vlivem simulovaných útoků typu DoS. Druhy útoků, které jsem za účelem vytvoření datové sady nasimuloval jsou následující:

1. **TCP SYN flood** – Jedná se o útok, při kterém útočník zahájí komunikaci odesláním zpráv TCP s příznakem SYN, avšak na odpověď serveru k navázání spojení již nereaguje. Spojení tedy zůstane způli navázané a může zahltit TCP modul serveru [27].

   K provedení útoku jsem využil nástroj *hping3*, který umožňuje generovat různé druhy paketů v závislosti na vstupní konfiguraci a parametrech.

   Útok jsem provedl ve dvou iteracích, kdy v první iteraci byl použit příkaz `sudo hping3 -S --flood -p 80 192.168.1.178`, kde parametr *-S* nastavuje SYN tcp flag, parametr *--flood* nastaví odesílání paketů na nejvyšší možnou rychlost, parametr *-p* nastavuje cílový port a posledním vloženým parametrem je cílová IP adresa. Ve druhé iteraci byl použit příkaz `sudo hping3 -S --flood -p 8000 192.168.1.178`. Oba příkazy se liší v cílovém portu, přičemž na cílové stanici byl aktivní jednoduchý Python HTTP server[1] naslouchající na portu 8000, na portu 80 žádná naslouchající služba nebyla. V následujících útocích je situace s porty obdobná.

2. **ICMP flood** – Tento útok, někdy rovněž označovaný jako *ping flood*, spočívá v tom, že útočník zahltí cílové zařízení pakety typu ICMP echo-request (ping). Toto cílové zařízení je následně nuceno odpovědět stejným množstvím paketů, což může vést k zahlcení sítě a nedostupnosti daného zařízení [8].

   Útok byl opět proveden s využitím nástroje *hping3* s následujícími parametry: `sudo hping3 --icmp --flood 192.168.1.178`, parametr *--icmp* indikuje, že budou odesílány paketu typu ICMP a parametr *--flood* nastavuje nejvyšší rychlost odesílání paketů.

3. **UDP flood** – Při tomto útoku může útočník zacílit velké množství paketů typu UDP na jeden či různé porty cílového zařízení, které se vždy snaží najít aplikaci obsluhující daný port a pokud žádný takový nenalezne, odešle zpět zprávu typu *destination unreachable*. Toto opět může způsobit zahlcení daného systému a ten se tak stane

[1] https://docs.python.org/3/library/http.server.html

nedostupným. Rovněž je možné podvrhnout zdrojovou IP adresu, takže návratové ICMP zprávy se zpět k útočníkovy nedostanou [8].

Útok jsem opět provedl s využitím nástroje *hping3* ve dvou iteracích s rozdílnými čísly portu. V první iteraci byl útok proveden s těmito parametry: `sudo hping3 --udp --flood -p 80 192.168.1.178`, kde parametr *--udp* nastavuje odesílání paketů typu UDP a parametrem *--flood* se nastavuje nejrychlejší odesílání paketů. Ve druhé iteraci byl proveden tentýž příkaz, nicméně parametr *-p* byl nastaven na hodnotu 8000, kde na tomto portu naslouchal jednoduchý Python UDP server[2].

4. **SSH Bruteforce** – Bruteforce, neboli útok hrubou silou, je typ útoku, při kterém se testují všechny možné kombinace uživatelského jména a hesla. V této práci jsem provedl útok na službu SSH s využitím nástroje *Hydra*[3].

   Útok byl proveden s následujícími parametry: `sudo hydra -l "daniel stepan"-x 4:4:aA1 192.168.1.178 ssh`, kde parametr *-l* určuje název uživatele, jehož heslo se bude prolamovat. Parametr *-x* značí, že hesla se budou generovat (nebudou brána z žádného wordlistu) dle schématu *4:4:aA1* (MIN:MAX:CHARSET). Předposledním parametrem je IP adresa zařízení. Poslední parametr značí službu, v mém případě tedy SSH.

5. **Slowloris** – Princip útoku Slowloris[4] spočítá v tom, že se otevře spojení, které se následně udržuje po dlouhou dobu otevřené. Server čeká na dokončení HTTP požadavku, který ovšem nikdy ukončen nebude, naopak se serveru podsouvají bezvýznamné části hlavičky, aby spojení zůstalo stále otevřené. Tento útok je velmi obtížné detekovat, protože neútočí na síťová rozhraní, ale probíhá na aplikační vrstvě [12]. Útok byl proveden tímto příkazem: `sudo slowloris -p 8000 192.168.1.178`

5.2.2 Získání potřebných hodnot z databáze

Zde je potřeba zmínit, že ačkoliv LibreNMS potřebuje ke svému provozu klasickou SQL databázi, tak pro ukládání metrik je využíván jiný systém. Pro ukládání metrik je využíván *RRDtool*, což je databáze speciálního typu, vytvořená pro ukládání dat, jenž mají povahu časové řady.

Výhodou i nevýhodou tohoto nástroje je to, že každý vytvořený soubor s uloženými metrikami má vždy konstantní velikost bez ohledu na množství uložených dat. To je způsobeno tím, že jde o cyklickou datovou strukturu a stará data se přepisují za nová. Další vlastností tohoto nástroje je to, že metriky ukládá jako datový typ *counter*, který funguje tak, že data ukládá ve formě "per-second rate", vypočítá a uloží průměrnou hodnotu za sekundu v daném intervalu, pro který jsou data uložená [20].

Pro každou monitorovanou stanici v systému LibreNMS existuje vlastní adresář, který obsahuje řadu RRD souborů, kde v každém RRD souboru je uložena jiná sada metrik. Tento adresář se zpravidla nachází v umístění `/opt/librenms/rrd`.

Získání dat za účelem vytvoření tréninkové sady jsem provedl příkazem `rrdtool dump netstats-ip.rrd dump_ip_group.xml` (stejným způsobem byla získána data ze souborů *netstats-tcp.rrd, netstats-udp.rrd* a *netstats-icmp.rrd*) jakožto právě v souboru *netstats-ip.rrd* jsou uloženy metriky ze skupiny IP databáze MIB. Tento příkaz provede dump data-

[2] https://wiki.python.org/moin/UdpCommunication
[3] https://github.com/vanhauser-thc/thc-hydra
[4] https://github.com/gkbrk/slowloris

báze a vytvoří soubor *dump_ip_group.xml*, ve kterém je popsána struktura uložených dat v RRD souboru a také samotná data, přičemž vše je ve formě XML.

Následně vyberu data v daných časových intervalech, které odpovídají konkrétním provedeným útokům, a tato uložím do samostatných textových souborů. Za účelem převedení dat z formátu XML formátu do formátu CSV jsem si vytvořil jednoduchý skript, který čte soubor typu XML po řádcích a za pomoci regulérního výrazu získá požadovaná data, která uloží do datové struktury typu *seznam*, jehož obsah na závěr uloží do souboru typu CSV a doplní se značení o data ze kterého útoku se jedná. Tato operace se provede pro každý typ útoku a výsledkem je CSV soubor, který obsahuje označená data. S využitím těchto dat proběhne natrénování mého klasifikačního modelu tak, aby byl schopen správně klasifikovat nová data.

Tabulka 5.1 popisuje kolik záznamů pro každý typ útoku je uloženo v CSV souboru určeném pro trénování modelu, stejně tak, kolik záznamů bylo sesbíráno za normálního provozu.

Tabulka 5.1: Počet záznamů v trénovacím souboru pro každý typ útoku. Hodnoty jsou ukládány do databáze každou minutu, jeden záznam tak odpovídá hodnotám získaným během jedné minuty simulace útoku.

Útok	Počet záznamů
TCP SYN flood	313
ICMP flood	149
UDP flood	322
SSH Bruteforce	171
Slowloris	156
Normální provoz	297

5.2.3 Výběr rysů

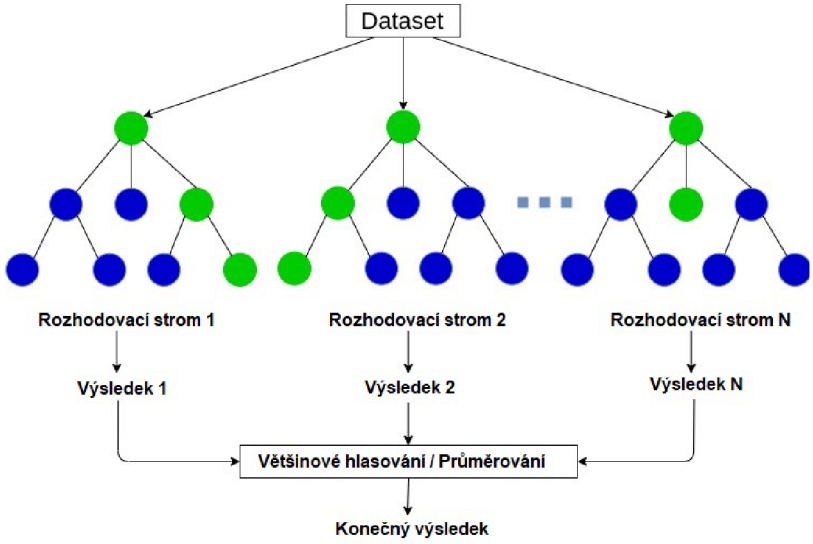
Celkový počet všech proměnných se kterými se pracuje je poměrně vysoký. Skupina IP obsahuje 16 proměnných, skupina TCP obsahuje 1 proměnných, skupina UDP obsahuje 4 proměnné a skupina ICMP obsahuje 26 proměnných. V celkovém součtu jde tedy o 56 proměnných. Takto vysoký počet vstupních proměnných negativně ovlivňuje výkon algoritmu a skriptu. Cílem procesu, který je nazýván *Výběr rysů* (Feature Selection), je vytvořit takovou podmnožinu dat, která nebude obsahovat redundantní či nerelevantní atributy. Výsledkem je určení skóre pro každý rys, v závislosti na jeho důležitosti při probíhajícím procesu klasifikace.

Tuto proceduru využívám v kapitole 6, kde se věnuji analýze zpracovaných dat při jednotlivých útocích. K ohodnocení jednotlivých rysů (proměnných) budu využívat vlastnost `feature_importances_` mého vybraného modelu. Pět nejrelevantnějších proměnných při každém z útoků budu graficky reprezentovat pomocí Python knihovny *matplotlib*.

První modul, pro klasifikaci známých útoků, využívá všechny proměnné, pro které bylo vypočítané skóre vyšší než 0. Seznam těchto proměnných se nachází v příloze D. Druhý modul, který se zabývá detekcí anomálního provozu, pracuje se všemi dostupnými proměnnými ze skupin IP, TCP, UDP a ICMP, což umožňuje detekovat jakoukoliv změnu hodnot a zachytí i takové případy, které první modul nedetekuje.

5.3 Model pro klasifikaci útoků

Při vytváření tohoto modelu jsem se rozhodl využít klasifikační algoritmus *Random Forest Classifier* (v české literatuře občas označovaný jako Náhodný les). Jedná se o souborovou metodu učení, která spočívá v konstrukci mnoha rozhodovacích stromů, kde každý jednotlivý strom provede svoji vlastní predikci třídy. Výslednou predikcí modelu je ta třída, která má mezi výsledky rozhodovacích stromů nejvyšší zastoupení [30]. Funkce algoritmu je ilustrována na obr. 5.1. Důvod, proč jsem zvolil tento algoritmus (a ne jiný z algoritmů uvedených v podkapitole 2.5) je ten, že disponuje vysokou mírou tzv. *explainability*, kdy celý proces a výsledek rozhodování je velmi srozumitelný a je zřejmé na základě kterých hodnot se model primárně rozhoduje. Během procesu trénování si model ukládá informace o každé zpracovávané proměnné (feature). Po ukončení procesu trénování je každé proměnné přiřazeno skóre, tzv. *Feature importance*, které vyjadřuje, jak moc je daná proměnná relevantní během procesu rozhodování [9].



Obrázek 5.1: Princip klasifikace algoritmu Náhodný les (Random Forest) [25]

Implementace mého modelu je provedena s využitím programovacího jazyka *Python* a pokud jde o konkrétní implementaci algoritmu náhodný les, tak pro potřeby tohoto projektu využiji implementaci `RandomForestClassifier` z knihovny *scikit-learn*.

5.3.1 Skript pro vytvoření a trénink klasifikačního modelu

V této části bude popsána funkce skriptu *attack_classification_training.py*, ve kterém je vytvořen a natrénován model pro klasifikaci útoků DoS. Výstupem skriptu je potom tzv. dump tohoto modelu, neboli převod do souborové podoby za účelem další práce s tímto modelem. V obecné rovině jsou činnosti prováděné skriptem následující:

1. Načtení potřebných knihoven.
2. Načtení dat z CSV souboru, který obsahuje data pro trénování modelu.
3. Sloupcům dat v souboru se přiřadí názvy sloupců.

4. Datová sada se rozdělí na nezávislé (features) a závislé proměnné (sloupec reprezentující třídu, resp. typ útoku).

5. Datová sada se rozdělí s využitím funkce `train_test_split()` na data určená k trénování a testování.

6. S využitím třídy `StandardScaler` se provede *feature scalling* (škálování funkcí). Jde o důležitou část procesu předzpracování dat, během které dojde k normalizaci jejich rozsahu, což zvyšuje účinnost některých algoritmů strojového učení.

7. K vytvoření klasifikátoru vytvořím objekt třídy `RandomForestClassifier` a k jeho natrénování využiji funkci klasifikátoru **fit()**.

8. Následně mohu vytvořit predikce z části dat, které byly určeny k testování. A z výsledků vytvořit matici záměn, tímto při vytváření modelu dostávám okamžitou zpětnou vazbu k jeho funkčnosti.

9. Za účelem dalšího zpracování a klasifikace nových dat, provedu uložení objektu klasifikátoru a objektu pro škálování a jejich převod do souborové podoby. Převod se provede metodou `dump()` z knihovny *joblib*.

Skript je podrobně dokumentován pomocí komentářů přítomných ve zdrojovém souboru.

5.3.2 Skript pro využití modelu ke klasifikaci nových dat

Tento skript, pojmenován *attack_classification.py*, je hlavní součástí vytvořeného modulu `ATTACK_CLASSIFICATION`. Slouží k získávání aktuálních metrik z RRD databáze. Tyto hodnoty vloží do připraveného modelu a výsledkem je třída, do které se s nejvyšší pravděpodobností řadí provedený útok. Funkce skriptu je následující:

1. Načtení potřebných knihoven.

2. Kontrola vstupních argumentů.

3. Získání dat z RRD databáze obsahující metriky. V tomto případě se ovšem k získání dat použije příkaz *fetch* nástroje *rrdtool*, který požadované informace vypíše na standardní výstup, který je zachycen a data jsou zpracována s využitím regulárních výrazů.

4. Užitím metody `load` z knihovny *joblib* dojde k načtení uloženého modelu klasifikátoru a rovněž k načtení uloženého škálovače (scaler).

5. Hodnoty v *seznamu* obsahujícího aktuální metriky se normalizují s využitím škálovače a následně dojde k předpovědi a klasifikaci metrik do jedné z tříd.

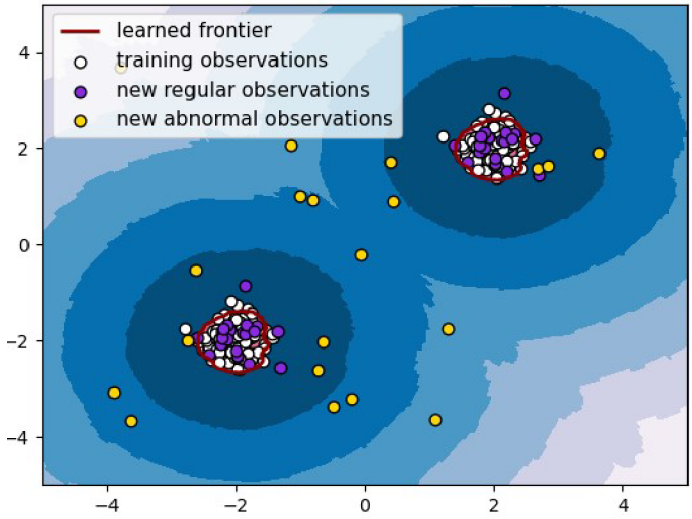
6. Výsledek klasifikace je vypsán na standardní výstup programu. Tento výstup je zachycen Bash skriptem, který tento Python skript spustil, a je dále zpracován v rámci mého modulu klasifikace útoků. Modul na základě získané predikce odešle nástroji LibreNMS jeden ze stavů popsaných v podkapitole 5.1, na základě kterého je potom vyvolán patřičný alarm.

Skript je podrobněji zdokumentován komentáři přítomnými ve zdrojovém souboru.

5.4 Model pro detekci anomálního provozu

Druhý vytvářený modul se zabývá detekcí anomálního provozu, nerozlišuje jednotlivé typy útoků, pouze podává binární informaci o tom, zda nově získané aktuální hodnoty odpovídají hodnotám normálního provozu. Tento proces se v odborné terminologii označuje jako *Novelty detection* [18]. V mém modelu jsem se rozhodl využít algoritmus *Local Outlier Factor* z knihovny *scikit-learn*. Pro tento algoritmus jsem se rozhodl z toho důvodu, že je schopen rozpoznat, zda nový vzorek dat patří do stejné skupiny jako data, na kterých byl natrénován, což odpovídá mým potřebám k řešení tohoto problému. Ze své podstaty se jedná o kombinaci učení s učitelem a učení bez učitele, protože model je trénován pouze na známých datech, které odpovídají normálnímu síťovému provozu. Snahou je zjistit, zda nový vzorek dat patří do stejné skupiny či ne.

Tento algoritmus využívá stejnojmenného principu, kde hodnota Local Outlier Factor se dá označit za "skóre anomálnosti"zkoumaného prvku. Toto skóre v podstatě vyjadřuje jak moc je prvek izolovaný od n nejbližších sousedních prvků [13]. Graficky je princip vyznačen v obr. 5.2.



Obrázek 5.2: Graf zobrazující princip Local Outlier Factor (LOF). Hodnoty, které jsou izolované od ostatních prvků jsou označeny jako anomální [21]

5.4.1 Skript pro vytvoření a trénink modelu detekce anomálního provozu

Podobně jako v předchozím případě i tento modul obsahuje první skript, nazvaný *traffic_anomaly_training.py*, ve kterém je vytvořen a natrénován vybraný model. V tomto případě se jedná o jednoduchý skript provádějící následující činnosti:

1. Načtení potřebných knihoven.
2. Načtení souboru CSV, který obsahuje data sesbíraného za normálního provozu.
3. Vytvoření objektu třídy `LocalOutlierFactor` a jeho natrénování.
4. Vytvoření binární kopie objektu pro pozdější zpracování dalším skriptem v rámci mého modulu.

5.4.2 Skript pro využití modelu ke klasifikaci nových dat

Druhý skript *traffic_anomaly_detection.py* je hlavní součástí druhého vytvořeného modulu `ANOMALY_TRAFFIC` a získává aktuální hodnoty metrik z RRD databáze. Informace předá načtenému modelu jehož výsledkem je binární informace o tom, zda charakteristika aktuálního provozu na síti odpovídá normálnímu či anomálnímu provozu. Funkce skriptu je následující:

1. Načtení potřebných knihoven.
2. Kontrola vstupních argumentů.
3. Získání dat z RRD databáze obsahující metriky.
4. Metodou `load` z knihovny *joblib* dojde k načtení uloženého modelu pro detekci anomálního provozu.
5. Model zpracuje nový vzorek dat a vypočítá hodnotu Local Outlier Factor.
6. Vypočtená hodnota je porovnána s nastavenou hranicí a pokud tato hranice je překročeno, skript vypíše na standardní výstup informaci, že zkoumaný prvek je anomální.
7. Výstup je zachycen a zpracován mým modulem, který tento skript spustil, a na základě získané informace odešle nástroji LibreNMS jeden ze stavů popsaných v podkapitole 5.1, na základě kterého je poté vyvolán patřičný alarm.

Skript je podrobněji zdokumentován komentáři přítomnými ve zdrojovém souboru.

5.5 Shrnutí

Na začátku této kapitoly je popsána principiální funkčnost obou mých modulů, které jsou implementovány v systému LibreNMS a odpovídají svojí strukturou pluginům systému Nagios. To znamená, že přebírají vstupní parametry, kterými je název stanice se kterou plugin v daném cyklu pracuje a výstupem je jeden ze stavů popsaných v podkapitole 5.1.

Pro detekci známých typů útoků se dá použít klasifikace jakožto způsob detekce anomálií a jejich zařazení do správné třídy. K tomuto procesu je ovšem potřeba mít kvalitní data se kterými model bude pracovat. Za tímto účelem byla nasimulována řada síťových útoků, které jsou detailněji popsány v podkapitole 5.2. Tato podkapitola dále popisuje proces získání dat z databáze RRDTool a proces Výběr rysů (Feature Selection), který je dále popsán u každého simulovaného útoku v kapitole 6.

Druhá polovina této kapitoly popisuje v podkapitolách 5.3 a 5.4 oba dva vytvářené moduly pro nástroj LibreNMS. Popisuji principy využitých modelů a pomocné skripty, které slouží k jejich trénování a aplikování v provozu v reálném čase.

Kapitola 6

Analýza získaných dat a testování

V této kapitole bude provedena analýza dat získaných během testování síťových útoků, na závěr bude provedeno testování obou modulů na provozu v reálném čase. Cílem této analýzy je zjistit, jakou roli mají jednotlivé proměnné ze skupin IP, TCP, UDP a ICMP při detekci provedených útoků.

V určitých případech je možné rozlišit anomální chování pouhým okem, zejména pokud máme k dispozici grafickou vizualizaci dat. Z toho důvodu se pokusím popsat vlastnosti jednotlivých útoků na odpovídajících grafech, které mám k dispozici z nástroje LibreNMS.

Dále se budu snažit vyvodit závěr, pomocí procesu *Výběr rysů*, které proměnné jsou nejdůležitější při klasifikaci každého z testovaných typů útoků a nakonec provedu numerickou analýzu hodnot, abych zjistil, jak se jejich hodnoty pod vlivem útoku liší od hodnot za normálního provozu.

Popis všech nenulových proměnných ze SNMP skupin IP, TCP, UDP a ICMP (které se ukládají do databáze nástroje LibreNMS a jsou využívány ke klasifikaci), společně s jejich názvem a identifikátory OID, je uveden v tabulkách v příloze D.

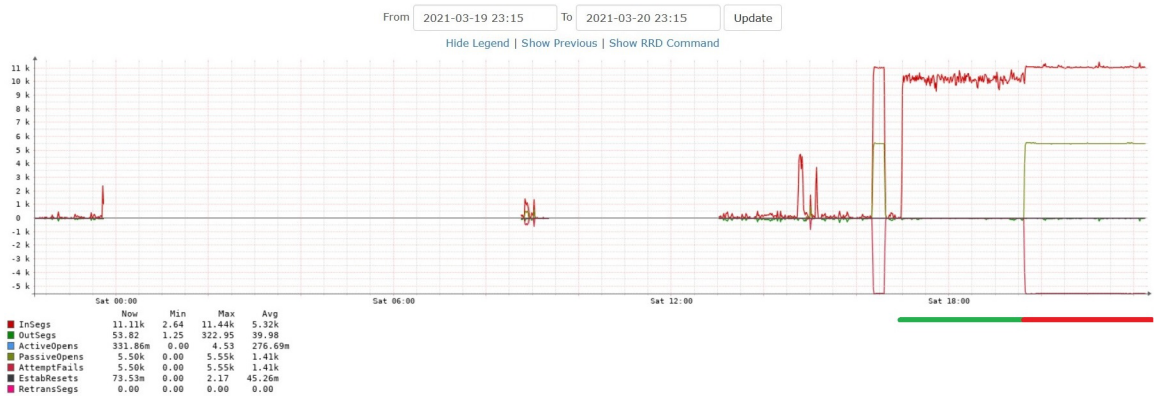
6.1 Analýza útoku TCP-SYN flood

Grafická analýza

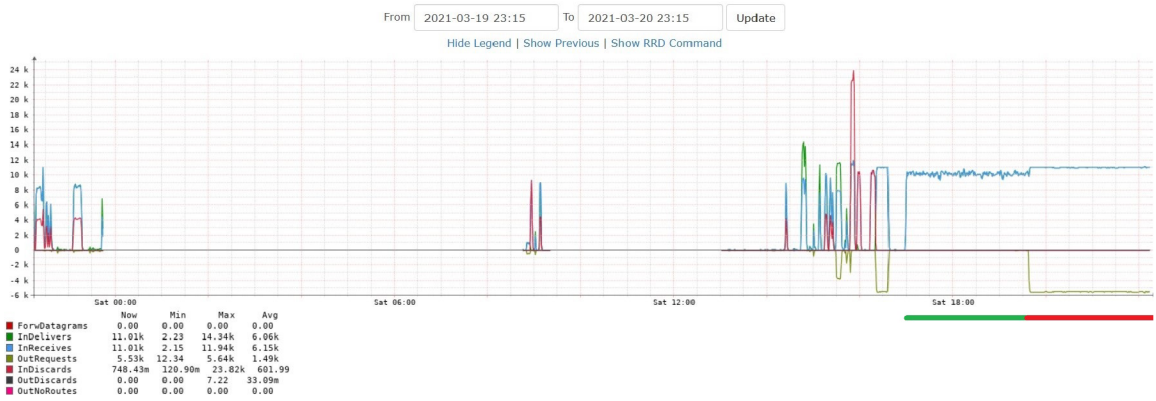
Útok typu TCP-SYN flood má již ze své povahy nejvyšší vliv na SNMP skupiny IP a TCP, ostatní skupiny byly ovlivněny minimálně, proto je v této sekci nebudu uvádět. V prvním obr. 6.1 je uveden graf z nástroje LibreNMS zobrazující vybrané proměnné ze skupiny TCP.

Část označená zelenou linií značí útok, který byl proveden na uzavřený port 80. Druhá část, označená červenou linií, značí část útoku, který byl prováděn na port 8000, na kterém naslouchal HTTP server. Z grafu je patrné, že v obou případech byla značně ovlivněna hodnota `InSegs`, naopak pouze ve druhém případě byla citelně ovlivněna i hodnota `PassiveOpens`.

Ve druhém obr. 6.2 reprezentujícím graf s proměnnými ze skupiny IP lze pozorovat podobný trend. V obou případech byla ovlivněna hodnota `InReceives`, ovšem pouze ve druhém případě, při útoku, který směřoval na naslouchající port 8000, došlo k nárůstu u hodnoty `OutRequests`.



Obrázek 6.1: Graf zobrazující vybrané proměnné skupiny TCP během útoku TCP-SYN flood



Obrázek 6.2: Graf zobrazující vybrané proměnné skupiny IP během útoku TCP-SYN flood

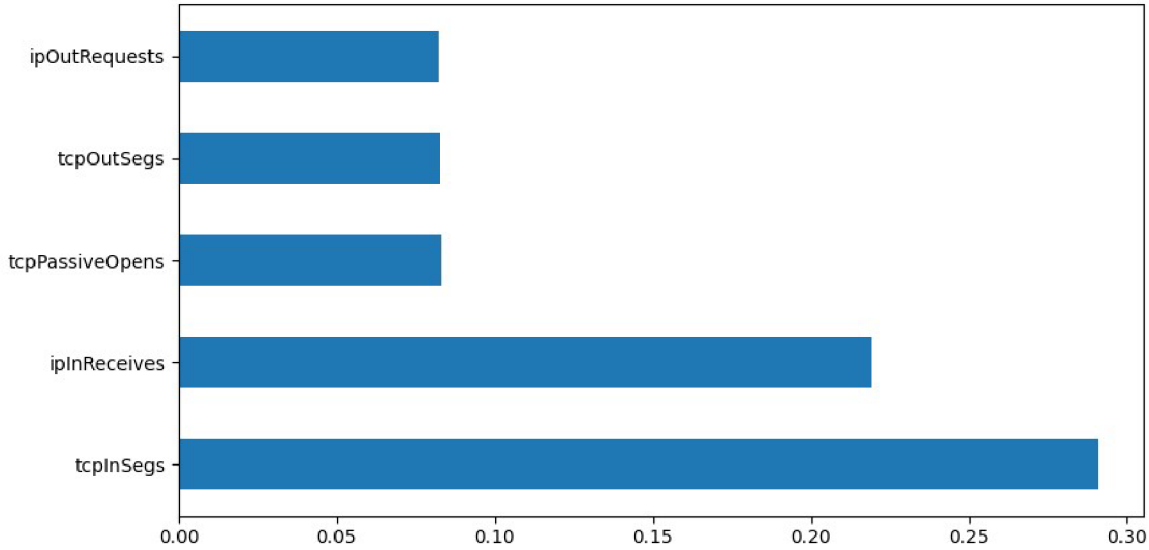
Výběr rysů

Jak již bylo řečeno v sekci 5.2.3, všechny SNMP skupiny se kterými pracuji v této práci dohromady obsahují velké množství proměnných, což má vliv zejména na výkon klasifikačních algoritmů. Cílem tohoto procesu je tedy vyřadit takové proměnné, které jsou buď duplicitní, redundantní, či mají velmi nízkou informační hodnotu.

V této sekci budu porovnávat a zjišťovat vliv proměnných pouze na datech sesbíraných za normálního provozu a datech sesbíraných pod vlivem útoku TCP-SYN flood. Data, se kterými pracuji v této sekci, jsou uložena v souboru *normal_tcp_all_vars.csv*. Soubor obsahuje všechny proměnné ze SNMP skupin IP, TCP, UDP a ICMP, v celkovém součtu se jedná o 55 proměnných.

Tento soubor použiji jako vstupní soubor do mého modelu, jenž se nachází v souboru *multiclass_classification_snmp_all_groups_all_vars.py*, po natrénování modelu vykreslím hodnoty, které jsou uloženy v proměnné modelu `feature_importances_`. Stejný postup budu aplikovat při analýze ostatních útoků.

Obr. 6.3 obsahuje výsledný graf s 5 hodnotami seřazenými dle důležitosti při klasifikaci normálního provozu a provozu pod probíhajícím útokem TCP-SYN flood. Z grafu je patrné, že mezi 5 nejdůležitějších proměnných se řadí: `tcpInSegs`, `ipInReceives`, `tcpPassiveOpens`, `tcpOutSegs`, `ipOutRequests`.



Obrázek 6.3: 5 nejdůležitějších SNMP proměnných při detekci útoku TCP-SYN flood

Tabulka 6.1: Průměr a směrodatná odchylka 5 vybraných proměnných při útoku TCP-SYN flood

Operace	Proměnná	Normální provoz	TCP-SYN flood útok
Průměr	ipInReceives	7.67	10592.67
	ipOutRequests	17.91	2820.70
	tcpPassiveOpens	0.03	2795.33
	tcpInSegs	26.68	10639.90
	tcpOutSegs	13.54	28.97
Směrodatná odchylka	ipInReceives	11.32	473.10
	ipOutRequests	10.23	2757.96
	tcpPassiveOpens	0.044	2755.45
	tcpInSegs	58.12	479.34
	tcpOutSegs	30.26	29.17

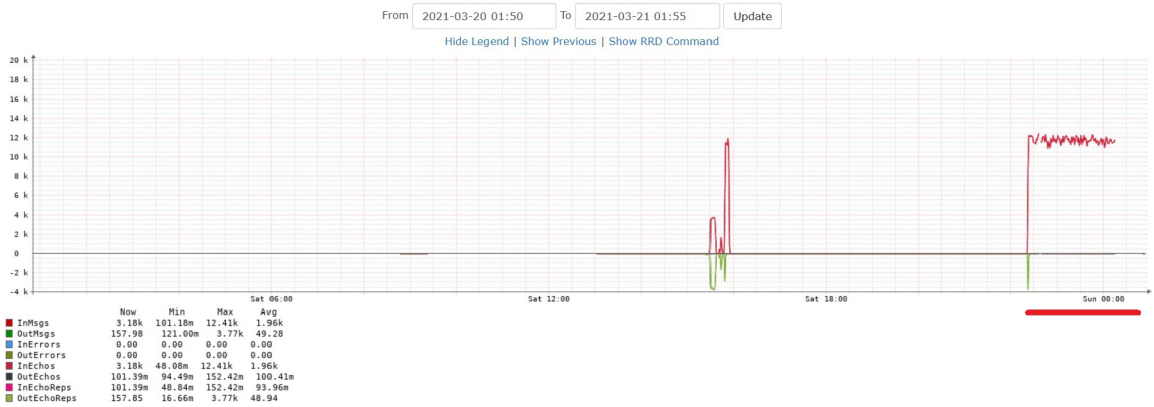
Numerická analýza

V předchozí sekci jsem zjistil, kterých 5 proměnných má nejvyšší vliv během procesu klasifikace útoku TCP-SYN. V této sekci na těchto pěti proměnných provedu matematické operace průměr a směrodatná odchylka, abych zjistil, jaká je hodnota proměnných pod vlivem útoku a jak se jejich hodnoty liší od normálního provozu. Výsledek je uveden v tabulce 6.1. Z tabulky je možné vyčíst, že k nejvyšším nárůstům hodnot došlo u proměnných `ipInReceives` a `tcpInSegs`. Vzhledem k tomu, že nárůst hodnot proměnné `inInReceives` je typický pro všechny záplavové útoky DoS, tak nejvýznamnější proměnnou, která charakterizuje přímo útok TCP-SYN flood je proměnná `tcpInSegs`.

6.2 Analýza útoku ICMP flood

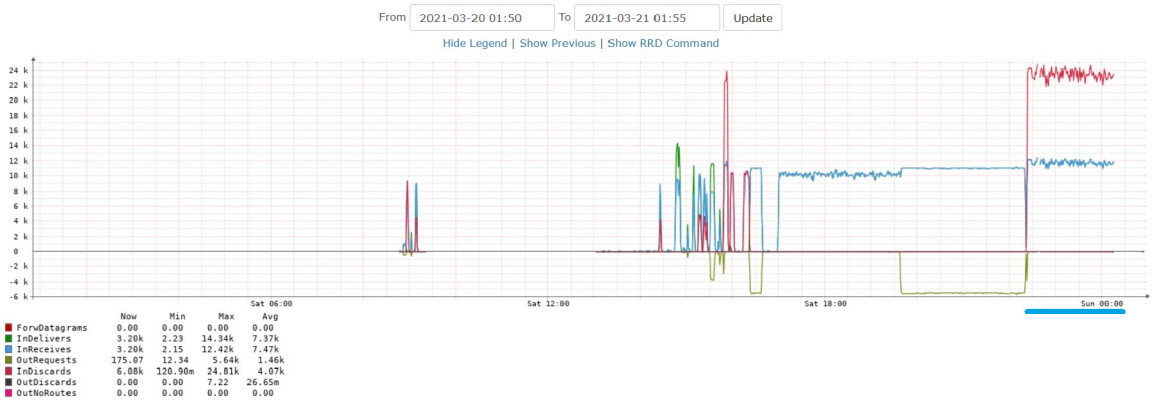
Grafická analýza

Největší vliv útoku ICMP flood lze pozorovat na hodnotách ze skupin ICMP a IP. V grafu 6.4, který reprezentuje skupinu ICMP, je útok vyznačen červenou spodní čárou. Při pohledu na graf je možné zřetelně vidět, že útokem nejvíce ovlivněná proměnná je `InEchos`.



Obrázek 6.4: Graf zobrazující vybrané proměnné skupiny ICMP během útoku ICMP flood

Ve druhém grafu 6.5 reprezentujícím skupinu IP se pozoruje nárůst hodnot dvou proměnných. Zcela nejvyšších hodnot dosahuje proměnná `InDiscards`, druhou nejvíce ovlivněnou proměnnou je `InReceives`, která oproti proměnné `InDiscards` dosahuje přibližně poloviních hodnot.

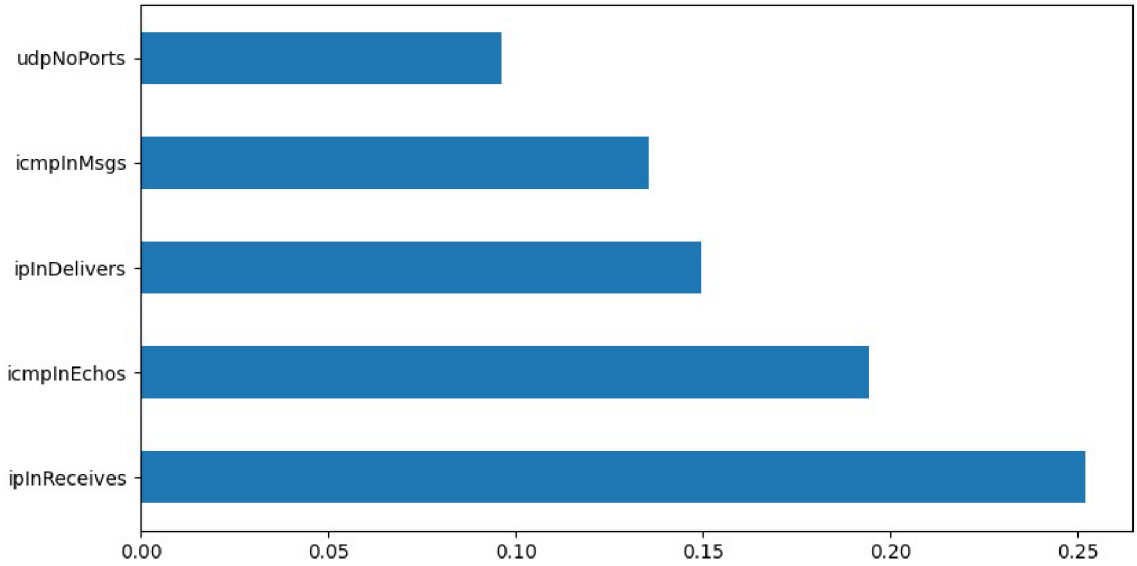


Obrázek 6.5: Graf zobrazující vybrané proměnné skupiny IP během útoku ICMP flood

Výběr rysů

V této sekci opět porovnám vliv proměnných na datech sesbíraných za normálního provozu a na datech sesbíraných pod vlivem útoku ICMP flood. Data, která budu zpracovávat jsou uložena v souboru *normal_icmp_all_vars.csv*.

Jak je patrné z výsledku klasifikátoru v obr. 6.6, mezi 5 nejvlivnějších proměnných se řadí `ipInReceives, icmpInEchos, ipInDelivers, icmpInMsgs, udpNoPorts`.



Obrázek 6.6: Graf zobrazuje 5 nejdůležitějších SNMP proměnných při detekci útoku ICMP flood

Tabulka 6.2: Průměr a směrodatná odchylka 5 nejužitečnějších proměnných při útoku ICMP flood

Operace	Proměnná	Normální provoz	ICMP flood útok
Průměr	ipInDelivers	7.36	11760.52
	ipInReceives	7.67	11760.59
	udpNoPorts	0.45	0.21
	icmpInMsgs	0.15	11749.38
	icmpInEchos	0.05	11749.29
Směrodatná odchylka	ipInDelivers	11.14	348.39
	ipInReceives	11.32	348.40
	udpNoPorts	0.28	0.28
	icmpInMsgs	0.02	344.71
	icmpInEchos	0.00	344.71

Numerická analýza

Tabulka 6.2 obsahuje průměrné hodnoty a směrodatnou odchylku pěti nejvýznamnějších SNMP proměnných pro detekci útoku ICMP flood. Lze si povšimnout hodnot proměnné `udpNoPorts`, která ačkoliv na základě procesu *Feature selection* byla označena jako jedna z pěti nejužitečnějších proměnných při klasifikaci útoku ICMP flood, tak její průměrná hodnota se liší pouze minimálně od hodnoty za běžného provozu, dokonce nabývá nižší hodnoty. Lze tedy říci, že tato proměnná má na klasifikaci minimální, či dokonce žádný vliv. Naopak u ostatních proměnných byl zaznamenán vysoký nárůst hodnot. Jak je již typické pro útoky typu flood, pozorujeme nárůst hodnot `ipInDelivers` a `ipInReceives`. Nicméně proměnné, které jsou typické pro útok ICMP flood, jak je z tabulky dále patrné, jsou `icmpInMsgs` a `icmpInEchos`.

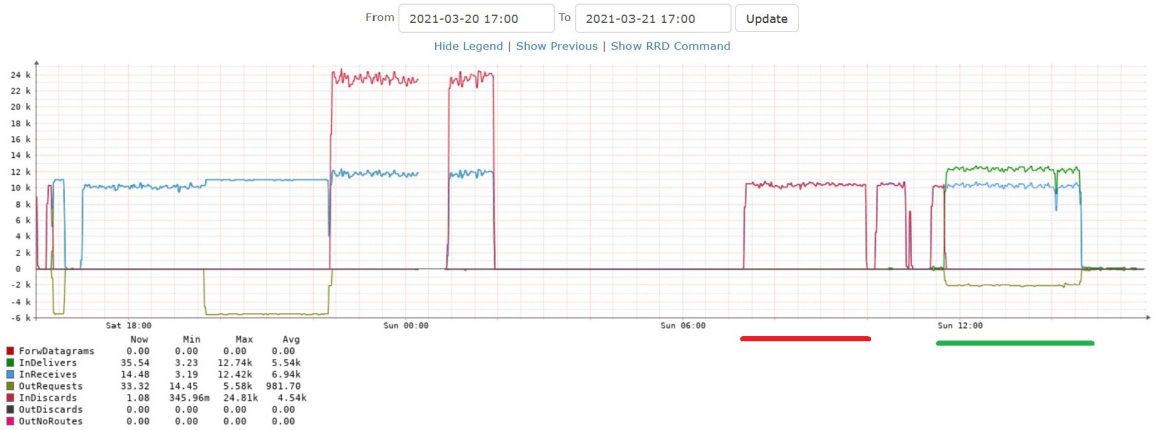
6.3 Analýza útoku UDP flood

Grafická analýza

Po provedení útoku UDP flood došlo k viditelné změně hodnot u třech skupin. Jedná se o skupiny UDP, IP a TCP. V obrázku 6.7 je opět vyobrazen graf z nástroje LibreNMS, který zobrazuje hodnoty proměnných z UDP skupiny. Červená spodní linie značí trvání útoku z první iterace, který byl veden na neotevřený port 80. Tato skutečnost je reflektována vysokou hodnotou proměnné `NoPorts`. Jiná situace nastává při útoku vedeném ve druhé iteraci, který je vyznačen zelenou linií, v tomto případě byl útok veden na port 8000, na kterém naslouchal jednoduchý UDP server. Tato změna se projevuje tak, že namísto proměnné `NoPorts` došlo k růstu hodnot proměnné `InDatagrams`.

Obrázek 6.7: Graf zobrazující vybrané proměnné skupiny UDP během útoku UDP flood

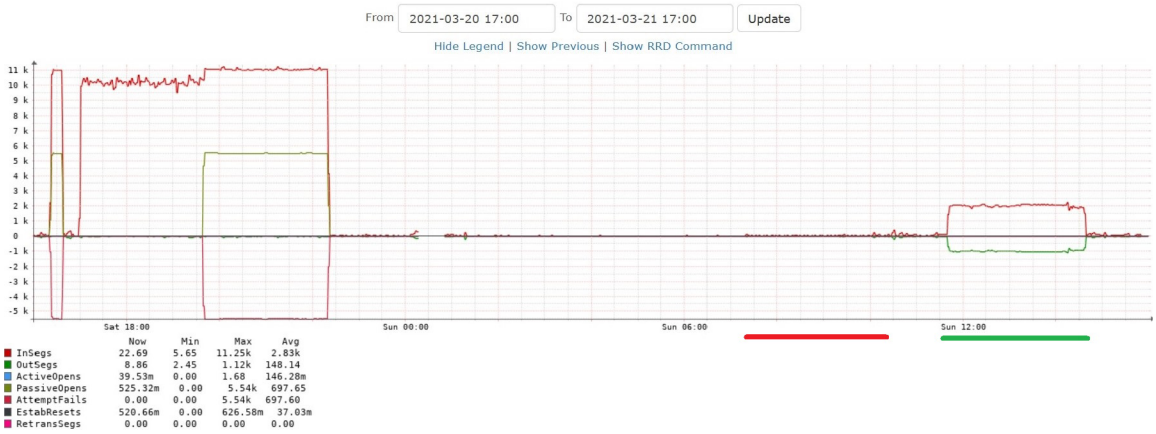
Zajímavější změny je možné pozorovat na grafu 6.8 reprezentujícím proměnné skupiny IP. Během útoku při první iteraci, kdy byl cílový port 80 (uzavřený), docházelo zejména k nárůstu hodnoty proměnné `InDiscards`. Ve druhé iteraci, při útoku na naslouchající UDP port 8000, se staly dominantními proměnnými `InDelivers` a `InReceives`. Dále lze pozorovat lehký nárůst hodnoty proměnné `OutRequests`.



Obrázek 6.8: Graf zobrazující vybrané proměnné skupiny IP během útoku UDP flood

Posledním grafem, ve kterém nastala nějaká viditelná změna hodnot, je skupina TCP v obr. 6.9. Zatímco hodnoty všech proměnných během útoku z první iterace se pohybují

v rozmezí nízkého síťového provozu, tak při druhé iteraci útoku již dochází k nárůstu určitých hodnot. Konkrétně jde o hodnoty `InSegs` a `OutSegs`. V případě proměnné `OutSegs` lze vidět, že její hodnoty korespondují s hodnotami proměnné `OutRequests` z grafu v obr. 6.8.

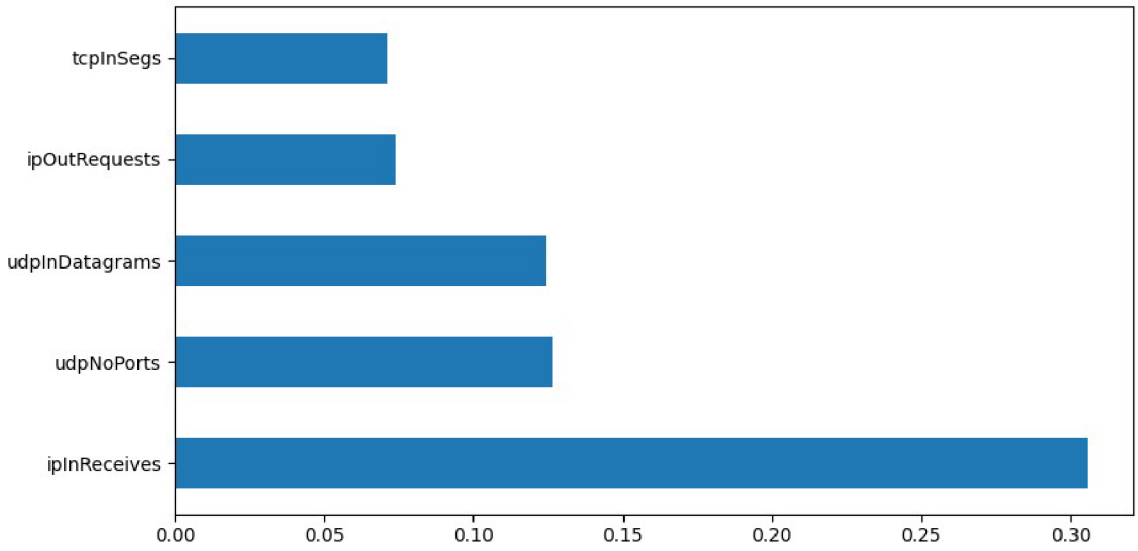


Obrázek 6.9: Graf zobrazující vybrané proměnné skupiny TCP během útoku UDP flood

Výběr rysů

Data zpracovávaná v této sekci jsou nasbíraná za normálního provozu a za provozu pod vlivem útoku UDP flood. Data jsou uložena v souboru *normal_udp_all_vars.csv*.

Výsledek procesu je vyobrazen v grafu na obr. 6.10. Jak je z obrázku patrné, mezi 5 proměnných s největším vlivem na detekci útoku UDP flood patří: `ipInReceives`, `udpNoPorts`, `udpInDatagrams`, `ipOutRequests`, `tcpInSegs`. Lze říci, že výsledky tohoto procesu odpovídají analýze pomocí grafu.



Obrázek 6.10: Graf zobrazuje 5 nejdůležitějších SNMP proměnných při detekci útoku UDP flood

Tabulka 6.3: Průměr a směrodatná odchylka 5 nejužitečnějších proměnných při útoku UDP flood

Operace	Proměnná	Normální provoz	UDP flood útok
Průměr	ipInReceives	7.67	10323.50
	ipOutRequests	17.91	1094.93
	tcpInSegs	26.68	1097.24
	udpInDatagrams	14.20	5454.83
	udpNoPorts	0.45	4876.28
Směrodatná odchylka	ipInReceives	11.32	368.27
	ipOutRequests	10.23	1017.20
	tcpInSegs	58.12	1003.47
	udpInDatagrams	1.05	5130.10
	udpNoPorts	0.28	5199.72

Numerická analýza

Tabulka 6.3 obsahuje průměrné hodnoty a směrodatnou odchylku pěti nejvýznamnějších SNMP proměnných pro detekci útoku UDP flood. Jak je z tabulky patrné, všech 5 proměnných během útoku UDP flood nabývá v porovnání s normálním provozem opravdu vysokých hodnot. Hodnoty proměnných ze skupiny IP jsou však velice podobné hodnotám těchto proměnných při ostatních záplavových útocích, útok UDP flood je tedy nejvíce charakterizován nárůstem hodnot proměnných `udpInDatagrams` a `udpNoPorts`.

6.4 Analýza útoku SSH Bruteforce

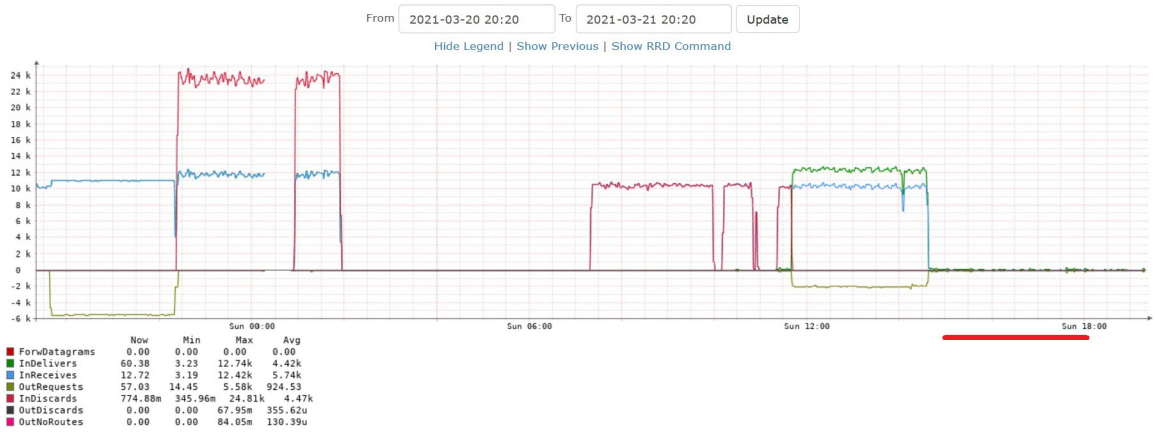
Grafická analýza

Tento typ útoku se od již dříve zmíněných útoků principiálně liší. Zatímco předchozí útoky typu DoS lze považovat za útoky záplavového typu, které se vyznačují anomálním zvýšením provozu a nárůstem hodnot ve skupinách IP, TCP, UDP nebo ICMP, tak v případě útoku Bruteforce na službu SSH se jedná o útok na aplikační vrstvě. Tento útok se na grafech, které poskytuje systém LibreNMS projevuje pouze v minimální míře, mj. pouze ve skupině IP a TCP, jak je patrné z obr. 6.11 a 6.12. Detekovat tento útok na základě základní grafické vizualizace dat, které poskytuje LibreNMS tedy není téměř možné.

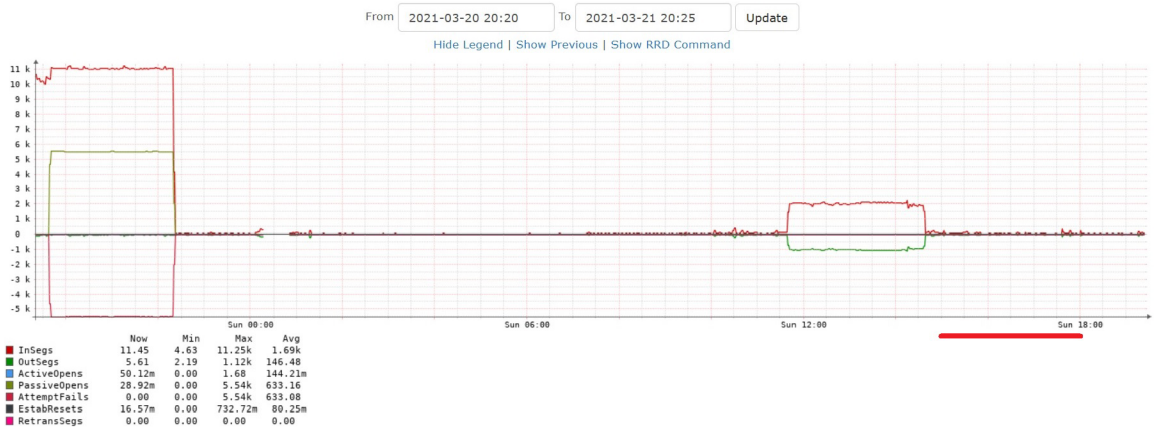
Výběr rysů

V této sekci porovnám a zjistím, které proměnné mají nejvyšší vliv při rozlišení normálního provozu od provozu, který se potýká s probíhajícím SSH Bruteforce útokem. Zpracovávaná data jsou uložena v souboru *normal_bruteforce_all_vars.csv*. Datové sada opět obsahuje kompletní sadu všech SNMP proměnných ze skupin IP, TCP, UDP a ICMP.

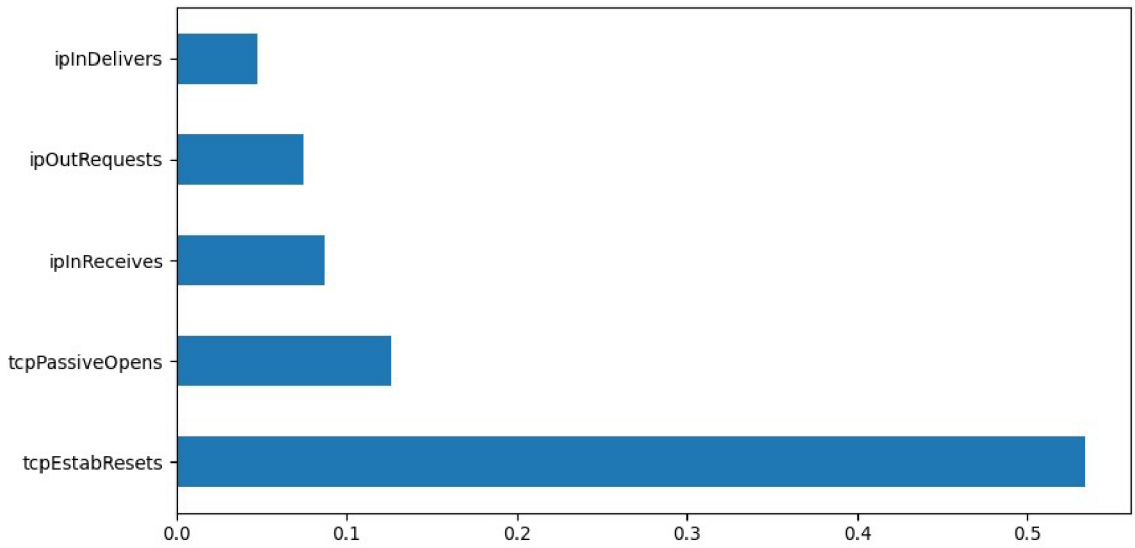
Výsledný graf tohoto procesu je vyobrazen v obr. 6.13. Zatímco v základních grafech nástroje LibreNMS nebylo možné vizuálně rozlišit žádný probíhající útok, jiná situace nastává v tomto případě. Jak je zřejmé z grafu, nejvyšší vliv na rozpoznání útoku SSH Bruteforce má proměnná `tcpEstabResets`, následováno proměnnými `tcpPassiveOpens, ipInReceives, ipOutRequests, ipInDelivers`.



Obrázek 6.11: Graf zobrazující vybrané proměnné skupiny IP během útoku SSH Bruteforce



Obrázek 6.12: Graf zobrazující vybrané proměnné skupiny TCP během útoku SSH Bruteforce



Obrázek 6.13: Graf zobrazuje 5 nejdůležitějších SNMP proměnných při detekci útoku SSH Bruteforce

Tabulka 6.4: Průměr a směrodatná odchylka 5 nejužitečnějších proměnných při útoku SSH Bruteforce

Operace	Proměnná	Normální provoz	SSH Bruteforce útok
Průměr	ipInDelivers	7.36	64.33
	ipInReceives	7.67	18.74
	ipOutRequests	17.91	61.16
	tcpPassiveOpens	0.03	0.52
	tcpEstabResets	0.01	0.52
Směrodatná odchylka	ipInDelivers	11.14	53.30
	ipInReceives	11.32	8.50
	ipOutRequests	10.23	52.03
	tcpPassiveOpens	0.04	0.08
	tcpEstabResets	0.03	0.08

Numerická analýza

Tabulka 6.4 obsahuje průměrné hodnoty a směrodatnou odchylku pěti nejvýznamnějších SNMP proměnných pro detekci útoku SSH Bruteforce. Jak je z tabulky patrné, hodnoty nedosahují takových rozměrů jako v případě předchozích záplavových útoků. Z předchozí sekce jsem zjistil, že nejvyšší vliv při klasifikaci tohoto typu útoku má proměnná `tcpEstabResets`, její hodnota oproti stavu za normálního provozu vzrostla přibližně 44x.

6.5 Analýza útoku Slowloris

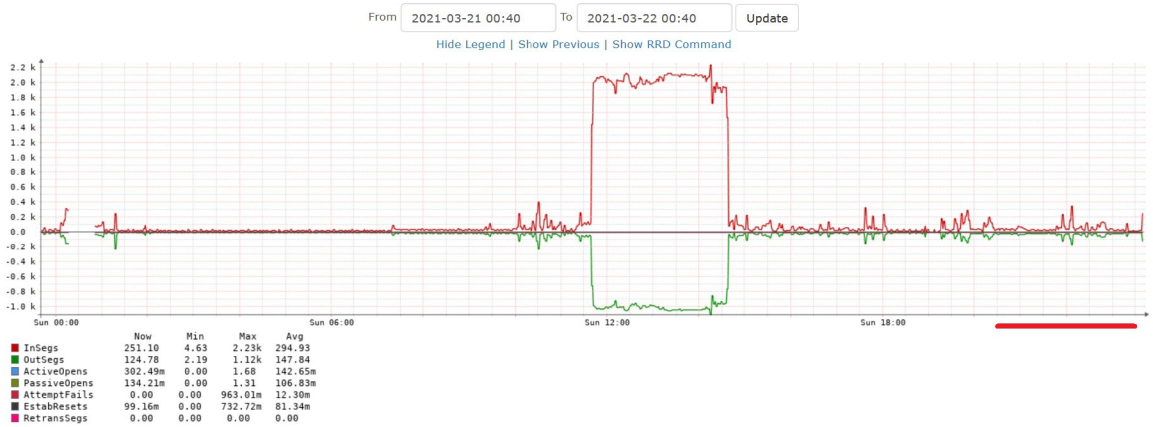
Grafická analýza

Podobně jako v případě útoku SSH Bruteforce, tak i útok Slowloris se řadí mezi útoky na aplikační vrstvě. Jak již bylo vysvětleno v sekci 5.2.1, jedná se o útok, při kterém nevzniká žádný abnormálně vysoký provoz. Jeho síla spočítá v navázání velkého množství spojení, které udržuje otevřené po maximálně možnou dobu. Na obr. 6.14 reprezentujícím graf s hodnotami skupiny TCP lze vidět, že hodnoty odeslaných a přijatých dat odpovídání téměř normálnímu provozu na síti. Situace je tedy podobná jako v případě útoku SSH Bruteforce, kdy dostupné grafické zobrazení dat v nástroji LibreNMS není vhodné k rozpoznání tohoto probíhajícího útoku.

Výběr rysů

V této sekci porovnám vliv proměnných SNMP na datech sesbíraných za normálního provozu s daty, které jsem sesbíral při provádění útoku Slowloris. Zpracovávaná data jsou uložena v souboru *normal_slowloris_all_vars.csv*. Po provedení procesu *Výběr rysů* a získání hodnot z vlastnosti modelu `feture_importances_` jsem získal hodnoty uvedené v grafu na obr. 6.15. 5 hodnot, které mají nejvyšší váhu při klasifikaci tohoto typu útoku jsou následující: `ipInDelivers, ipOutRequests, udpNoPorts, ipInReceives, tcpOutSegs`.

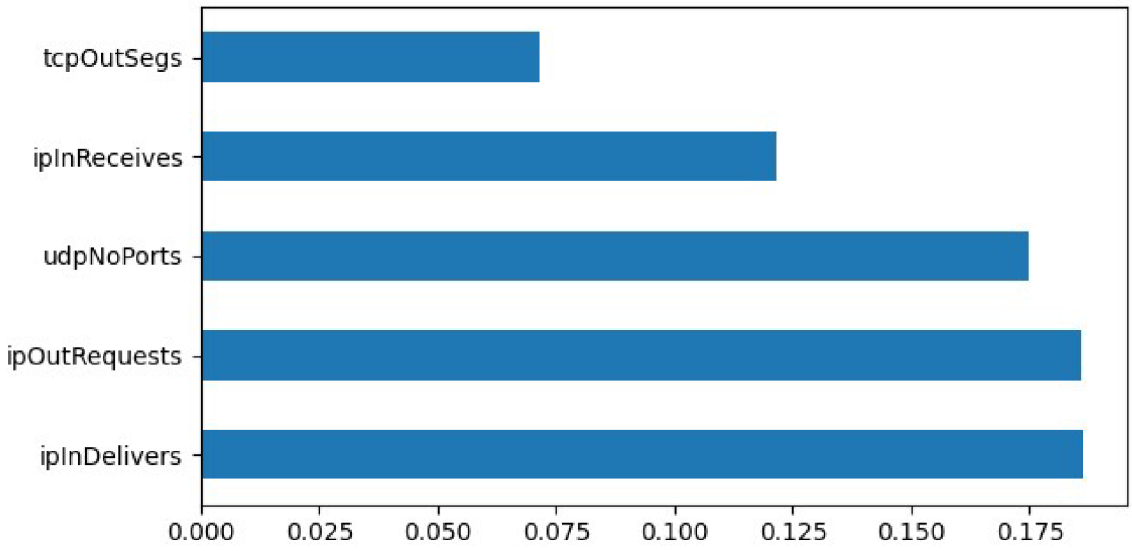
V tomto případě se hodnoty od ostatních útoků podstatě liší. Zatímco ve většina předchozích případů existovala jedna nebo dvě proměnné, které byly typické pro charakteristiky daného útoku, tak v tomto případě mají nejlépe hodnocené proměnné spíše rovnoměrné rozložení a jsou více obecného charakteru. Ze své podstaty se jedná o proměnné, jejichž



Obrázek 6.14: Graf zobrazující vybrané proměnné skupiny TCP během útoku Slowloris

hodnota může být v tomto rozložení zvýšena obyčejnou aktivitou uživatele, tj. procházení webových stránek, sledování videa na internetu atd.

Vzhledem k tomu, že na základě velikosti hodnot a při pohledu na graf z části grafické analýzy tohoto útoku, se jednalo o podobné hodnoty jako při normálním provozu, lze odhadnout, že tento typ útoku může být často falešně klasifikován jako normální provoz či naopak, normální provoz může být klasifikován jako útok Slowloris.



Obrázek 6.15: Graf zobrazuje 5 nejdůležitějších SNMP proměnných při detekci útoku Slowloris

Numerická analýza

Tabulka 6.5 obsahuje průměrné hodnoty a směrodatnou odchylku pěti nejvýznamnějších SNMP proměnných pro detekci útoku Slowloris.

Tabulka 6.5: Průměr a směrodatná odchylka 5 nejužitečnějších proměnných při útoku Slowloris

Operace	Proměnná	Normální provoz	Slowloris
Průměr	ipInDelivers	7.36	66.09
	ipInReceives	7.67	20.27
	ipOutRequests	17.90	63.69
	tcpOutSegs	13.54	31.25
	udpNoPorts	0.45	0.28
Směrodatná odchylka	ipInDelivers	11.14	62.89
	ipInReceives	11.32	10.16
	ipOutRequests	10.23	61.51
	tcpOutSegs	30.26	30.57
	udpNoPorts	0.28	0.53

6.6 Testování

V této kapitole se budu zabývat testováním mých modulů na útocích v reálném čase. K testování, podobně jako při trénování útoků, využiji můj notebook s operačním systémem Windows 10, který bude sloužit jako cílové zařízení při testování útoků, které povedu ze zařízení Raspberry Pi s operačním systémem Raspbian 10. K testování provedu typově stejné útoky jako v kapitole 5.2.1, ovšem s různými variacemi a změnami v parametrech. Jedná se tedy o testování formou provádění experimentů.

Při tomto testování budu používat model v modulu `ATTACK_CLASSIFICATION`, který byl natrénován na datové sadě, která neobsahuje data nasbíraná při útoku Slowloris. Jak jsem uvedl v kapitole 6.5, je zde velká podoba s charakteristikami normálního provozu. V moment, kdy byla tato data zahrnuta do trénovacího datasetu, v reálném provozu docházelo téměř k padesátiprocentnímu výskytu false-positive alarmů a nebylo možné testovat ostatní typy útoků.

Obrázek 6.16 zachycuje stav modulů za normálního provozu, kdy neprobíhá žádný síťový útok.

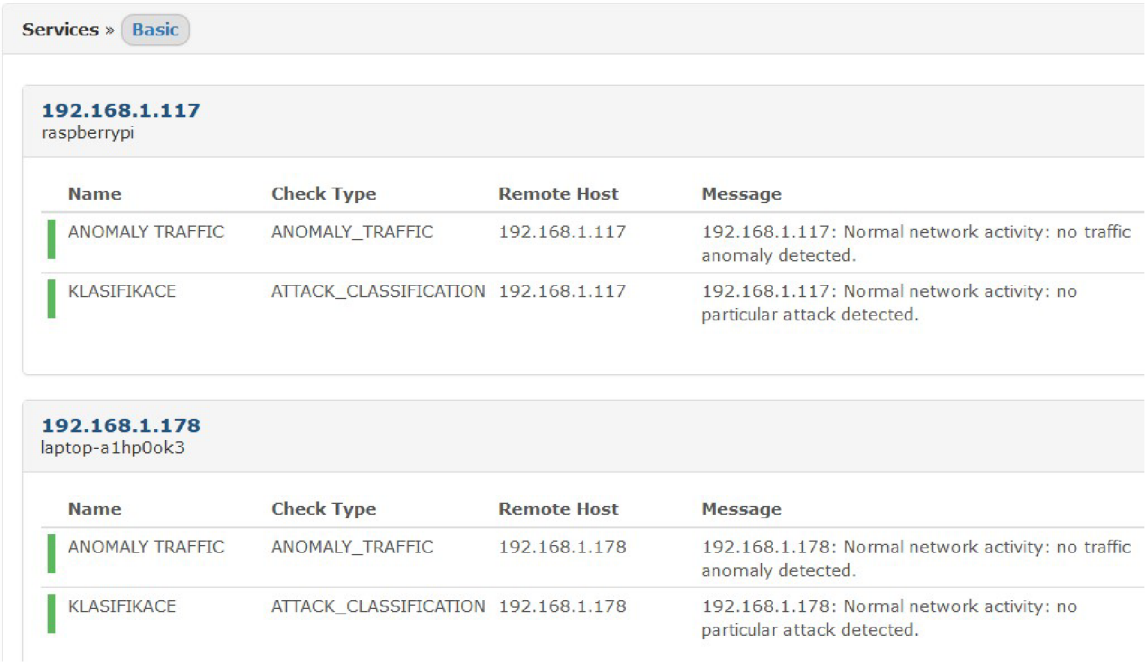
6.6.1 Test detekce útoku SSH Bruteforce

Prvním útokem, který budu testovat je útok SSH Bruteforce. Na mém notebooku běží OpenSSH server, který je dostupný ve Windows 10 jako "optional-feature"[1]. Útok provedu spuštěním následujícího příkazu na mém zařízení Raspberry Pi:

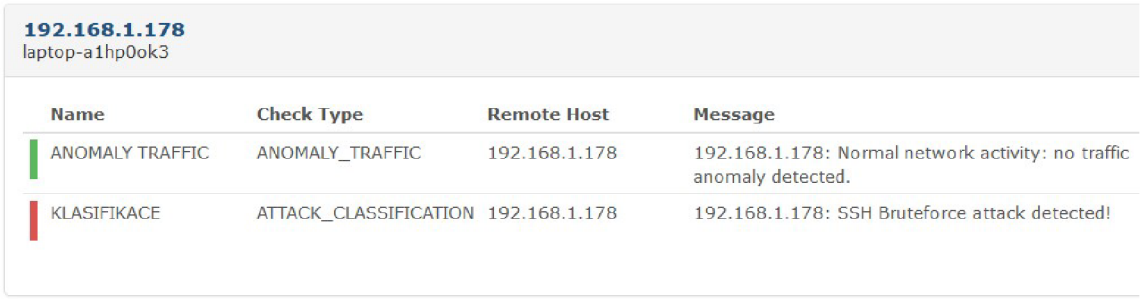
1. `sudo hydra -l "daniel stepan"-x 4:4:aA1 192.168.1.178 ssh`

Reakce modulů je uvedena v obr. 6.17. Jak je z obrázku patrné, modul provádějící klasifikaci útoků úspěšně detekoval útok SSH Bruteforce. Zatímco druhý modul detekce anomálního provozu nehlásí žádnou anomálii. Toto chování je na základě analýzy provedené v podkapitole 6.4 očekávané, poněvadž charakteristika tohoto provozu je rovněž velice podobná charakteristice normálního provozu.

[1] https://www.pugetsystems.com/labs/hpc/How-To-Use-SSH-Client-and-Server-on-Windows-10-1470



Obrázek 6.16: Stav obou modulů pro sledovaná zařízení za normálního síťového provozu



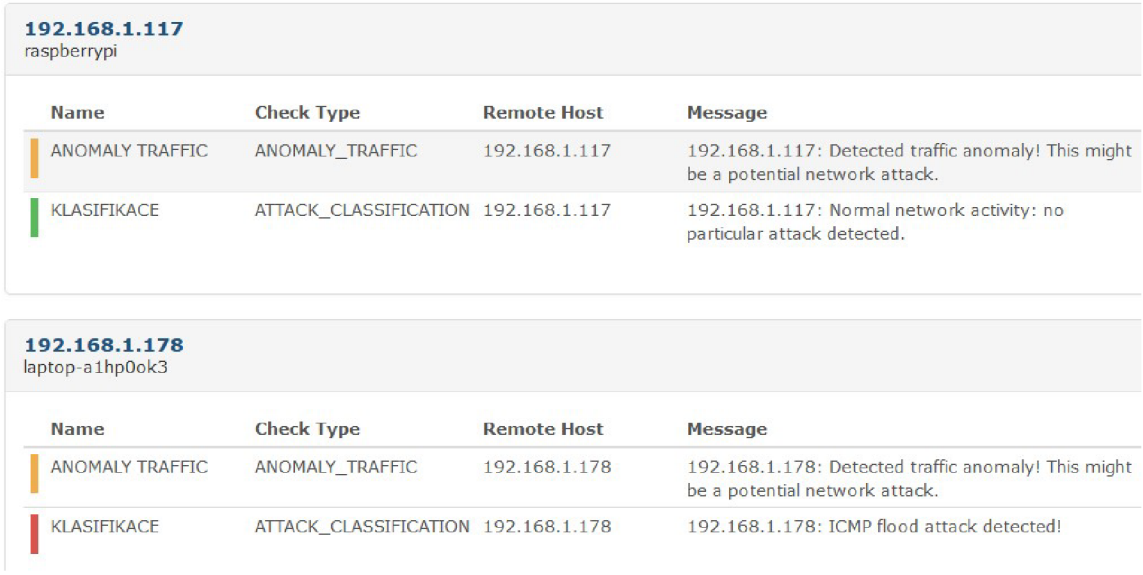
Obrázek 6.17: Stav obou modulů při útoku SSH Bruteforce

6.6.2 Test detekce útoku ICMP flood

Druhým testovaným útokem je útok ICMP flood. Útok je opět veden ze zařízení Raspberry Pi na notebook se systémem Windows 10. Za účelem testování jsem simuloval následující varianty tohoto útoku:

1. `sudo hping3 --icmp --flood 192.168.1.178`
2. `sudo hping3 --icmp -i u100 --data 1300 192.168.1.178`

Výsledek tohoto testu je na obr. 6.18. V tomto případě zareagovaly oba moduly při zařízení na které byl veden tento útok. Lze si povšimnout, že modul `ANOMALY_TRAFFIC` reagoval i u zařízení, které provádělo tento útok, vzhledem k velkému nárůstu odchozího provozu. Reakce modulů byla stejná u obou variant provedeného útoku. Lze tedy říci, že moduly v tomto případě reagují i na nižší intenzitu útoku ICMP flood obsahujícího data.



Obrázek 6.18: Stav obou modulů při útoku ICMP flood

6.6.3 Test detekce útoku TCP-SYN flood

Třetím testovaným útokem je útok TCP-SYN flood, který byl veden opět ze zařízení Raspberry Pi na notebook s operačním systémem Windows 10. Tento typ útoku byl simulován v následujících variantách:

1. `sudo hping3 --flood -S -p 80 192.168.1.178`
2. `sudo hping3 --flood -S -p 8000 192.168.1.178`

V prvním případě na mé stanici nebyla žádný naslouchající aplikace na portu 80. Ve druhém případě jsem měl spuštěn jednoduchý TCP server.

Stav modulů během simulace prvního útoku je zobrazen v obr. 6.19. Dle očekávání byl útok klasifikován jako útok TCP-SYN flood. Rovněž vzhledem k tomu, že se jedná o útok záplavového typu, dochází i ke změně stavu u modulu detekujícího anomálnosti v síťovém provozu a to jak u příjemce tak i u původce útoku.

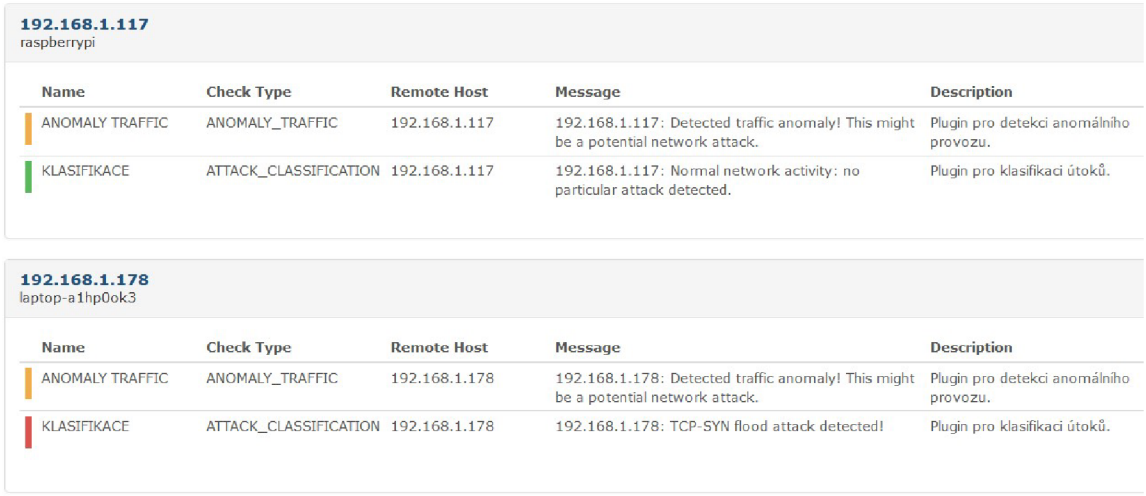
6.6.4 Test detekce útoku UDP flood

Posledním testovaným útokem je útok UDP flood. Stejně jako v předchozích případech, vedený ze zařízení Raspberry Pi na počítač s operačním systémem Windows 10. Útok byl při testování spuštěn v této konfiguraci:

1. `sudo hping3 --udp --flood -p 80 192.168.1.178`
2. `sudo hping3 --udp --flood -p 8000 192.168.1.178`

V prvním případě opět nebyla na stanici žádná naslouchající aplikace na portu 80. Ve druhém případě byl spuštěn jednoduchý UDP server naslouchající na portu 8000.

Stav modulů během provádění prvního útoku je uveden v obr. 6.20. Dle očekávání byl příchozí útok klasifikován jako útok typu UDP flood. U obou stanic došlo také k aktivaci druhého modulu, který upozornil v obou případech na anomální úroveň provozu, kdy



192.168.1.117
raspberrypi

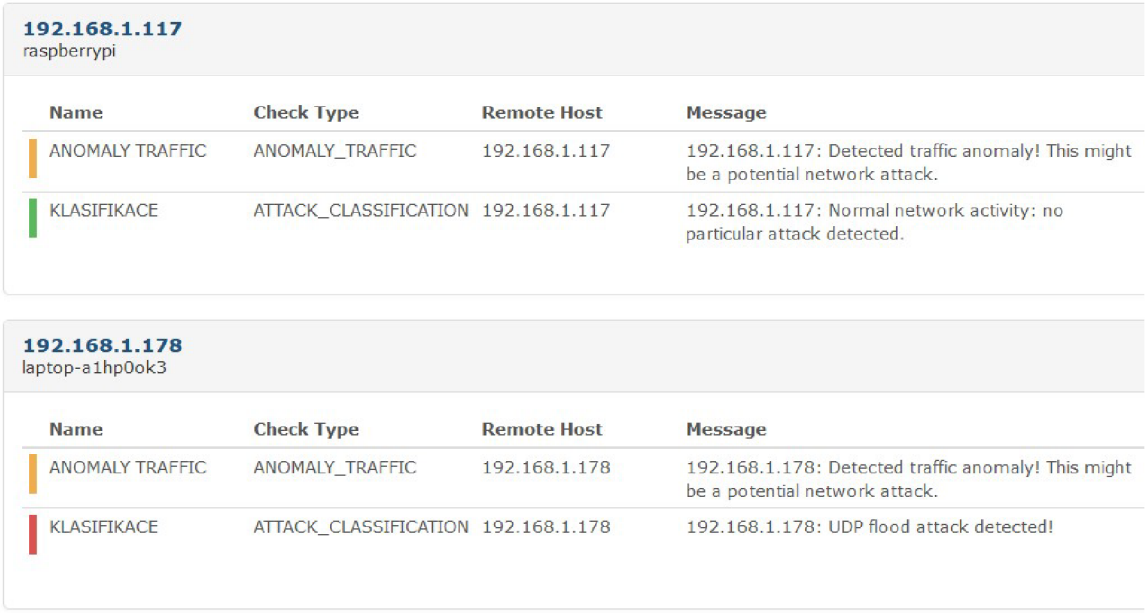
Name	Check Type	Remote Host	Message	Description
ANOMALY TRAFFIC	ANOMALY_TRAFFIC	192.168.1.117	192.168.1.117: Detected traffic anomaly! This might be a potential network attack.	Plugin pro detekci anomálního provozu.
KLASIFIKACE	ATTACK_CLASSIFICATION	192.168.1.117	192.168.1.117: Normal network activity: no particular attack detected.	Plugin pro klasifikaci útoků.

192.168.1.178
laptop-a1hp0ok3

Name	Check Type	Remote Host	Message	Description
ANOMALY TRAFFIC	ANOMALY_TRAFFIC	192.168.1.178	192.168.1.178: Detected traffic anomaly! This might be a potential network attack.	Plugin pro detekci anomálního provozu.
KLASIFIKACE	ATTACK_CLASSIFICATION	192.168.1.178	192.168.1.178: TCP-SYN flood attack detected!	Plugin pro klasifikaci útoků.

Obrázek 6.19: Stav obou modulů při útoku TCP-SYN flood

v případě příjemce útoku šlo o anomální, respektive nadměrný, příjem paketů a v případě útočícího zařízení se výjimečně vysoký počet odchozích paketů.



192.168.1.117
raspberrypi

Name	Check Type	Remote Host	Message
ANOMALY TRAFFIC	ANOMALY_TRAFFIC	192.168.1.117	192.168.1.117: Detected traffic anomaly! This might be a potential network attack.
KLASIFIKACE	ATTACK_CLASSIFICATION	192.168.1.117	192.168.1.117: Normal network activity: no particular attack detected.

192.168.1.178
laptop-a1hp0ok3

Name	Check Type	Remote Host	Message
ANOMALY TRAFFIC	ANOMALY_TRAFFIC	192.168.1.178	192.168.1.178: Detected traffic anomaly! This might be a potential network attack.
KLASIFIKACE	ATTACK_CLASSIFICATION	192.168.1.178	192.168.1.178: UDP flood attack detected!

Obrázek 6.20: Stav obou modulů při útoku UDP flood

6.6.5 Shrnutí

V této kapitole jsem provedl analýzu všech nasimulovaných útoků TCP-SYN flood, ICMP flood, UDP flood, SSH Bruteforce a Slowloris. Při každém z těchto typů útoků jsem rozdělil analýzu na tři části: Grafická analýza, Výběr rysů, Numerická analýza. V části *Grafická analýza* komentuji grafy přístupné v nástroji LibreNMS. V části *Výběr rysů* provádím trénink modelu klasifikátoru na datasetu, který obsahuje data sesbíraná za normálního provozu

a při simulaci daného útoku. Během tréninku provede model ohodnocení každé proměnné (feature), které uloží do své proměnné `feature_importances_`. Z této proměnné zobrazím a popíši graf 5 proměnných s nejvyšší hodnotou. Jedná se o proměnné, které jsou nejvíce relevantní při procesu klasifikace u tohoto algoritmu *Random forest*. Zároveň tak získám informace o tom, které SNMP proměnné jsou nejužitečnější při rozpoznávání jednotlivých síťových útoků. Ve třetí části *Numerická analýza* vždy provádím porovnání 5 nejrelevantnějších hodnot z předchozí části *Výběr rysů*.

V poslední části této kapitoly provádím testování obou mnou vytvořených modulů na datech v reálném čase, kdy simuluji natrénované typy útoků.

Konkrétní výsledky analýz z této kapitoly popisuji v poslední závěrečné kapitole, kde rovněž provádím shrnutí celé práce, diskutuji získané výsledky a uvažuji možnosti dalšího rozšíření.

Kapitola 7

Závěr

Cílem této práce bylo seznámit se s principy analýzy anomálií v prostředí počítačových sítí s využitím systému, který umožňuje analýzu aktivních dotazů prostřednictvím protokolu SNMP a následně navrhnout systém, který bude bezpečnostní anomálie detekovat. Po dohodě s vedoucím jsem práci zaměřil nejen na detekci anomálního provozu s využitím strojového učení, ale i na klasifikaci vybraných síťových útoků, především útoků DoS.

V první části této práce, která je teoretickým úvodem do analýzy anomálií a systému SNMP, jsem popsal typy anomálií i metody využívané pro jejich detekci, následováno popisem systému SNMP a souvisejících pojmů.

Systém detekce anomálií a klasifikace síťových útoků jsem se rozhodl vytvořit ve formě dvou modulů jako rozšíření pro existující open-source nástroj LibreNMS využívaný k monitorování síťové infrastruktury prostřednictvím protokolu SNMP. Ve druhé části této práce jsou tedy popsány požadavky na takový systém a je popsán proces konfigurace klíčových prvků u vybraného nástroje, konkrétně aktivace mnou vytvořených modulů a nastavení upozornění (alertů) na konkrétní výstupy těchto modulů.

Třetí část práce se již zaměřuje na konkrétní řešení a implementaci obou modulů. První modul s názvem `ATTACK_CLASSIFICATION`, který provádí klasifikaci známých síťových útoků, je založen na modelu využívající algoritmus *Random Forest Classifier*. Data, na kterých jsem trénoval tento model, jsem sám generoval simulováním řady síťových útoků. Druhý modul nese název `ANOMALY_TRAFFIC` a využívá algoritmus *Local Outlier Factor*. Oba algoritmy jsou dostupné v open-source knihovně *scikit-learn*.

Poslední, čtvrtá část se věnuje analýze dat získaných během simulace útoků z předchozí části. Analyzuji data získaná ze všech provedených útoků, tj. útoky TCP-SYN flood, ICMP flood, UDP flood, SSH Bruteforce, Slowloris.

Z analýzy útoku TCP-SYN flood v sekci 6.1 vyplývá, že nejvyšší vliv při detekci tohoto útoku má ze SNMP TCP skupiny hodnota `tcpInSegs` s OID 1.3.6.1.2.1.6.10, která vyjadřuje počet přijatých TCP paketů. Tato hodnota `tcpInSegs` během probíhajícího útoku dosahuje téměř 398krát větší hodnoty oproti průměrné hodnotě za normálního provozu. Tato skutečnost je vzhledem k povaze tohoto útoku očekávaná a moje moduly klasifikace i detekce anomálií na tento útok velice dobře reagují.

Z analýzy druhého útoku ICMP flood v sekci 6.2 vyplývá, že při tomto útoku je ze skupiny ICMP nejvíce ovlivněná hodnota `icmpInEchos` s OID 1.3.6.1.2.1.5.8, která reprezentuje počet přijatých paketů ICMP Echo, jenž se typicky nazývají ping. Oproti průměrné hodnotě za běžného provozu je tato hodnota pod vlivem útoku ICMP flood vyšší téměř 234 985krát a to z toho důvodu, že pakety typu ICMP Echo se až na určité případy diagnostiky sítě v běžném provozu téměř nevyskytují.

Z analýzy útoku UDP flood v sekci 6.3 lze vypozorovat, že při útoku na port, na kterém nebyla žádná naslouchající aplikace, došlo k velkému nárůstu hodnoty `udpNoPorts` s OID 1.3.6.1.2.1.7.2, která v během útoku dosahovala v průměru 10 835násobku průměrné hodnoty za normálního provozu. Ve druhém případě, kdy útok probíhal na port, na kterém naslouchal jednoduchý UDP server, došlo k nejvyššímu nárůstu hodnoty `udpInDatagrams` s OID 1.3.6.1.2.1.7.1, která vyjadřuje počet přijatých UDP paketů. Oproti hodnotám za normálního provozu došlo k 384násobnému nárůstu. Všechny dosud jmenované a analyzované útoky jsou útoky tzv. záplavového typu, nebo-li útoky typu DoS. Každý z těchto útoků, ať již je to TCP, UDP nebo ICMP flood, velmi ovlivňuje SNMP proměnné z MIB databáze odpovídajících skupin, tj. pro útok typu TCP flood jsou ovlivněny převážně proměnné této kategorie, analogicky je to s ostatními typy DoS útoků. U všech těchto útoků záplavového typu lze dále pozorovat jednu společnou vlastnost, a to nárůst hodnot u proměnných skupiny IP, zejména `ipInReceives`. Na základě těchto znalostí lze tedy nejen detekovat to, že dochází k útoku DoS, ale vzhledem k tomu, že dochází k nárůstu hodnot i u specifických skupin proměnných databáze MIB, lze provést klasifikaci a rozpoznání konkrétního typu útoku.

Předposledním analyzovaným útokem v sekci 6.4 je útok SSH Bruteforce. V porovnání s předchozími útoky zde nedochází k výrazné změně intenzity provozu, nicméně dochází k mnohačetným navázáním TCP spojení a pokusům o přihlášení, což se nejvíce projeví na nárůstu hodnoty `tcpEstabResets` s OID 1.3.6.1.2.1.6.8. To umožňuje bezproblémovou klasifikaci tohoto typu útoku mým modulem pro klasifikaci známých útoků. Na druhou stranu, poněvadž nedochází k výraznému zvýšení síťového provozu, tak druhý modul pro detekci anomálního provozu tento útok nedetekuje.

Posledním analyzovaným útokem v sekci 6.5 je útok Slowloris. Na základě jeho analýzy jsem došel ke zjištění, že síťový provoz se liší minimálně od normálního provozu. Zároveň zde není žádná hodnota, na základě které by bylo možné tento útok spolehlivě klasifikovat. V případě trénování mých modelů s daty zahrnujícími tento útok docházelo k velkému výskytu případů false-positive, kde normální provoz byl mylně klasifikován jako útok Slowloris. Proto jsem další testování prováděl bez dat tohoto útoku.

Poslední částí této práce je, jak již bylo naznačeno, testování mých dvou modulů na řadě útoků. S výsledky testování jsem naprosto spokojen. V případě záplavových útoků DoS a útoku SSH Bruteforce došlo k bezproblémové klasifikaci těchto útoků mým modulem pro klasifikaci. Zároveň při záplavových útocích byl mým druhým modelem detekován anomální nárůst síťového provozu, pouze při útoku SSH Bruteforce můj modul detekce anomálního provozu nedetekoval žádnou podezřelou aktivitu, protože se jedná o typ útoku na aplikační vrstvě, avšak síťový provoz se podobá síťovému provozu normálního provozu. Toto chování je tedy očekávané.

Na základě výsledků získaných z analýzy provedených útoků a z testování lze dojít k závěru, že s využitím dat získaných prostřednictvím protokolu SNMP lze úspěšně detekovat a klasifikovat řadu síťových útoků, zejména útoky typu DoS. Vzhledem k tomu, že můj druhý vytvořený modul detekuje anomální hodnoty síťového provozu, není zde omezení pouze na typy útoků DoS uvedených v této práci. Je zde předpoklad, že může úspěšně detekovat další typy útoků, které pracují na principu zahlcení cílové stanice, například amplifikační útoky zneužívající veřejné DNS či NTP servery a zesilující útok na cílovou stanici. Na druhou stranu, jak bylo vypozorováno z analýzy útoku Slowloris, detekování útoků aplikační vrstvy s daty získatelnými prostřednictvím protokolu SNMP, se nejeví jako efektivní a to z toho důvodu, že z pohledu síťového provozu nedochází k žádnému anomálnímu chování.

Všechny body zadání této práce byly úspěšně splněny. Co se týče dalšího možného vývoje, bylo by například možné zaměřit se na hodnoty vnitřních zdrojů kontrolovaných stanic a jejich korelací detekovat neobvyklé vzory chování, např. vysoký výkon procesoru a vytížení pamětí v dlouhodobém či krátkodobém časovém horizontu, případně s ohledem na uptime systému. Tímto způsobem by bylo možné např. detekovat malware, který využívá výpočetní výkon počítače ku vlastnímu prospěchu.

Literatura

[1] Ahmed, M., Mahmood, A. N. a Hu, J. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*. 1. vyd. 2016, sv. 60, č. 01, s. 19–31. ISSN 1084-8045.

[2] Alla, S. a Adari, S. K. *Beginning Anomaly Detection Using Python-Based Deep Learning*. 1. vyd. Apress, 2019. ISBN 978-1-4842-5177-5.

[3] Berry, M. W., Mohamed, A. a Yap, B. W. *Supervised and Unsupervised Learning for Data Science*. 1. vyd. Springer, 2020. ISBN 978-3-030-22475-2.

[4] Bhuyan, M. H., Bhattacharyya, D. K. a Kalita, J. K. *Network Traffic Anomaly Detection and Prevention: Concepts, Techniques, and Tools*. 1. vyd. Spinger, 2017. ISBN 978-3-319-65186-6.

[5] Case, J., Fedor, M., Schoffstall, M. a Davin, J. *A Simple Network Management Protocol (SNMP)* [online]. Květen 1990 [cit. 2021-01-03]. Dostupné z: https://tools.ietf.org/html/rfc1157.

[6] Case, R., Mundy, R., Partain, D. a Stewart, B. *Introduction and Applicability Statements for Internet Standard Management Framework* [online]. Prosinec 2002 [cit. 2021-04-25]. Dostupné z: https://tools.ietf.org/html/rfc3410.

[7] Chandola, V., Banerjee, A. a Kumar, V. Anomaly Detection: A survey. *ACM Computing Surveys*. 1. vyd. 2009, sv. 41, č. 3. ISSN 0360-0300.

[8] Cloudflare. *Ping (ICMP) Flood DDoS Attack* [online]. 2021 [cit. 2021-02-23]. Dostupné z: https://www.cloudflare.com/learning/ddos/ping-icmp-flood-ddos-attack/.

[9] Donges, N. *A complete guide to the random forest algorithm* [online]. Červen 2019 [cit. 2021-05-01]. Dostupné z: https://builtin.com/data-science/random-forest-algorithm.

[10] *Global OID reference database* [online]. 2021 [cit. 2021-03-29]. Dostupné z: https://oidref.com/1.3.6.1.2.1.

[11] Kozierok, C. M. *TCP/IP Internet Standard Management Framework Architecture and Protocol Components* [online]. Září 2005 [cit. 2021-01-02]. Dostupné z: http://www.tcpipguide.com/free/t_TCPIPInternetStandardManagementFrameworkArchitectu.htm.

[12] Krčmář, P. *Útok Slowloris aneb plíživé nebezpečí pro web servery* [online]. Květen 2011 [cit. 2021-03-20]. Dostupné z: https://www.root.cz/clanky/utok-slowloris-aneb-plizive-nebezpeci-pro-web-servery/.

[13] *LocalOutlierFactor* [online]. 2021 [cit. 2021-04-05]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html#sklearn.neighbors.LocalOutlierFactor.

[14] Matoušek, P. *Síťové aplikace a jejich architektura*. 1. vyd. VUTIUM, 2014. ISBN 978-80-214-3766-1.

[15] Mauro, D. a Schmidt, K. *Essential SNMP*. 2. vyd. O'Reilly, 2005. ISBN 2-596-00840-6.

[16] McCloghrie, K. a Rose, M. *Management Information Base for Network Management of TCP/IP-based internets: MIB-II* [online]. Březen 1991 [cit. 2021-01-04]. Dostupné z: https://tools.ietf.org/html/rfc1213.

[17] Mueller, J. P. a Massaron, L. *Machine Learning For Dummies*. 1. vyd. John Wiley & Sons, 2016. ISBN 978-1-119-24551-3.

[18] *Novelty Detection* [online]. 2021 [cit. 2021-04-05]. Dostupné z: https://deepai.org/machine-learning-glossary-and-terms/novelty-detection.

[19] NÚKIB. *Zpráva o stavu kybernetické bezpečnosti České republiky za rok 2019* [online]. NÚKIB, listopad 2019 [cit. 2020-12-11]. Dostupné z: https://www.nukib.cz/download/publikace/zpravy_o_stavu/NUKIB_ZSKB_2019.pdf.

[20] Oetiker, T. *About RRDtool* [online]. Únor 2017 [cit. 2021-01-18]. Dostupné z: https://oss.oetiker.ch/rrdtool/doc/rrdcreate.en.html.

[21] *Outlier detection with Local Outlier Factor (LOF)* [online]. 2021 [cit. 2021-04-05]. Dostupné z: https://scikit-learn.org/stable/auto_examples/neighbors/plot_lof_novelty_detection.html.

[22] Parisi, A. *Hands-On Artificial Intelligence for Cybersecurity*. 1. vyd. Packt Publishing, 2019. ISBN 9781789804027.

[23] Presuhn, R., Case, J., McCloghrie, K., Rose, M. a Waldbusser, S. *Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)* [online]. Prosinec 2002 [cit. 2021-01-03]. Dostupné z: https://tools.ietf.org/html/rfc3416.

[24] Rose, M. a McCloghrie, K. *Structure and Identification of Management Information for TCP/IP-based Internets* [online]. Květen 1990 [cit. 2021-01-02]. Dostupné z: https://tools.ietf.org/html/rfc1155.

[25] Sharma, A. *Decision Tree vs. Random Forest – Which Algorithm Should you Use?* [online]. Květen 2020 [cit. 2021-02-25]. Dostupné z: https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/.

[26] Smith, L. I. *A tutorial on Principal Components Analysis* [online]. Únor 2002 [cit. 2021-01-09]. Dostupné z: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf.

[27] Stallings, W. *Network security essentials: applications and standards.* 4. vyd. Prentice Hall, 2010. ISBN 9780136108054.

[28] Stanton, M. *8.3 The Internet Network Management Framework* [online]. Květen 2018 [cit. 2021-01-05]. Dostupné z: http://www2.ic.uff.br/~michael/kr1999/8-netman/8_03-snmp.htm.

[29] *Top SNMP Alternatives because SNMP is dying (updated 2021)* [online]. Leden 2021 [cit. 2021-04-26]. Dostupné z: https://bestmonitoringtools.com/top-snmp-alternatives-because-snmp-is-dying/.

[30] Yiu, T. *Understanding Random Forest* [online]. Červen 2019 [cit. 2021-02-25]. Dostupné z: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

Příloha A

Obsah přiloženého média

Na přiloženém médiu se nachází:

- Kořenový adresář: soubory README.txt, licence, návod k instalaci, text bakalářské práce, zdrojové soubory bakalářské práce v LaTeXu
- Adresář zpracovani_dat: data ze zařízení nasbíraná během simulace síťových útoků, data využitá k výpočtům v sekcích *Numerická analýza*, data využitá během procesů *Výběr rysů*
- Adresář udp_server: jednoduchý UDP server využitý při simulaci útoku typu UDP flood
- Adresář moduly: soubory mnou vytvořených modulů, pomocné soubory využívané těmito moduly, skripty pro trénink modelů využívající strojové učení a potřebná trénovací data

Příloha B

Seznam použitých zkratek

- ASN.1 – Abstract Syntax Notation One
- CSV – Comma-Separated Values
- DNS – Domain Name System
- DoS – Denial of Service
- HTTP – Hypertext Transfer Protocol
- ICMP – Internet Control Message Protocol
- LOF – Local Outlier Factor
- MIB – Management Information Base
- NMS – Network Management System
- NTP – Network Time Protocol
- OID – Object Identifier
- RRDtool – Round-Robin Database Tool
- SSH – Secure Shell Protocol
- SMI – Structure of Management Information
- SNMP – Simple Network Management Protocol
- TCP – Transmission Control Protocol
- UDP – User Datagram Protocol
- WMI – Windows Management Instrumentation

Příloha C

Přehled nástrojů používaných pro SNMP monitorování počítačových sítí

SolarWinds Network Performance Monitor

Nástroj Network Performance Monitor od firmy SolarWinds je komplexní komerční řešení pro monitorování síťové infrastruktury. Nabízí pokročilé možnosti analýzy sítě, zobrazení topologie, nastavení alertování a další. Nicméně nejedná se o open-source řešení a cena za licenci je poměrně vysoká, licence pro 100 zařízení začíná na ceně €1.334 ve formě předplatného.

Zabbix

Zabbix je monitorovací open-source software určený pro dohled nad různými IT komponenty, což zahrnuje počítačové sítě, servery, virtuální stroje a cloudové služby. Jeho SNMP modul obsahuje řadu přednastavených šablon pro monitorování vybraných objektů u zařízeních od různých výrobců. Pokud jde o počet monitorovaných zařízení, nejsou zde žádná omezení.

Nagios Core

Nagios Core je open-source nástroj pro monitorování počítačových sítí a v nich poskytovaných služeb. Je vyvíjen primárně pro systém Linux. Nevýhodou je poměrně náročná konfigurace, která se provádí primárně přes příkazovou řádku. S uživatelsky dostupnějším rozhraním a rozšířením základních funkcí přichází nadstavba Nagios XI, který je již ale komerčním produktem, jenž ve verzi zdarma nabízí možnost monitorování pouhých 7 síťových hostů.

Icinga 2

Další dohledový systém, který zjišťuje dostupnost zdrojů v síti a informuje uživatele o výpadcích, se jmenuje Icinga. Tento systém vznikl jako fork projektu Nagios, se kterým sdílí

možnosti konfigurace. Základní distribuční balíček je poměrně prostý a nenabízí žádné moduly, které by byly schopny např. tvořit grafy. Proto je potřeba tyto moduly manuálně nainstalovat. Modularita je zároveň jedna z největších výhod tohoto systému, pro který existuje velké množství oficiálních i komunitně vytvořených modulů.

PRTG Network Monitor

PRTG Network Monitor je software dodávaný firmou Paessler AG, používaný k monitorování přepínačů, směrovačů, serverů a ostatních síťových zařízení či aplikací. PRTG Network Monitor disponuje módem *auto-discovery*, který skenuje předdefinované oblasti sítě a nově nalezená zařízení začne automaticky monitorovat. Cena licence se liší dle počtu senzorů, nikoliv dle počtu monitorovaných stanic. Jeden senzor, v pojetí výrobce softwaru, je jedna sledovaná veličina na kterémkoliv zařízení.

Cacti

Cacti je poměrně jednoduchý open-source nástroj, který je založen na monitorování pomocí protokolu SNMP. Vznikl jako nadstavba nástroje RRDTool. Podobně jako ostatní nástroje, Cacti udržuje svůj seznam monitorovaných zařízení, sleduje jejich dostupnost a v případě výpadku na něj dokáže upozornit např. prostřednictvím e-mailu. Nevýhodou je menší komunita uživatelů a omezené možnosti rozšíření s menším množstvím dostupných pluginů.

OpenNMS

OpenNMS je další z řady open-source nástrojů, vytvořený v programovacím jazyce Java. Je vyvinut a podporován skupinou OpenNMS Group. Cílem této skupiny je vytvoření škálovatelného a flexibilního řešení, vhodného pro uspokojení všech potřeb ohledně monitorování počítačové sítě či aplikací. Existuje pouze jedna verze zdarma, nicméně vývojáři nabízí placenou podporu aplikace. Mezi funkcemi, které tento nástroj nabízí je funkce discovery, kde po zadání IP rozsahu přidá monitorovatelná zařízení do svého seznamu. Dále nabízí monitorování dostupnosti síťových služeb, generování notifikací, pokud vybrané veličiny dosáhnou určité hodnoty, a další.

LibreNMS

LibreNMS je open-source nástroj, který původně vznikl jako fork nástroje Observium. LibreNMS disponuje funkcí autodiscovery, obsahuje propracovaný systém alertování, je schopen monitorovat vybrané aplikace s využitím *SNMP extend*. Také je schopen získávat logy aplikace *syslog* a vyhledávat v nich určité řetězce. V neposlední řadě podporuje pluginy systému Nagios a umí řadit zařízení do skupin dle vybraných parametrů, následně na skupiny automaticky aplikuje vytvořené šablony. Ze všech dostupných nástrojů jsem se nakonec rozhodl využít tento, protože nabízí značnou flexibilitu a zároveň splňuje všechny potřebné požadavky, které jsou kladeny na řešení této práce.

Příloha D

Proměnné z databáze MIB využívané ke klasifikaci

D.1 Skupina IP

Tabulka D.1: Proměnné databáze MIB ze skupiny IP [10]

OID	Název	Popis
1.3.6.1.2.1.4.9	ipInDelivers	The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).
1.3.6.1.2.1.4.3	ipInReceives	The total number of input datagrams received from interfaces, including those received in error.
1.3.6.1.2.1.4.10	ipOutRequests	The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission.
1.3.6.1.2.1.4.8	ipInDiscards	The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (e.g., for lack of buffer space).
1.3.6.1.2.1.4.5	ipInAddrErrors	The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity.

D.2 Skupina TCP

Tabulka D.2: Proměnné databáze MIB ze skupiny TCP [10]

OID	Název	Popis
1.3.6.1.2.1.6.5	tcpActiveOpens	The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.
1.3.6.1.2.1.6.6	tcpPassiveOpens	The number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.
1.3.6.1.2.1.6.7	tcpAttemptFails	The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.
1.3.6.1.2.1.6.8	tcpEstabResets	The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.
1.3.6.1.2.1.6.10	tcpInSegs	The total number of segments received, including those received in error. This count includes segments received on currently established connections.
1.3.6.1.2.1.6.11	tcpOutSegs	The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets.
1.3.6.1.2.1.6.15	tcpOutRsts	The number of TCP segments sent containing the RST flag.

D.3 Skupina UDP

Tabulka D.3: Proměnné databáze MIB ze skupiny UDP [10]

OID	Název	Popis
1.3.6.1.2.1.7.1	udpInDatagrams	The total number of UDP datagrams delivered to UDP users.
1.3.6.1.2.1.7.2	udpNoPorts	The total number of received UDP datagrams for which there was no application at the destination port.
1.3.6.1.2.1.7.3	udpInErrors	The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.
1.3.6.1.2.1.7.4	udpOutDatagrams	The total number of UDP datagrams sent from this entity.

D.4 Skupina ICMP

Tabulka D.4: Proměnné databáze MIB ze skupiny ICMP [10]

OID	Název	Popis
1.3.6.1.2.1.5.1	icmpInMsgs	Total number of ICMP messages which the entity received. Note that this counter includes all those counted by icmpInErrors.
1.3.6.1.2.1.5.14	icmpOutMsgs	Total number of ICMP messages which this entity attempted to send. Note that this counter includes all those counted by icmpOutErrors.
1.3.6.1.2.1.5.8	icmpInEchos	Number of ICMP Echo (request) messages received.
1.3.6.1.2.1.5.21	icmpOutEchos	The number of ICMP Echo (request) messages sent.
1.3.6.1.2.1.5.9	icmpInEchoReps	Number of ICMP Echo Reply messages received.
1.3.6.1.2.1.5.22	icmpOutEchoReps	The number of ICMP Echo Reply messages sent.
1.3.6.1.2.1.5.16	icmpOutDestUnreachs	The number of ICMP Destination Unreachable messages sent.